

Estudo das quartas feiras: Euler9

P. Kantek

8 de abril de 2025

Sumário

1	Problema 1: Soma de múltiplos de 3 ou 5 abaixo de 1000	13
1.1	Solução via Gemini	13
1.2	Solução em APL	13
1.3	Solução em J	14
1.4	Solução em Python	14
1.5	Solução em C++	15
1.6	Solução em Javascript	15
1.7	Solução em CLISP	15
1.8	Solução em Clipper/Harbour	16
1.9	Solução em GNU Octave	16
1.10	Solução em CPC	17
1.11	Solução em Maple	18
2	Problema 2: Soma dos termos pares de fibonacci menores que 4.000.000	19
2.1	Solução em APL	19
2.2	Solução em J	19
2.3	Solução em Python	20
2.4	Solução em C++	20
2.5	Solução em JS	20
2.6	Solução em CLISP	21
2.7	Solução em CPC	21
3	Problema 3: O maior fator primo de 600851475143	23
3.1	APL	23
3.2	J	24
3.3	Python	24
3.4	LISP	25
4	Problema 4: O maior número palindrômico obtido pelo produto de 2 fatores de 3 dígitos cada	26
4.1	APL	26
4.2	J	27
4.3	Python	27
4.4	CLISP	27
4.5	CPC	27
5	Problema 5: O menor inteiro que pode ser dividido por 1..20 sem deixar resto	29
5.1	APL	29
5.2	J	29
5.3	Python	30
5.4	CLISP	30
5.5	Octave	30
5.6	CPC	31
5.7	MAPLE	32

6	Problema 6: A diferença entre a soma 1..100 ao quadrado e a soma de cada elemento ao quadrado	33
6.1	APL	33
6.2	J	33
6.3	PYTHON	33
6.4	C++	34
6.5	Octave	34
6.6	CPC	34
6.7	MAPLE	35
7	Problema 7: O 10001º número primo	36
7.1	GEMINI	36
7.2	J	36
7.3	APL	36
7.4	PYTHON	37
7.5	C++	37
8	Problema 8: O maior produto de 4 dígitos adjacentes	39
8.1	APL	39
8.2	J	40
8.3	PYTHON	40
8.4	C++ com IA (Gemini)	40
9	Problema 9: tripla pitagoreana com $a + b + c = 1000$	42
9.1	IA: Gemini	42
9.2	APL	43
9.3	J	43
9.4	PYTHON	44
9.5	CLISP	44
9.6	OCTAVE	45
9.7	CPC	45
9.8	MAPLE	46
10	Problema 10: soma dos primos abaixo de 2 milhões	47
10.1	IA: Gemini	47
10.2	APL	47
10.3	J	47
10.4	PYTHON	48
10.5	C++	48
11	Problema 11: maior soma de 4 vizinhos em 9 direções	49
11.1	APL	49
11.2	J	50
12	Problema 12: Primeiro triangular com mais de 500 divisores	52
12.1	IA: Gemini	52
12.2	APL	52
12.3	J	53
12.4	PYTHON	53
12.5	CPC	54
13	Problema 13: 10 dígitos iniciais da soma de 100 números de 50 dígitos cada	56
13.1	APL	58
14	Problema 14: maior sequência de Collatz	59
14.1	APL	59
14.2	J	60
14.3	C++	60
15	Problema 15: Caminhos na treliça	61
15.1	APL	61
15.2	J	61
15.3	PYTHON	61

16 Problema 16: Soma dos dígitos de uma potência	63
16.1 APL	63
16.2 J	64
17 Problema 17: Contador do número de letras em nomes de números	65
18 Problema 18: Soma do caminho máximo I em triângulo numérico	66
18.1 APL	66
18.2 J	67
18.3 PYTHON	68
19 Problema 19: Quantos domingos caíram no 1º do mês	69
19.1 A enrolada do Gemini	69
19.2 APL	69
19.3 J	70
19.4 PYTHON	71
19.5 JS	71
19.6 LISP	72
19.7 CPC	72
20 Problema 20: Soma dos dígitos do fatorial de 100	75
20.1 J	75
20.2 Python	75
21 Problema 21: Soma dos números amigos abaixo de 10000	76
21.1 Solução em Gemini	76
21.2 Solução em APL	77
21.3 Solução em J	77
22 Problema 22 - Escores de nomes	79
22.1 Solução em APL	79
22.2 Solução em J	79
23 Problema 23: Soma de não abundantes	81
23.1 Solução em APL	81
23.2 Solução em J	82
24 Problema 24 - Qual a milionésima permutação de 0..9	84
24.1 Solução em APL	84
24.2 Solução em J	84
25 Problema 25 - qual o termo fibonacci a ter mais de 1000 dígitos	85
25.1 Solução em APL	85
25.2 Solução em J	86
26 Problema 26 - Maior ciclo recorrente de inversa	87
26.1 Solução em APL	87
26.2 Solução em J	88
27 Problema 27 - Maior ciclo de primos quadráticos	90
27.1 Solução em APL	90
28 Problema 28 - Soma das diagonais	92
28.1 Solução em APL	92
28.2 Solução em J	92
29 Problema 29 - Termos distintos em a*b e b*a	94
29.1 Solução em APL	94
29.2 Solução em J	96
30 Problema 30 - Números iguais à soma da potência quinta de seus dígitos	97
30.1 Solução em APL	97
30.2 Solução em J	98

31 Problema 31 - Quantas maneiras de obter 2 libras	99
31.1 Solução em APL	99
31.2 Solução em J	100
32 Problema 32 - Produtos pandigitais	101
32.1 Solução em APL	101
32.2 Solução em J	102
33 Problema 33 - Frações com cancelamento de dígitos	103
33.1 Solução em APL	103
33.2 Solução em J	104
34 Problema 34: Fatoriais dos dígitos	106
34.1 Solução em APL	106
34.2 Solução em J	106
35 Problema 35 - Primos circulares	108
35.1 Solução em APL	108
35.2 Solução em J	108
36 Problema 36 - Palíndromos em base dupla	110
36.1 Solução em APL	110
36.2 Solução em J	110
37 Problema 37 - Primos truncáveis	112
37.1 Solução em APL	112
37.2 Solução em J	112
38 Problema 38 - Múltiplos pandigitais	114
38.1 Solução em APL	114
38.2 Solução em J	115
39 Problema 39 - Triângulos retângulos inteiros	116
39.1 Solução em J	116
40 Problema 40 - Constante de Champernowne	117
40.1 Solução em APL	117
40.2 Solução em J	117
41 Problema 41 - Primo pandigital	119
41.1 Solução em APL	119
41.2 Solução em J	119
42 Problema 42 - Números triangulares codificados	121
42.1 Solução em APL	121
42.2 Solução em J	122
43 Problema 43 - Divisibilidade de substrings	123
43.1 Solução em APL	123
43.2 Solução em J	124
44 Problema 44 - Números pentagonais	125
44.1 Solução em APL	125
44.2 Solução em J	126
45 Problema 45 - Triangular, pentagonal e hexagonal	127
45.1 Solução em APL	127
45.2 Solução em J	127
46 Problema 46 - Outra conjectura de Goldbach	129
46.1 Solução em APL	129
46.2 Solução em J	130

47 Problema 47 - Fatores primos distintos	131
47.1 Solução em APL	131
47.2 Solução em J	132
48 Problema 48 - Auto potências	133
48.1 Solução em APL	133
48.2 Solução em J	135
49 Problema 49 - Permutações de primos	136
49.1 Solução em APL	136
49.2 Solução em J	136
50 Problema 50 - Soma de primos consecutivos	138
50.1 Solução em APL	138
50.2 Solução em J	139
50.3 Solução em MatLab	139
51 Problema 51 - Troca de dígitos em primos	140
51.1 Solução em Python	140
52 Problema 52 - Múltiplos permutados	142
52.1 Solução em APL	142
52.2 Solução em J	142
53 Problema 53 - Seleções combinatoriais	144
53.1 Solução em J	144
54 Problema 54 - Mãos de poker	146
54.1 Solução em APL	147
54.2 Solução em Javascript	148
54.3 Solução em Python	150
55 Problema 55 - Números de Lychrel	152
55.1 Solução em APL	152
56 Problema 56 - Potência dos dígitos somada	154
56.1 Solução em APL	154
56.2 Minha solução em J	154
57 Problema 57 - Convergência da raiz quadrada	156
57.1 Solução em Python	156
57.2 Solução em J	157
58 Problema 58 - Primos espirais	158
58.1 Solução em APL	158
58.2 Solução em J	160
59 Problema 59 - Decriptação XOR	161
59.1 Solução em APL	161
60 Problema 60 - Conjunto de pares de primos	163
60.1 Uma solução em Python	165
61 Problema 61 - Números figurativos cíclicos	166
61.1 Solução em APL	166
61.2 Solução em Python	169
62 Problema 62 - Permutações cúbicas	171
62.1 Solução em APL	171
62.2 Solução em J	172
63 Problema 63 - Contagens de dígitos poderosas	173
63.1 Solução em APL	173
63.2 Solução em J	173
63.3 Solução em Python	174

64 Problema 64 - Período ímpar de raízes quadradas	175
64.1 Solução em J	176
64.2 Solução em Python	176
65 Problema 65 - Convergência de e	178
65.1 Solução em Python	178
65.2 Solução em J	179
66 Problema 66 - Equação diofantina	181
66.1 Solução em Python	181
67 Problema 67 - Soma do caminho máximo II	183
67.1 Solução em APL	185
67.2 Solução em J	185
68 Problema 68 - Anel 5-gon mágico	186
68.1 Solução em Python	186
69 Problema 69 - Totiente máximo	188
69.1 Solução em J	188
70 Problema 70 - Permutação de totientes	189
70.1 Solução em Python	189
71 Problema 71 - Frações ordenadas	191
71.1 Soluções em J	191
71.2 Solução em Python	191
72 Problema 72 - Contando frações	192
72.1 Solução em Python	192
72.2 Solução em J	192
73 Problema 73 - Contando frações em intervalo	194
73.1 Solução em Python	194
73.2 Solução em J	195
74 Problema 74 - Cadeia de dígitos no fatorial	196
74.1 Solução em APL	196
74.2 Solução em J	197
75 Problema 75 - Triângulos retângulos inteiros singulares	198
75.1 Solução em Python	198
75.2 Solução em J	199
76 Problema 76 - Contando somas	200
76.1 Solução em Python	200
76.2 Solução em J	200
77 Problema 77 - Sumarizações de primos	202
77.1 Solução em Python	202
77.2 Solução em J	202
78 Problema 78 - Partições de moedas	204
78.1 Solução em Python	204
79 Problema 79 - Derivação de senha	206
80 problema 80 - Expansão digital da raiz quadrada	207
80.1 Solução em APL	207
80.2 Solução em Python	209
81 Problema 81 - Soma dos caminhos: duas maneiras	210
81.1 Solução em APL	210
81.2 Solução em J	210

82 Problema 82 - Soma do caminho: três caminhos	212
82.1 Solução em APL	212
82.2 Solução em J	213
83 Problema 83 - Soma do caminho: 4 caminhos	214
83.1 Solução em APL	214
83.2 Solução em Python	215
84 Problema 84 - Probabilidades do banco imobiliário	217
84.1 Solução em Python	218
85 Problema 85 - Contando retângulos	220
85.1 Solução em Python	220
85.2 Solução em J	221
86 Problema 86 - Rota cubóide	222
86.1 Solução em Python	222
87 Problema 87 - Triplas potências de primos	224
87.1 Solução em APL	224
88 Problema 88 - Números soma e produto	226
88.1 Solução em C++	227
88.2 Solução em APL	228
89 Problema 89 - Numerais romanos	229
89.1 Solução em APL	229
90 Problema 90 - Pares de dígitos do cubo	231
90.1 Solução em Python	231
91 Problema 91 - Triângulos retângulos com coordenadas inteiras	233
91.1 Solução em APL	235
91.2 Solução em J	235
92 Problema 92 - Correntes de dígitos quadrados	237
92.1 Solução em APL	237
92.2 Solução em J	238
93 Problema 93 - Expressões aritméticas	239
94 Problema 94 - Triângulos quase equiláteros	242
94.1 Solução em C++	244
94.2 Solução em APL	244
94.3 Solução em J	245
95 Problema 95 - Cadeias amigáveis	247
95.1 Solução em C++	247
95.2 Solução em APL	248
96 Problema 96 - Sudoku	250
96.1 Solução em APL	251
97 Problema 97 - Maior primo não mersenne	253
97.1 Solução em C++	253
97.2 Solução em APL	253
98 Problema 98 - Quadrados anagramáticos	255
98.1 Solução em APL	255
99 Problema 99 - Maior exponencial	257
99.1 Solução em APL	257
100 Problema 100 - Probabilidade arranjada	258
100.1 Solução em APL	258

101 Problema 101 - Polinômio ótimo	260
101.1 Solução em APL	263
102 Problema 102 - Contenção de triângulo	264
102.1 Solução em APL	264
103 Problema 103 - Somas de conjuntos especiais ótimas	266
103.1 Solução em Python	266
104 Problema 104 - Número de Fibonacci pandigital no fim	268
104.1 Solução em APL	269
104.2 Solução em J	269
105 Problema 105 - a resolver	271
106 Problema 106 - a resolver	272
107 Problema 107 - Rede mínima	273
107.1 Solução em APL	274
108 Problema 108 - Recíprocas diofantinas - I	276
108.1 Solução em Python	276
108.2 Solução em APL	276
108.3 Solução em J	277
109 Problema 109 - Dardos	278
109.1 Solução em Python	279
110 Problema 110 - Diofantinas Recíprocas II	280
110.1 Solução em Python	280
111 Problema 111 - Primos com corridas	281
111.1 Solução em Python	281
112 Problema 112 - Números rebotados	284
112.1 Solução em APL	284
112.2 Solução em J	285
113 Problema 113 - Números não rebotados	286
113.1 Solução em Python	286
114 Problema 114 - Combinações de blocos de contagem I	287
114.1 Solução em Python	288
114.2 Solução em J	288
115 Problema 115 - Combinações de blocos de contagem II	289
115.1 Solução em Python	289
116 Problema 116 - Azulejos vermelhos, verdes ou azuis	290
116.1 Solução em Python	290
116.2 Solução em J	291
117 Problema 117 - Azulejos vermelhos, verdes e azuis	292
117.1 Solução em Python	292
117.2 Solução em J	293
118 Problema 118 - Conjuntos primos pandigitais	294
118.1 Solução em Python	294
118.2 Solução em J	294
119 Problema 119 - Soma dos dígitos potenciados	296
119.1 Solução em J	296
119.2 Solução em Python	297

120 Problema 120 - Restos quadrados	298
120.1 Solução em Python	298
120.2 Solução em J	298
121 Problema 121 - Fundo de prêmios do jogo de disco	300
121.1 Solução em Python	300
122 Problema 122 - Exponenciação eficiente	302
122.1 Solução em APL	302
123 Problema 123 - Restos de quadrados primos	304
123.1 Solução em J	304
124 Problema 124 - Radicais ordenados	305
124.1 Solução em Python	305
124.2 Solução em J	305
125 Problema 125 - Somas palindrômicas	307
125.1 Solução em Python	307
125.2 Solução em J	307
126 Problema 126 - Camadas em cubóide	309

Este estudo: EULER em 11 ambientes

Este documento serve para registrar os avanços no estudo das quartas feiras. Trata-se da resolução de problemas do ambiente Euler (<https://projecteuler.net/>) usando diversas linguagens/ambientes. Cada uma delas é transcrita em uma cor diferente, a saber:

Inteligência artificial **Aqui não é uma linguagem de programação, e sim um engenho de IA. Vou me socorrer de 2: o gêmíni e o deepseek. O nome da cor é roxo**

APL **Soluções em APL. O nome da cor é vermelho.** Esta linguagem remonta à década de 60 do século passado. É antiga, mas AINDA está à frente do seu tempo. É a que gera menores programas fonte. Se não quiser me acreditar olhe a seguir. No problema 1 abaixo, ela resolveu o problema em 1 linha de código fonte. É inacreditável. Se eu tivesse que ir para uma ilha deserta levando apenas 1 linguagem, não tenha dúvida que seria esta.

No que diz respeito ao APL, há sempre pelo menos 2 abordagens radicalmente distintas. A primeira é *thinking APL*, ou pensando APL, que se traduz por abandonar loops e achar a resposta mediante uma ou 2 linhas de puro APL. A segunda é usar ferramentas da programação estruturada (blocos e loops) para resolver o mesmo problema. As soluções ficam maiores, mas em compensação compartilham idéias com Python, C++, JS entre outras. Usei as 2 abordagens mas com predomínio da segunda.

J **Soluções em J, sua cor é carmincla.** J é o maravilhoso sucessor de APL. Parece coisa do Harry Potter, tamanha é sua capacidade de concisão.

Python **Soluções em Python. O nome da cor é marinho.** Já o Python, ao contrário do APL, é uma das mais recentes (e modernas). Por conta disso, aproveitou muita coisa das boas características das linguagens que a precederam (inclusive do APL). 80% das universidades do mundo usam Python para ensinar as pessoas a programar. E, como se pode ver no gráfico TIOBE abaixo, é a linguagem mais usada no mundo nos dias atuais. Como é interpretada, pode deixar a desejar no quesito desempenho, mas quem liga para isso nestes tempos em que a unidade de aquisição de CPUs (e GPUs) é o milhão ?

C++ **Soluções em C++. O nome da cor é verdão.** Este aqui é o campeão de desempenho. Se velocidade é necessária, o C++ só perde para o Assembly, mas aí já é pedir demais.

JS **Soluções em Javascript. O nome da cor é rosa.** A grande vocação do Javascript – que a propósito, não tem nada a ver com Java, é aproveitar a interface com o operador do HTML (e CSS), gerando telas coloridas e bonitas de uma maneira fácil, padronizada e conhecida por todos os que usam a Internet, ou seja a grande maioria da humanidade. Nesta lista de linguagens, este sujeito é o que melhor resolveu o quesito interface.

LISP **Soluções em lisp. O nome da cor é azulcla.** O LISP não poderia faltar. Ainda que tenha ficado meio vendido na entrada da IA no mainstream da programação, seu estudo e uso ainda se justifica por ser uma linguagem funcional, com tudo o que este paradigma tem de bom. O uso espetacular da recursividade no âmbito da programação é o que há. Todo aquele que quer entender e usar recursividade ao natural, deve estudar e usar lisp. Vale aqui uma referência: com a palavra Leigh Caldwell, do Stack Overflow:

Os loops (interação) podem melhorar o desempenho de seu programa. A recursão melhora o desempenho do seu programador. Escolha o que for mais importante para a sua situação.

Harbour **Soluções em Clipper/Harbour. O nome da cor é amarelo.** O Harbour tem uma história antiga e acidentada. Começou com o dBaseII que foi uma pérola nos anos 80 ao permitir a implementação de bancos de dados em computadores muito pequenos. Era interpretado, e deu origem ao Clipper, quando passou a ser compilado. Muito software nos anos 90 se fez em Clipper. Na virada do milênio, o Clipper foi descontinuado, desconfio que por questões comerciais. Essa cessação deixou muita gente no pincel, que foi obrigada a migrar para outras soluções. Mas, uma parte dessas pessoas se engajou no desenvolvimento do Harbour que é freeware e que migrou o ambiente para processadores 64 bits, que são o estado da arte hoje.

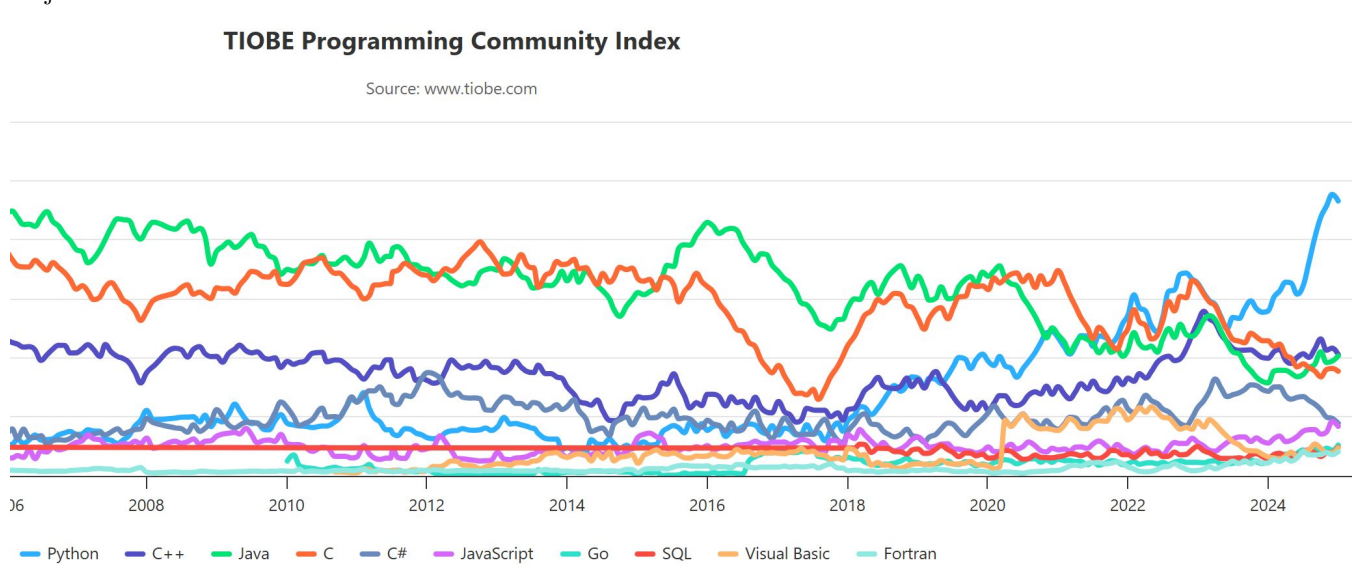
Octave **Soluções em GNU Octave. O nome da cor é ocre.** A origem deste ambiente é Matlab, um software muito bom que era e é usado em muitas escolas de engenharia mundo afora. Mas, tem um problema: uma licença de Matlab começa custando 1000 dólares e é portanto proprietária. A iniciativa Octave é uma das que se propuseram a migrar o Matlab para o mundo livre. Houve outras (como por exemplo Freemath, que começou a mil por hora, mas rapidamente perdeu ímpeto e hoje não se fala mais dela). Octave continua a ser desenvolvida, sua versão aqui mostrada é de dezembro de 2024, a 9.3.

CPC **Soluções em CPC. O nome da cor é laranja.** CPC é mais sobre compiladores do que sobre linguagens. Daí que mostro aqui ambos os programas: o fonte (em CPC) e o objeto (em APL). Para quem gosta destas coisas, é para ser uma delícia de olhar e estudar.

Maple **Soluções em Maple**. O nome da cor é **verde**. Finalmente, Maple aparece aqui por ter sido o precursor de processadores simbólicos. Também não é freeware, razão pela qual ficamos na versão V, que rola meio livre na Internet, mas que está muito longe da última versão paga que é a Maple 2024 (75 US\$/ano para estudantes e 1570 US\$/ano para profissionais)

Para materializar esta discussão vamos a um gráfico de uso das linguagens no mundo. A fonte é o TIOBE

(<https://www.tiobe.com/tiobe-index/>), que mostra as principais mês a mês e que se pode ver aqui, a listagem de janeiro de 2025:



Para contextualizar esse gráfico e essa discussão, note-se a queda de Java (que aliás não está neste estudo, eu nunca gostei de Java). Sua grande vantagem é a interconectividade, obtida graças ao conceito de máquina Java que é um meio caminho entre a linguagem interpretada e compilada.

Veja-se também o arranque imenso de Python, que em 2023 assume a ponta e daí não a larga mais.

Vale menção a C e seu sucessor o C++. Note que se eles forem somados (já que em certo enfoque eles são a mesma linguagem) ainda surgem muito fortes na corrida. Aliás, se somados C e C++, não sei se eles ganham ou perdem do Python (perdem em jan/2025, mas em compensação ganham se vier o reforço do **C#**), mas de qualquer maneira é por pouco. Para encerrar, ficam de fora deste estudo, algumas pérolas: COBOL (não acho que existam compiladores fáceis de achar e usar) e Pascal (idem, idem), além de Prolog e Natural, além de outras bem menos votadas.

Ambiente Euler

Este é um site (<https://projecteuler.net/>) que oferece problemas matemáticos e de programação de complexidades variadas (alguns são MUITO difíceis). A associação é gratuita, e a partir de um enunciado simples, o participante é convidado a achar a solução pedida (usualmente um número). Achada a resposta, ela é submetida ao site que responde SIM ou NÃO. Quando o participante acerta (SIM), ele ganha acesso a um fórum de todas as pessoas que resolveram este problema. É uma fonte riquíssima de idéias e insights para melhorar a solução. Aliás, os problemas tem como característica nunca exigir mais do que 1 segundo de CPU em computadores médios.

Um exemplo real: ao resolver o problema 105, encontrei a solução certa após 1 semana de CPU. Ao acessar o fórum, achei dicas que permitiram corrigir minha solução de 1 semana → 1 segundo de CPU.

O Projeto Euler foi iniciado por Colin Hughes (também conhecido como euler) em outubro de 2001 como uma subseção no mathschallenge.net. Desde então, a associação continuou a crescer e o Projeto Euler mudou para seu próprio domínio em 2006.

Infelizmente, há muitos lugares na internet que publicam as respostas e as soluções na forma de código para muitos problemas. Assim, a alegação de ter resolvido algum problema deve ser encarada com razoável ceticismo. Entretanto, já ouvi falar de empresas que durante o processo seletivo de mão de obra de programação, mandam o candidato resolver 2 ou 3 problemas Euler escolhidos. Uma sugestão que sempre dei aos meus alunos foi elaborar um dossiê com as respostas próprias achadas para os problemas Euler. Qualquer recrutador de programação sério deve se interessar bastante por um dossiê deste tipo.

Capítulo 1

Problema 1: Soma de múltiplos de 3 ou 5 abaixo de 1000

Múltiplos de 3 ou 5

Se listarmos os números naturais abaixo de 10 que são múltiplos de 3 ou de 5, nós obteremos 3, 5, 6 e 9. A soma destes múltiplos é 23. Ache a soma de todos os múltiplos de 3 ou 5 abaixo de 1000. A resposta é 233168.

1.1 Solução via Gemini

qual a soma dos multiplos de 3 ou 5 abaixo de 1000 ?

A soma dos múltiplos de 3 ou 5 abaixo de 1000 é 233168. Aqui está como você pode calcular isso:

1. Múltiplos de 3:

O maior múltiplo de 3 abaixo de 1000 é 999.

Divida 999 por 3 para obter o número de múltiplos: $999 / 3 = 333$.

Use a fórmula da soma de uma progressão aritmética: $S = n/2 * (a1 + an)$, onde n é o número de termos, $a1$ é o primeiro termo e an é o último termo.

$$S = 333/2 * (3 + 999) = 166833.$$

2. Múltiplos de 5:

O maior múltiplo de 5 abaixo de 1000 é 995.

Divida 995 por 5 para obter o número de múltiplos: $995 / 5 = 199$.

Use a fórmula da soma de uma progressão aritmética: $S = 199/2 * (5 + 995) = 99500$.

3. Múltiplos de 15 (para evitar contagem dupla):

O maior múltiplo de 15 abaixo de 1000 é 990.

Divida 990 por 15 para obter o número de múltiplos: $990 / 15 = 66$.

Use a fórmula da soma de uma progressão aritmética: $S = 66/2 * (15 + 990) = 33165$.

4. Some os múltiplos de 3 e 5 e subtraia os múltiplos de 15:

$$166833 + 99500 - 33165 = 233168.$$

Portanto, a soma dos múltiplos de 3 ou 5 abaixo de 1000 é 233168.

1.2 Solução em APL

Primeiro uma solução ao gosto do APL, ou como se diz *thinking APL*. Note que a linguagem é tão escoreita que nem precisa definir uma função.

```
+ / ((0=5 | a) v (0=3 | a)) / a ← 1999
233168
```

Agora, uma solução trivial, usando as instruções de programação estruturada do APLX.

```
∇ eu001; soma; i
[1] soma ← 0
[2] :For i :In 1999
[3]   :If ((0=3 | i) v (0=5 | i))
[4]     soma ← soma + i
[5]   :EndIf
[6] :EndFor
[7] soma ∇
```

Eventual solução em APL canônico seria um pouco diferente, como está na definição de eu001c, a seguir

```
∇ eu001c; soma; i
[1] soma ← 0
[2] i ← 1
[3] testa → (i > 999) / fim
[4] → ((0=3 | i) v (0=5 | i)) / ad
[5] incr: i ← i + 1
[6] → testa
[7] ad: soma ← soma + i
[8] → incr
[9] fim: soma
∇
```

1.3 Solução em J

```
+ / ~. (3 * i. 334), 5 * i. 200
```

1. Primeiro é gerado o vetor de múltiplos de 5, menores que 1000, pela expressão `5 * i. 200`, que é 0, 5, 10, 15, 20, 25, 30, 35, 40, ... 995.
2. Depois (veja os parênteses) é gerado o vetor de múltiplos de 3, menores que 1000 pela expressão `3 * i. 334` que é 0, 3, 6, 9, 12, 15, 18, 21, 24, ... 999.
3. A seguir, os dois vetores acima são concatenados, pela `,`.
4. Depois, os duplicados são eliminados, por `~`.
5. Finalmente o resultado é somado, obtendo-se o resultado esperado.

1.4 Solução em Python

Não é necessário usar nenhum pacote especial.

```
def eu001():
    soma=0
    for i in range(1,1000):
        if ((i%3)==0) or ((i%5)==0):
            soma=soma+i
    print('a soma e ',soma)
eu001()
```

1.5 Solução em C++

Eis uma solução trivial.

```
#include<iostream>
using namespace std;
int main(){
    int soma=0;
    int i;
    for (i=1;i<1000;i++){
        if (((i%3)==0) || ((i%5)==0)) {
            soma=soma+i;
        }
    }
    cout<<soma;
}
```

1.6 Solução em Javascript

Esta é uma solução envolvendo HTML+JS+CSS. A conferir

```
<html><head><title>Euler 001</title>
<meta charset="UTF-8">
<script type="text/javascript">
function eu001() {
    soma=0
    for(i=1;i<1000;i++) {
        if ((0==i%3) || (0==i%5)) {
            soma=soma+i
        }
    }
    document.getElementById('resp').value=soma
}
</script>

</head><body>
<h1>Cálculo do problema euler001</h1>
<form>
<button type="button" onclick="eu001()"> Calcular </button>
<input type="text" id="resp" size="40" readonly> </form>
</body></html>
```

1.7 Solução em CLISP

Aqui a coisa é mais artesanal. Vamos lá

Primeiro é preciso definir um programa com o nome de **eu001.lsp** e com o seguinte conteúdo:

```
(defun eu001()
  (let ((n 0))
    (dotimes (i 1000)
      (when (or (zerop (mod i 5))(zerop (mod i 3)))
        (incf n i)))
    n)
)
```

salvando-o em uma biblioteca à qual o Clisp tenha acesso. No caso foi no mesmo diretório onde está o executável da linguagem. (**\xli sp**). Depois, deve-se carregar o programa XLISP e estando dentro dele, emitir

```
(load 'common)
(load 'eu001)
(eu001)
233168
```

1.8 Solução em Clipper/Harbour

Só para diminuir a saudade deste ambiente incrível (Harbour). Esse cara só aparece aqui por puro saudosismo. Afinal para resolver este problema pequeno, ele gerou um executável de 1.9MB. É demais, mas lembremos que a vocação do Harbour é gerar programas que manuseiem grandes bancos de dados, e nessa hora ele é muito (muito !) bom. Mas, vamos lá

Primeiro, cria-se o módulo abaixo salvando-o com o nome de EU001.PRG

```
#include "INKEY.CH"
#include "FILEIO.CH"
Function main()
  clear
  i := 1
  soma := 0
  do while i<1000
    if (0==i%3) .or. (0==i%5)
      soma := soma + i
    endif
    i++
  enddo
  ? soma
return nil
```

Dai, cria-se um outro módulo de nome EU001.HBP, contendo

```
-oeu001
eu001.prg
```

Finalmente, em uma janela DOS

```
c:/hb32/setarhb -- barras invertidas, por favor
cd/p/quarta -- o diretório do projeto
hbm2 eu001.hbp
```

E agora o harbour cria o eu001.exe que ao ser executado

```
c:/p/quarta> eu001 <enter>
233168
```

1.9 Solução em GNU Octave

Primeiro deve-se criar o módulo abaixo e salvá-lo com o nome de eu001.m em um diretório ao qual o Octave tenha acesso. Para isso, modifique a biblioteca de carga no alto à esquerda da tela do Octave. Por enquanto, a relação é 1 função = 1 arquivo.

Se você quiser colocar várias funções no mesmo arquivo, é possível, mas a chamada agora é `arquivo.funcao(XXX)` para que ele consiga se localizar. Nesta chamada o `arquivo` é escrito sem a terminação `.m`

```
function eu001 % salvo como eu001.m
i=1;
soma=0;
```

```

while (i<1000)
  if (0==mod(i,3)) || (0==mod(i,5))
    soma = soma + i;
  end
  i=i+1;
end
disp(soma)
end

```

Depois, dentro do Octave, basta comandar

```

>> eu001
233168
>>

```

1.10 Solução em CPC

Esta solução aparece aqui para lembrar e exercitar o uso de uma linguagem que eu criei quando era professor de programação e precisava gerar código nas folhas de exercícios e depois corrigi-lo com esforço menor. A propósito, CPC significa Compilador em Pseudo-Código. Como é minha criação, expus neste texto o código fonte (em CPC) e o código objeto (em APLX).

```

)load 2 cpc

algoritmo eu001
i <- 1
soma <- 0
enquanto i<1000
  se (0=3|i) v (0=5|i)
    soma <- soma+i
  fim{se}
  i <- i+1
fim{enquanto}
imprima soma
fim{algoritmo}

```

que deu origem a

```

eu001                                ; i ; soma ; Δ001 ; Δ002 ; Δ003 ; Δ004 ; Δ005 ; Δ006
i←1
soma←0
Δ001:
Δ003←i<1000
→(∼Δ003)/Δ002
Δ004←(0=3|i) v (0=5|i)
→(∼Δ004)/Δ005
soma←soma+i
Δ005:
i←i+1
→Δ001
Δ002:
soma

```

e daí

```

      f x cpc a
eu001
      eu001
233168

```

1.11 Solução em Maple

Vai-se usar aqui a versão 5 do Maple, já que é aquela que está circulando livremente pela Internet. Vale ressaltar que o Maple não é freeware. Cria-se o módulo abaixo, com um editor qualquer (notepad++ ?) e salva-se-o com o nome de fulano.mpl

```
eu001 := proc()
  local i,soma;
  i := 1;
  soma := 0;
  while i<1000 do
    if (0=(i mod 3)) or (0=(i mod 5)) then
      soma := soma + i;
    fi;
    i := i + 1;
  od;
  print(soma);
end;
```

Daí, dentro do Maple, faz-se

```
read"c:/p/quarta/eu001.mpl";
eu001();
233168
```

Capítulo 2

Problema 2: Soma dos termos pares de fibonacci menores que 4.000.000

Cada novo termo na sequência de Fibonacci é gerado pela adição dos dois termos anteriores. Começando com 1 e 2, os primeiros 10 termos são

1, 2, 3, 5, 8, 13, 21, 34, 55, 89

Considerando os termos da sequência de Fibonacci que não excedem a 4 milhões, ache a soma dos termos que são pares. A resposta é 4613732.

2.1 Solução em APL

```
▽eu002;soma;x;next
[1] soma←2
[2] x←1 2
[3] :While x[px]<4000000
[4]     next←x[-1+px]+x[px]
[5]     :If 0=2|next
[6]         soma←soma+next
[7]     :EndIf
[8]     x←x,next
[9] :EndWhile
[10] soma
▽ 2025-02-01 15.23.34
```

2.2 Solução em J

```
euler2 =: 3 : 0
  a =. 1
  b =. 2
  soma =. 2
  while. b<4000000 do.
    next =. a+b
    if. 0=2|next do.
      soma=.soma+next
    end.
    a =. b
    b =. next
  end.
  soma
)
```

Depois dentro do J:

```

load 'c:/p/quarta/euler2.ijs'
euler2 ''
4613732

```

Deve-se notar, na chamada, a passagem de um vetor vazio ao programa. Se você chamar simplesmente o programa (`euler2`, no caso) o J vai listar o programa. Outra observação é que esta solução é a tradução simples do enunciado. Ainda não é pensar J.

2.3 Solução em Python

```

def eu002():
    f=[1,2]
    soma=2
    while f[len(f)-1]<4000000:
        pro=f[len(f)-2]+f[len(f)-1]
        if 0==pro%2:
            soma=soma+pro
        f=f+[pro]
    print(soma)
eu002()

```

2.4 Solução em C++

```

#include<iostream>
using namespace std;
int main(){
    int soma=2;
    int a,b,c;
    a=1;
    b=2;
    c=3;
    while (c<4000000){
        if (0==c%2){
            soma=soma+c;
        }
        a=b;
        b=c;
        c=a+b;
    }
    cout<<soma;
}

```

2.5 Solução em JS

```

<html><head><title>Euler 002</title>
<meta charset="UTF-8">
<script type="text/javascript">
function eu002(){
    soma=2
    a=1
    b=2
    c=3
    while (c<4000000){
        if (0==c%2){

```

```

    soma=soma+c;
}
a=b;
b=c;
c=a+b;
}
document.getElementById('resp').value=soma
}
</script>
</head><body>
<h1>Cálculo do problema euler002</h1>
<form>
<button type="button" onclick="eu002()"> Calcular </button>
<input type="text" id="resp" size="40" readonly> </form>
</body></html>

```

2.6 Solução em CLISP

```

(defun eu002()
  (do ((n 2 (+ n nm1)) (nm1 1 n) (sum 0))
      ((> n 4000000) sum)
      (when (evenp n) (incf sum n))
  )
)

```

2.7 Solução em CPC

```

algoritmo eu002
  soma <- 2
  a <- 1
  b <- 2
  c <- 3
  enquanto c<4000000
    se 0=2|c
      soma <- soma+c
    fim{se}
    a <- b
    b <- c
    c <- a+b
  fim{enquanto}
  imprima soma
fim{algoritmo}

eu002
soma+2
a+1
b+2
c+3
Δ001:
Δ003←c<4000000
→(∼Δ003)/Δ002
Δ004←0=2|c
→(∼Δ004)/Δ005
soma←soma+c
Δ005:
a←b
b←c
c←a+b
; soma;b;c;Δ001;Δ002;Δ003;Δ004;Δ005;Δ006

```

→Δ001
Δ002:
soma

Capítulo 3

Problema 3: O maior fator primo de 600851475143

Os fatores primos de 13195 são 5, 7, 13 e 29. Qual é o maior fator primo do número 600851475143 ? A resposta é 6857.

3.1 APL

```

    ▽r←divisores x;i
[1]  r←1,x
[2]  i←2
[3]  :While i≤[x*0.5
[4]      :If 0=i|x
[5]          r←r,i
[6]          :If x≠i*2
[7]              r←r,x÷i
[8]          :EndIf
[9]      :EndIf
[10]  i←i+1
[11] :EndWhile
[12] r←r[⍋r]
    ▽ 2025-02-03 10.35.08
```

```

    ▽eu003;x;y;i;salva
[1]  x←600851475143
[2]  y←x*0.5
[3]  :For i :In 1↓⍳[y
[4]      :If (0=i|x)^(isprime i)
[5]          salva←i
[6]      :EndIf
[7]  :EndFor
[8]  salva
    ▽ 2025-02-03 9.48.38
```

```

    ▽r←isprime x;z;f
[1]  :If x=1
[2]      r←0
[3]      :Return
[4]  :EndIf
[5]  :If x<4
[6]      r←1
[7]      :Return
[8]  :EndIf
[9]  :If 0=2|x
[10]     r←0
[11]     :Return
[12] :EndIf
[13] :If x<9
```

```

[14]      r←1
[15]      :Return
[16]      :EndIf
[17]      :If 0=3|x
[18]      r←0
[19]      :Return
[20]      :EndIf
[21]      z←⌊x*0.5
[22]      f←5
[23]      :While f≤z
[24]      :If 0=f|x
[25]      r←0
[26]      :Return
[27]      :EndIf
[28]      :If 0=(f+2)|x
[29]      r←0
[30]      :Return
[31]      :EndIf
[32]      f←f+6
[33]      :EndWhile
[34]      r←1 ▽

```

3.2 J

```

>./,2 p: 600851475143
6857

```

3.3 Python

```

import math
def isprime(x):
    if x==1:
        return False
    if x<4:
        return True
    if 0==x%2:
        return False
    if x<9:
        return True
    if 0==x%3:
        return False
    z=math.floor(x**0.5)
    f=5
    while f<=z:
        if 0==x%f:
            return False
        if 0==x%(f+2):
            return False
        f=f+6
    return True

def eu003():
    x=600851475143
    y=math.ceil(x**0.5)
    for i in range(2,y):
        if x%i==0 and isprime(i):
            salva=i
    print(salva)

```

eu003()

É de se notar que os fatores (primos e não primos) do número dado são 1, 71, 839, 1471, 6857, 59569, 104441, 486847, 1234169, 5753023, 10086647, 87625999, 408464633, 716151937, 8462696833 e 600851475143. Destes, o maior primo é 6857.

3.4 LISP

```
(defun primfac (num)
  (when (> num 1)
    (do ((x 2 (1+ x)))
        ((zerop (mod num x))
         (cons x (primfac (/ num x)))))))
> (load'eu003)
> (primfac 600851474143)
(71 839 1471 6857)
```

Capítulo 4

Problema 4: O maior número palindrômico obtido pelo produto de 2 fatores de 3 dígitos cada

Um número palindrômico é aquele que tem o mesmo valor se lido nas duas direções. O maior palindrômico obtido pelo produto de 2 números de 2 dígitos cada é $9009 = 91 \times 99$. Ache o maior palindrômico obtido a partir do produto de dois números de 3 dígitos. A resposta é 906609.

4.1 APL

Primeiro a solução em *thinking APL*.

```
∇r←pal4 x;y
[1] y←⊖x
[2] r←x=⊖y
∇

∇eu004c;a;i;b
[1] b←b[⊖b←,a°.x←99+ι900]
[2] i←1
[3] oba:→(pal4 b[i])/fim
[4] i←i+1
[5] →oba
[6] fim:b[i]
∇ 2025-02-03 10.03.41
906609
```

Agora a solução APL estruturada:

```
∇eu004;x;maio;z;mx;mz
[1] x←999
[2] maio←-99999
[3] t1:z←999
[4] t2:→(1=pal4 x×z)/achou
[5] vorta:→(z≤x)/diminui
[6] →(z<100)/diminui
[7] z←z-1
[8] →t2
[9] diminui:x←x-1
[10] →(x<100)/ufa
[11] →t1
[12] achou:→(maio<x×z)/troca
[13] →vorta
[14] troca:maio←x×z
[15] mx←x
[16] mz←z
```

```
[17] →vorta
[18] ufa:maio,mx,mz
      ▽
```

4.2 J

```
>./(#~ (-: |.)@":"0) ,/*/~i.1000
906609
```

4.3 Python

```
def pal4(x):
    y=str(x)
    z=''.join(reversed(y))
    return int(z)==x
def eu004():
    maior=-9999999
    for i in range(100,1000):
        for j in range(100,1000):
            if pal4(i*j):
                if (i*j)>maior:
                    maior=i*j
                    si=i
                    sj=j
    return maior,si,sj
print(eu004())
```

4.4 CLISP

```
(defun palindromep (val)
  (let ((str (format nil "~s" val)))
    (equal str (reverse str))
  )
)

(defun euler4 ( &aux (maxfound 0))
  (do ((v1 999 (1- v1)))
    ((or (< v1 100) (< (* v1 v1) maxfound))
     maxfound)
    (do* ((v2 v1 (1- v2))
          (product (* v1 v2) (* v1 v2)))
      ((or (< v2 100) (<= product maxfound)))
      (when (palindromep product)
        (setf maxfound product)
      )
    )
  )
)

(euler4)
```

4.5 CPC

Para isto chama-se o compilador CPC, através de `)load 2 cpc` que é traduzido para `)load '/p/aplx/cpc.aws'`

```

algoritmo eu004
maior <- -9999999
para i de 100 ate 999
  para j de 100 ate 999
    p <- i×j
    se p = ±0∓p
      se p > maior
        maior <- p
        mi <- i
        mj <- j
      fim{se}
    fim{se}
  fim{para}
fim{para}
imprima maior, mi, mj
fim{algoritmo}

```

```

eu004 ;maior;p;mi;mj;Δ001;Δ002;Δ003;Δ004;Δ005;Δ006;Δ007;Δ008;Δ009;Δ010;Δ011
maior←-9999999
i      ←100
Δ001:
Δ003←i≤999
→(¬Δ003)/Δ002
j      ←100
Δ004:
Δ006←j≤999
→(¬Δ006)/Δ005
p←i×j
Δ007←p=±0∓p
→(¬Δ007)/Δ008
Δ009←p>maior
→(¬Δ009)/Δ010
maior←p
mi←i
mj←j
Δ010:
Δ008:
j      ←j      + 1.00000
→Δ004
Δ005:
i      ←i      + 1.00000
→Δ001
Δ002:
maior,mi,mj

```

Capítulo 5

Problema 5: O menor inteiro que pode ser dividido por 1..20 sem deixar resto

2520 é o menor número que pode ser dividido pelos números de 1 a 10 sem deixar nenhum resto. Qual é o menor positivo inteiro que é divisível pelos números de 1 a 20 sem deixar nenhum resto ? A resposta é 232792560.

5.1 APL

Primeiro a solução em thinking APL, aplicando a definição *as is*, sem pensar muito

```
∇eu005
[1] x←2520
[2] :While 0≠+/(ι20)|x
[3]   x←x+1
[4] :EndWhile
[8] x
∇ 2025-02-03 10.45.35
232792560
```

Só que este programa demora muito. Pensando mais, e lembrando que o número procurado (por hipótese y) precisa ser múltiplo de todos os primos entre 1 e 20 inclusive, basta procurar um outro número múltiplo de y que divida a série 1..20 sem deixar resto. A busca é rápida, já que $y \times 24$ é o número procurado.

```
∇eu005b;x;i;y
[1] y←x←x/2 3 5 7 11 13 17 19
[2] i←2
[3] testa:→(0=+/(ι20)|y)/achou
[4] y←x×i
[5] i←i+1
[6] →testa
[7] achou:y
∇ 2025-02-03 10.56.52
232792560
```

5.2 J

```
*./1+i.20
232792560
```

5.3 Python

```
def acha(n):
    lista=[0]*20
    for i in range(1,21):
        lista[i-1]=n%i
    return lista
def eu005b():
    y=x=2*3*5*7*11*13*17*19
    i=2
    while 0 != sum(acha(y)):
        y=x*i
        i=i+1
    print(y)
def eu005():
    x=2520
    while 0 !=sum(acha(x)):
        x=x+1
    print(x)
eu005() % =lento ou eu005b()=rápido
232792560
```

5.4 CLISP

```
(defun eu005()(
  (setf v nil)
  (dotimes (i 20) (push (1+ i) v))
  (apply #'lcm v)
) ; este arquivo é salvo como eu005.lsp no mesmo diretório do xlisp

(load'common)
(load'eu005)
(eu005)
232792560
```

5.5 Octave

```
function eu005() % salvo com o nome de eu005.m
    x=2*3*5*7*11*13*17*19;
    y=x;
    ind=2;
    while 0 != achav(y)
        y = x*ind;
        ind = ind+1;
    end
    disp(int64(y))
end
function res = achav(n)
    saida = 0;
    for me = 1:20
        saida = saida + mod(n, me);
    end
    res = saida;
end
% acabou o arquivo eu005.m
>> eu005
232792560
```

>>

5.6 CPC

```
funcao acha n
vetor lista[20]
lista <- 20p0
para ijk de 1 ate 20
  lista[ijk] <- ijk|n
fimpara
RRRR <- lista
fimfuncao

algoritmo eu005
x <- x/2,3,5,7,11,13,17,19
y <- x
i <- 2
enquanto 0 ≠ (+/acha ,y)
  y <- x×i
  i <- i+1
fim{enquanto}
imprima y
fim{algoritmo}

RRRR←acha AAAA          ;n;lista;RRRR;Δ001;Δ002;Δ003;Δ004
n←AAAA[ 1]
lista←20p0
lista←20p0
ijk      ←1
Δ001:
Δ003←ijk≤20
→(∼Δ003)/Δ002
lista[ijk]←ijk|n
ijk      ←ijk      + 1.00000
→Δ001
Δ002:
RRRR←lista

eu005          ;x;y;i;Δ001;Δ002;Δ003;Δ004
x←x/2,3,5,7,11,13,17,19
y←x
i←2
Δ001:
Δ003←0≠(+/acha,y)
→(∼Δ003)/Δ002
y←x×i
i←i+1
→Δ001
Δ002:
y

      □fx cpc cpceu005
acha
      □fx cpc cpceu005b
eu005
      eu005
232792560
```

PS: Lembre que por uma deficiência do ambiente CPC funções chamantes e chamadas não podem compartilhar variáveis

com o mesmo nome. É por essa razão que o índice em *acha* chama-se *ijk* e não apenas *i* como seria de se esperar. O conserto é fácil, basta deixar todas as variáveis da função chamada como locais. Fica na lista para ser feito algum dia.

5.7 MAPLE

```
achav := proc(x)
  local i,saida;
  saida := 0;
  for i from 1 to 20 do
    saida := saida + modp(x,i);
  od;
end:
eu005 := proc()
  local x,y,ind;
  x := 2*3*5*7*11*13*17*19;
  y := x;
  ind := 2;
  while (0 <> achav(y)) do
    y := x*ind;
  ind := ind + 1;
  od;
  y;
end:

> read"c:/p/quarta/eu005.mpl";
> eu005();
232792560
>
```

Capítulo 6

Problema 6: A diferença entre a soma 1..100 ao quadrado e a soma de cada elemento ao quadrado

A soma dos quadrados dos primeiros 10 números naturais é

$$1^2 + 2^2 + 3^2 + \dots + 10^2 = 385$$

O quadrado da soma dos dez primeiros naturais é

$$(1 + 2 + 3 + \dots + 10)^2 = 55^2 = 3025$$

Aqui a diferença entre a soma dos quadrados dos primeiros 10 naturais e o quadrado de sua soma é $3025 - 385 = 2640$. Ache a diferença entre a soma dos quadrados e o quadrado da soma dos primeiros 100 naturais. A resposta é 25164150.

6.1 APL

Insuperável, como sempre

```
((+/⍳100)*2)-+/(⍳100)*2
25164150
```

6.2 J

```
(([: *: +/)-([: +/ *:)) (i.101)
25164150
```

6.3 PYTHON

```
soma1=0
for i in range(1,101):
    soma1 = soma1+i
soma1=soma1**2
soma2=0
for i in range(1,101):
    soma2=soma2+i**2
print(soma1-soma2)

25164150
```

Um jeito mais pythonista de resolver o mesmo problema

```

num=list(range(1,101))
qua=[numero**2 for numero in num]
(sum(range(1,101))**2)-sum(qua)
25164150

```

ou em 1 linha só

```

(sum(range(1,101))**2)-sum([num**2 for num in list(range(1,101))])
25164150

```

6.4 C++

```

#include<iostream>
using namespace std;
int main(){
    long int s1=0,s2=0;
    int i;
    for(i=1;i<101;i++){
        s1=s1+i*i;
        s2=s2+i;
    }
    s2=s2*s2;
    cout<<s2-s1;
}

```

```

25164150

```

6.5 Octave

```

function eu006() %salvo como eu006.m
    s1=0;
    s2=0;
    for i = 1:100
        s1=s1+i*i;
        s2=s2+i;
    end
    s2=s2*s2;
    disp(int64(s2-s1));
end

```

```

>> eu006
25164150

```

6.6 CPC

```

        cpceu006
algoritmo eu006
s1 <- 0
s2 <- 0
para i de 1 ate 100
    s1 <- s1 + (i*2)
    s2 <- s2 + i
fimpara

```

```

s2 <- s2*2
imprima s2-s1
fimalgoritmo
    cpc cpceu006
eu006          ;s1;s2;Δ001;Δ002;Δ003;Δ004
s1←0
s2←0
i          ←1
Δ001:
Δ003←i≤100
→(¬Δ003)/Δ002
s1←s1+(i*2)
s2←s2+i
i          ←i          + 1.00000
→Δ001
Δ002:
s2←s2*2
s2-s1
eu006
25164150

```

6.7 MAPLE

```

eu006 := proc() # salvo como eu006.mpl
    local s1,s2,i;
    s1:=0;
    s2:=0;
    for i from 1 to 100 do
        s1:=s1+i*i;
        s2:=s2+i;
    od;
    s2:=s2*s2;
    print(s2-s1);
end:

> read"c:/p/quarta/eu006.mpl";
> eu006();
25164150

```

Capítulo 7

Problema 7: O 10001º número primo

Quando se lista os primeiros 6 números primos se obtém

2, 3, 5, 7, 11, 13

e pode-se ver que o sexto número primo é 13. Qual é o 10001º número primo ? A resposta é 104743.

7.1 GEMINI

```
qual é o 10001 número primo ?  
104.743
```

7.2 J

```
p:10000  
104743
```

7.3 APL

Usando APL raiz

```
▽eu007[⍳]▽  
▽eu007 onde;i;qual  
[1] i←1  
[2] qual←1  
[3] pula:→(qual>onde)/acabou  
[4] →(~isprime i)/nao  
[5] qual←qual+1  
[6] nao:i←i+1  
[7] →pula  
[8] acabou:i-1  
▽ 2025-02-04 9.47.45  
eu007 6  
13  
eu007 10001  
104743
```

Agora usando programação estruturada

```
▽eu007b[⍳]▽
```

```

    ∇eu007b onde;i;qual
[1]   i←qual←1
[2]   :While qual≤onde
[3]       :If isprime i
[4]           qual←qual+1
[5]       :EndIf
[6]       i←i+1
[7]   :EndWhile
[8]   i-1
∇ 2025-02-04 9.54.33

    eu007b 6
13
    eu007b 10001
104743

```

7.4 PYTHON

Este código está dentro do módulo EU003.PY, para compartilhar a função `isprime`.

```

def eu007(onde):
    i=1
    qual=1
    while qual <= onde:
        if isprime(i):
            qual=qual+1
        i=i+1
    print(i-1)
eu007(6)
13
eu007(10001)
104743

```

7.5 C++

```

#include<iostream>
#include<cmath>
using namespace std;
int isprime(int x){
    int z,f;
    if (x==1){return 0;}
    if (x<4){return 1;}
    if (0==(x%2)){return 0;}
    if (x<9){return 1;}
    if (0==(x%3)){return 0;}
    z=floor(sqrt(x));
    f=5;
    while (f<=z){
        if (0==(x%f)){ return 0;}
        if (0==(x%(f+2))){return 0;}
        f=f+6;
    }
    return 1;
}

int main(){
    int i=1,qual=1,onde;
    cin>>onde;

```

```
while (qual<=onde){
    if (isprime(i)){
        qual++;
    }
    i++;
}
cout<<i-1;
}
```

Capítulo 8

Problema 8: O maior produto de 4 dígitos adjacentes

Os 4 dígitos adjacentes no número de 1000 dígitos que têm o maior produto são $9 \times 9 \times 8 \times 9 = 5832$.

73167176531330624919225119674426574742355349194934
96983520312774506326239578318016984801869478851843
85861560789112949495459501737958331952853208805511
12540698747158523863050715693290963295227443043557
66896648950445244523161731856403098711121722383113
62229893423380308135336276614282806444486645238749
30358907296290491560440772390713810515859307960866
70172427121883998797908792274921901699720888093776
65727333001053367881220235421809751254540594752243
52584907711670556013604839586446706324415722155397
53697817977846174064955149290862569321978468622482
83972241375657056057490261407972968652414535100474
82166370484403199890008895243450658541227588666881
16427171479924442928230863465674813919123162824586
17866458359124566529476545682848912883142607690042
24219022671055626321111109370544217506941658960408
07198403850962455444362981230987879927244284909188
84580156166097919133875499200524063689912560717606
05886116467109405077541002256983155200055935729725
71636269561882670428252483600823257530420752963450

Ache o maior produto de 13 dígitos adjacentes nesse número. A resposta é 23514624000.

8.1 APL

```

Veu008;i;j;s;maior;quale
[1] i←1
[2] maior←-999999
[3] :While i<987
[4]   j←1
[5]   s←1
[6]   :While j<14
[7]     s←s×±N8[i+j]
[8]     j←j+1
[9]   :EndWhile
[10]  :If s>maior
[11]    quale←i
[12]    maior←s
[13]  :EndIf
[14]  i←i+1
[15] :EndWhile
```

```
[16] maior
[17] quale
    ▽ 2025-02-05 9.47.46
```

```
23514624000
197
```

8.2 J

Primeiro é criada a variável *y* contendo a literal, depois... Nota: vai verbatim porque o alltt estava comendo uma barra muito importante.

```
>./>([:*/10 10 10 10 10"_ #:[:".])&.>5<\y
```

8.3 PYTHON

```
def eu008():
    a='7316717653133062491922511...
```

```
600823257530420752963450'
    i=0
    maior=-99999
    while i<986:
        j=0
        s=1
        while j<13:
            s=s*int(a[i+j])
            j=j+1
        if s>maior:
            quale=i
            maior=s
        i=i+1
    print(maior,quale)
eu008()
```

```
23514624000 197
```

8.4 C++ com IA (Gemini)

Com a ajuda do gemini, a partir da seguinte consulta *escreva um código em C++ que processe um string de 1000 bytes usando o número que cada caracter representa* e ele escreveu o código que funcionou de prima.

```
#include <iostream>
#include <string>
#include <vector>
```

```
int main() {
    std::string minhaString(1000, ' '); // Cria uma string com 1000 espaços em branco
    minhaString={"73167176531330624919225119674426574742355349194934\
96983520312774506326239578318016984801869478851843\
85861560789112949495459501737958331952853208805511\
12540698747158523863050715693290963295227443043557\
66896648950445244523161731856403098711121722383113\
62229893423380308135336276614282806444486645238749\
30358907296290491560440772390713810515859307960866\
70172427121883998797908792274921901699720888093776\
65727333001053367881220235421809751254540594752243\
```

```

52584907711670556013604839586446706324415722155397\
53697817977846174064955149290862569321978468622482\
83972241375657056057490261407972968652414535100474\
82166370484403199890008895243450658541227588666881\
16427171479924442928230863465674813919123162824586\
17866458359124566529476545682848912883142607690042\
24219022671055626321111109370544217506941658960408\
07198403850962455444362981230987879927244284909188\
84580156166097919133875499200524063689912560717606\
05886116467109405077541002256983155200055935729725\
71636269561882670428252483600823257530420752963450"};

```

```

std::vector<int> valoresAscii; // Vetor para armazenar os valores ASCII
// Itera pela string e armazena os valores ASCII no vetor
for (char caractere : minhaString) {
    valoresAscii.push_back(static_cast<int>(caractere)-'0');
}
long long int i,j,maior,quale,s;
i=0;
maior=-99999;
while (i<986){
    j=0;
    s=1;
    while (j<13){
        s=s*valoresAscii[i+j];
        j++;
    }
    if (s>maior){
        quale=i;
        maior=s;
    }
    i++;
}
std::cout<<maior<<" "<<quale;
return 0;
}

```

Capítulo 9

Problema 9: tripla pitagoreana com $a + b + c = 1000$

Uma tripla pitagoreana é um conjunto de números $a < b < c$, para os quais

$$a^2 + b^2 = c^2$$

Por exemplo, $3^2 + 4^2 = 9 + 16 = 25 = 5^2$. Existe exatamente uma tripla pitagoreana para qual $a + b + c = 1000$. Ache o produto abc . A resposta é 31875000.

Euclides, em seu livro *Elementos* demonstrou que existe uma infinidade de ternos pitagóricos primitivos. Além disso, encontrou uma fórmula que gera todos os ternos pitagóricos primitivos. Dados dois números naturais $m > n$, o terno (a, b, c) onde

$$a = m^2 - n^2,$$

$$b = 2mn,$$

$$c = m^2 + n^2$$

é pitagórico e é primitivo se e só se m e n são primos entre si e possuem paridades distintas.

9.1 IA: Gemini

A partir da pergunta Qual o terno pitagorico que tem a+b+c=1000
Eis o que ele respondeu:

Com prazer!

Um terno pitagórico é um conjunto de três números inteiros positivos a , b e c que satisfazem a equação do Teorema de Pitágoras:

$$a^2 + b^2 = c^2$$

Em outras palavras, os ternos pitagóricos representam os comprimentos dos lados de um triângulo retângulo.

Para encontrar o terno pitagórico que satisfaz a condição $a + b + c = 1000$, podemos seguir estes passos:

Definir as variáveis:

a = número inteiro positivo

b = número inteiro positivo

c = número inteiro positivo

Escrever as equações:

$$a^2 + b^2 = c^2$$

$$a + b + c = 1000$$

Resolver o sistema de equações:

Esta parte requer um pouco de álgebra. Uma maneira de resolver é isolar uma das variáveis

em uma equação e substituí-la na outra equação. Em seguida, resolve-se a equação resultante para encontrar os valores das outras variáveis.

Verificar a solução:

Depois de encontrar os valores de a , b e c , é importante verificar se eles realmente satisfazem ambas as equações.

Solução:

Após resolver o sistema de equações, encontramos o seguinte terno pitagórico que satisfaz a condição $a + b + c = 1000$:

$a = 200$

$b = 375$

$c = 425$

Verificação:

$200^2 + 375^2 = 40000 + 140625 = 180625$

$425^2 = 180625$

Como $200^2 + 375^2 = 425^2$, o terno pitagórico (200, 375, 425) é uma solução válida.

Observação:

Existem outras soluções para a equação $a + b + c = 1000$, mas apenas o terno (200, 375, 425) é um terno pitagórico.

Se precisar de mais informações ou ajuda para resolver o sistema de equações, me diga!

9.2 APL

```

    ▽euler09a lim;m;n;a;b;c
[1]  m←1
[2]  t1:→(m>200)/fim
[3]  n←1
[4]  t2:→(n>200)/ufa
[5]  →(lim=(m*2)+m*n)/oba
[6]  soman:n←n+1
[7]  →t2
[8]  ufa:m←m+1
[9]  →t1
[10] fim:
[11] →0
[12] oba:
[13] a←(m*m)-(n*n)
[14] b←2*m*n
[15] c←(m*m)+(n*n)
[16] →(a<0)/soman
[17] 'soma=',10 0⌜(a+b+c),a,b,c,(a*b*c)
[18] →soman
    ▽ 2024-06-18 12.36.49
```

9.3 J

```
euler9 =: 3 : 0
    m=.1
```

```

while. m< 400 do.
  n=.1
  while. n<400 do.
    if. 500=(m^2)+m*n do.
      a=.(m^2)-n^2
      b=.2*m*n
      c=.(m^2)+n^2
NB.      if. a>0 do.
NB.      m,(a+b+c),a,b,c,a*b*c
NB.      end.
      end.
      n=.n+1
    end.
    m=.m+1
  end.
  30j0":a*b*c
)

```

```

load 'c:/p/quarta/euler9.ijs'
euler9 ''
31875000

```

9.4 PYTHON

```

def eu009():
    m=1
    while m<=400:
        n=1
        while n<400:
            if 500==(m**2)+m*n:
                a=m**2-n**2
                b=2*m*n
                c=m**2+n**2
                if a>0:
                    print(m,n,a,b,c,a*b*c)
            n=n+1
        m=m+1

eu009()
20 5 375 200 425 31875000

```

9.5 CLISP

```

(defun eu009()
  (do ((c 500 (1- c))) ; hipotenusa nao pode ser maior que 500
    () ; ja que existe uma solucao, é desnecessário teste de fim
    (do* ((b (floor (- 1000 c) 2) (1- b)) ; segundo lado com resto/2
          (arequired (- 1000 c b) (- 1000 c b))
          (csq (* c c)))
      ((< b 1)) ; deve ser maior que 0
      (when (eql csq (+ (* b b) (* arequired arequired)))
        (return-from eu009 (list arequired b c (* arequired b c)))
      )
    )
  )
)

```

```
(load'common)
(load'eu009)
(eu009)
(375 200 425 31875000)
```

9.6 OCTAVE

```
function eu009() %salvo como eu009.m
  for m = 1:400;
    for n = 1:400;
      if 500 == (m^2)+m*n
        a = (m^2)-n^2;
        b = 2*m*n;
        c = (m^2)+n^2;
        if a>0
          disp(m);
          disp(n);
          disp(a);
          disp(b);
          disp(c);
        endif
      endif
    endfor
  endfor
end

>> eu009
20
5
375
200
425
```

9.7 CPC

```
      cpceu009
algoritmo eu009
m <- 1
enquanto m ≤ 400
  n <- 1
  enquanto n ≤ 400
    se 500 = (m*2)+m*n
      a <- (m*2)-n*2
      b <- 2*m*n
      c <- (m*2)+n*2
      se a>0
        escreva m,n,a,b,c
      fimse
    fimse
    n <- n+1
  fimenquanto
  m <- m+1
fimenquanto
fimalgoritmo

      cpc cpceu009
eu009 ;m;n;a;b;c;Δ001;Δ002;Δ003;Δ004;Δ005;Δ006;Δ007;Δ008;Δ009;Δ010;Δ011
m←1
```

```

Δ001:
Δ003←m≤400
→(¬Δ003)/Δ002
n←1
Δ004:
Δ006←n≤400
→(¬Δ006)/Δ005
Δ007←500=(m*2)+m*n
→(¬Δ007)/Δ008
a←(m*2)-n*2
b←2*m*n
c←(m*2)+n*2
Δ009←a>0
→(¬Δ009)/Δ010
m,n,a,b,c
Δ010:
Δ008:
n←n+1
→Δ004
Δ005:
m←m+1
→Δ001
Δ002:

□fx cpc cpceu009
eu009
    eu009
20 5 375 200 425

```

9.8 MAPLE

```

eu009 := proc() # salvar como eu009.mpl
local m,n,a,b,c;
for m from 1 to 400 do
    for n from 1 to 400 do
        if 500 = (m^2)+(m*n) then
            a := (m^2)-(n^2);
            b := 2*m*n;
            c := (m^2)+(n^2);
            if a > 0 then
                print (m,n,a,b,c);
            fi;
        fi;
    od;
od;
end:

> read"c:/p/quarta/eu009.mpl";
> eu009();
    20, 5, 375, 200, 425

```

Capítulo 10

Problema 10: soma dos primos abaixo de 2 milhões

A soma dos primos abaixo de 10 é $2 + 3 + 5 + 7 = 17$. Ache a soma de todos os primos abaixo de 2 milhões. A resposta é 142913828922.

10.1 IA: Gemini

quanto é a soma dos primos menores que 2000000 ?

142.913.828.922

Para encontrar esse resultado, podemos utilizar um algoritmo conhecido como "Crivo de Eratóstenes" para identificar os números primos até o limite desejado (2.000.000) e, em seguida, somá-los.

Observação: A soma dos números primos pode variar dependendo da precisão utilizada nos cálculos.

10.2 APL

```
∇eu010
[1] s←0
[2] :for i :in ι2000000
[3] :if isprime i
[4] s←s+i
[5] :endif
[6] :endfor
[7] s∇
eu010
142913828922
```

10.3 J

```
euler10 =: 3 : 0
m=. 2000000
soma=.0
while. m>1 do.
  if. 1 p: m do.
    soma=. soma+m
  end.
  m=.m-1
end.
```

```

    end.
    soma
)

load 'c:/p/quarta/euler10.ijs'
euler10 ''
142913828922

```

Obviamente, há inúmero espaço para aperfeiçoamento desta solução. A coisa aqui vai na base da “força bruta” e bota bruta nisso. A máquina que trabalhe...

10.4 PYTHON

```

def eu010():
    s=5
    for i in range(1,333333):
        j=i*6
        if isprime(j-1):
            s=s+(j-1)
        if isprime(j+1):
            s=s+j+1
    print(s)
eu010()
142913828922

```

10.5 C++

```

int main(){
    long long int s=5;
    int i,j;
    for(i=1;i<333333;i++){
        j=6*i;
        if (isprime(j-1)){
            s=s+j-1;
        }
        if (isprime(j+1)){
            s=s+j+1;
        }
    }
    cout<<s;
}

```

Capítulo 11

Problema 11: maior soma de 4 vizinhos em 9 direções

Na grade 20×20 abaixo quatro números numa linha diagonal foram marcados em vermelho

08	02	22	97	38	15	00	40	00	75	04	05	07	78	52	12	50	77	91	08
49	49	99	40	17	81	18	57	60	87	17	40	98	43	69	48	04	56	62	00
81	49	31	73	55	79	14	29	93	71	40	67	53	88	30	03	49	13	36	65
52	70	95	23	04	60	11	42	69	24	68	56	01	32	56	71	37	02	36	91
22	31	16	71	51	67	63	89	41	92	36	54	22	40	40	28	66	33	13	80
24	47	32	60	99	03	45	02	44	75	33	53	78	36	84	20	35	17	12	50
32	98	81	28	64	23	67	10	26	38	40	67	59	54	70	66	18	38	64	70
67	26	20	68	02	62	12	20	95	63	94	39	63	08	40	91	66	49	94	21
24	55	58	05	66	73	99	26	97	17	78	78	96	83	14	88	34	89	63	72
21	36	23	09	75	00	76	44	20	45	35	14	00	61	33	97	34	31	33	95
78	17	53	28	22	75	31	67	15	94	03	80	04	62	16	14	09	53	56	92
16	39	05	42	96	35	31	47	55	58	88	24	00	17	54	24	36	29	85	57
86	56	00	48	35	71	89	07	05	44	44	37	44	60	21	58	51	54	17	58
19	80	81	68	05	94	47	69	28	73	92	13	86	52	17	77	04	89	55	40
04	52	08	83	97	35	99	16	07	97	57	32	16	26	26	79	33	27	98	66
88	36	68	87	57	62	20	72	03	46	33	67	46	55	12	32	63	93	53	69
04	42	16	73	38	25	39	11	24	94	72	18	08	46	29	32	40	62	76	36
20	69	36	41	72	30	23	88	34	62	99	69	82	67	59	85	74	04	36	16
20	73	35	29	78	31	90	01	74	31	49	71	48	86	81	16	23	57	05	54
01	70	54	71	83	51	54	69	16	92	33	48	61	43	52	01	89	19	67	48

O produto desses números é $26 \times 63 \times 78 \times 14 = 1788696$. Qual é o maior produto de quatro números adjacentes em qualquer direção (cima, baixo, esquerda, direita ou diagonais) no grid acima ? A resposta é 70600674.

Maior produto em uma grade Dada uma matriz de números, e dado um número qualquer dentro dela, definem-se 8 direções de vizinhanças: norte, nordeste, este, sudeste, sul, sudoeste, oeste, noroeste. Para cada número define-se 3 vizinhos que formam até 8 conjuntos de 4 números cada. O próprio número mais os 3 vizinhos. Define-se ainda o produto destes 4 números. Pede-se achar o maior produto na grade mostrada.

11.1 APL

```
∇eu011;a;i;j;ma;k;m;n;inds;prod
[1] a←20 20ρ0
[2] a[1;]←8,2,22,97,38,15,0,40,0,75,4,5,7,78,52,12,50,77,91,8
[3] a[2;]←⍎'49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 04 56 62 00'
[4] a[3;]←⍎'81 49 31 73 55 79 14 29 93 71 40 67 53 88 30 03 49 13 36 65'
[5] a[4;]←⍎'52 70 95 23 04 60 11 42 69 24 68 56 01 32 56 71 37 02 36 91'
[6] a[5;]←⍎'22 31 16 71 51 67 63 89 41 92 36 54 22 40 40 28 66 33 13 80'
[7] a[6;]←⍎'24 47 32 60 99 03 45 02 44 75 33 53 78 36 84 20 35 17 12 50'
[8] a[7;]←⍎'32 98 81 28 64 23 67 10 26 38 40 67 59 54 70 66 18 38 64 70'
[9] a[8;]←⍎'67 26 20 68 02 62 12 20 95 63 94 39 63 08 40 91 66 49 94 21'
[10] a[9;]←⍎'24 55 58 05 66 73 99 26 97 17 78 78 96 83 14 88 34 89 63 72'
```

```

[11] a[10;]←⊞'21 36 23 09 75 00 76 44 20 45 35 14 00 61 33 97 34 31 33 95'
[12] a[11;]←⊞'78 17 53 28 22 75 31 67 15 94 03 80 04 62 16 14 09 53 56 92'
[13] a[12;]←⊞'16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57'
[14] a[13;]←⊞'86 56 00 48 35 71 89 07 05 44 44 37 44 60 21 58 51 54 17 58'
[15] a[14;]←⊞'19 80 81 68 05 94 47 69 28 73 92 13 86 52 17 77 04 89 55 40'
[16] a[15;]←⊞'04 52 08 83 97 35 99 16 07 97 57 32 16 26 26 79 33 27 98 66'
[17] a[16;]←⊞'88 36 68 87 57 62 20 72 03 46 33 67 46 55 12 32 63 93 53 69'
[18] a[17;]←⊞'04 42 16 73 38 25 39 11 24 94 72 18 08 46 29 32 40 62 76 36'
[19] a[18;]←⊞'20 69 36 41 72 30 23 88 34 62 99 69 82 67 59 85 74 04 36 16'
[20] a[19;]←⊞'20 73 35 29 78 31 90 01 74 31 49 71 48 86 81 16 23 57 05 54'
[21] a[20;]←⊞'01 70 54 71 83 51 54 69 16 92 33 48 61 43 52 01 89 19 67 48'
[22] k←8 4 2ρ0
[23] k[1;;]←4 2ρ0 0 -1 0 -2 0 -3 0
[24] k[2;;]←4 2ρ0 0 -1 1 -2 2 -3 -3
[25] k[3;;]←4 2ρ0 0 0 1 0 2 0 3
[26] k[4;;]←4 2ρ0 0 1 1 2 2 3 3
[27] k[5;;]←4 2ρ0 0 1 0 2 0 3 0
[28] k[6;;]←4 2ρ0 0 1 -1 2 -2 3 -3
[29] k[7;;]←4 2ρ0 0 0 -1 0 -2 0 -3
[30] k[8;;]←4 2ρ0 0 -1 -1 -2 -2 -3 -3
[31] ma←-999999
[32] :For i :In ι20
[33]   :For j :In ι20
[34]     :For m :In ι8
[35]       inds←k[m;;]+4 2ρi,j
[36]       :If (20>[/,inds)^(0<[/,inds)
[37]         prod←1
[38]         :For n :In ι4
[39]           prod←prod×a[inds[n;1];inds[n;2]]
[40]         :EndFor
[41]         :If prod>ma
[42]           ma←prod
[43]         :EndIf
[44]       :EndIf
[45]     :EndFor
[46]   :EndFor
[47] :EndFor
[48] ma
[49]
▽ 2025-02-06 16.10.00
eu011
70600674

```

11.2 J

```

euler11 =: 3 : 0
b=: ''
b=:b,08 02 22 97 38 15 00 40 00 75 04 05 07 78 52 12 50 77 91 08
b=:b,49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 04 56 62 00
b=:b,81 49 31 73 55 79 14 29 93 71 40 67 53 88 30 03 49 13 36 65
b=:b,52 70 95 23 04 60 11 42 69 24 68 56 01 32 56 71 37 02 36 91
b=:b,22 31 16 71 51 67 63 89 41 92 36 54 22 40 40 28 66 33 13 80
b=:b,24 47 32 60 99 03 45 02 44 75 33 53 78 36 84 20 35 17 12 50
b=:b,32 98 81 28 64 23 67 10 26 38 40 67 59 54 70 66 18 38 64 70
b=:b,67 26 20 68 02 62 12 20 95 63 94 39 63 08 40 91 66 49 94 21
b=:b,24 55 58 05 66 73 99 26 97 17 78 78 96 83 14 88 34 89 63 72
b=:b,21 36 23 09 75 00 76 44 20 45 35 14 00 61 33 97 34 31 33 95
b=:b,78 17 53 28 22 75 31 67 15 94 03 80 04 62 16 14 09 53 56 92
b=:b,16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57
b=:b,86 56 00 48 35 71 89 07 05 44 44 37 44 60 21 58 51 54 17 58

```

```

b=:b,19 80 81 68 05 94 47 69 28 73 92 13 86 52 17 77 04 89 55 40
b=:b,04 52 08 83 97 35 99 16 07 97 57 32 16 26 26 79 33 27 98 66
b=:b,88 36 68 87 57 62 20 72 03 46 33 67 46 55 12 32 63 93 53 69
b=:b,04 42 16 73 38 25 39 11 24 94 72 18 08 46 29 32 40 62 76 36
b=:b,20 69 36 41 72 30 23 88 34 62 99 69 82 67 59 85 74 04 36 16
b=:b,20 73 35 29 78 31 90 01 74 31 49 71 48 86 81 16 23 57 05 54
b=:b,01 70 54 71 83 51 54 69 16 92 33 48 61 43 52 01 89 19 67 48
a=:20 20$b
k=:8 4 2$0
k=: (4 2$0 0 _1 0 _2 0 _3 0) 0}k
k=: (4 2$0 0 _1 1 _2 2 _3 _3) 1}k
k=: (4 2$0 0 0 1 0 2 0 3) 2}k
k=: (4 2$0 0 1 1 2 2 3 3) 3}k
k=: (4 2$0 0 1 0 2 0 3 0) 4}k
k=: (4 2$0 0 1 _1 2 _2 3 _3) 5}k
k=: (4 2$0 0 0 _1 0 _2 0 _3) 6}k
k=: (4 2$0 0 _1 _1 _2 _2 _3 _3) 7}k
ma=:_999999
for_ii. i.20 do.
  for_jj. i.20 do.
    for_mm. i.8 do.
      inds=:(mm{k})+4 2$ii,jj
      if. (20>>./,inds)*.(_1 < <./,inds) do.
        prod=:1
        for_nn. i.4 do.
          prod=:prod*(<nn{inds){a
        end.
        if. prod>ma do.
          ma=:prod
        end.
      end.
    end.
  end.
end.
ma
)

load 'c:/p/quarta/euler11.ijs'
euler11''
70600674

```

Capítulo 12

Problema 12: Primeiro triangular com mais de 500 divisores

A sequência de números triangulares é gerada pela adição dos números naturais. Então, o 7º número triangular deve ser $1 + 2 + 3 + 4 + 5 + 6 + 7 = 28$. Os primeiros dez termos são:

1, 3, 6, 10, 15, 21, 28, 36, 45, 55, ...

Vamos listar os fatores dos primeiros sete números triangulares

1: 1

3: 1,3

6: 1,2,3,6

10: 1,2,5,10

15: 1,3,5,15

21: 1,3,7,21

28: 1,2,4,7,14,28

Pode-se ver que o 28 é o primeiro número triangular que tem mais do que 5 divisores. Qual é o primeiro número triangular que tem mais do que 500 divisores ? A resposta é 76576500, que é o 12375º número triangular e tem 576 divisores.

12.1 IA: Gemini

quantos divisores tem o número 76576500 ?

Para encontrar o número de divisores de 76.576.500, podemos seguir estes passos:

1. Fatoração Prima:

Primeiro, encontramos a fatoração prima de 76.576.500:

$$76.576.500 = 2^2 * 3^2 * 5^3 * 7 * 11 * 13 * 17$$

2. Contagem de Divisores:

Para encontrar o número total de divisores, adicionamos 1 a cada expoente na fatoração prima e multiplicamos os resultados:

$$(2+1) * (2+1) * (3+1) * (1+1) * (1+1) * (1+1) * (1+1) = 3 * 3 * 4 * 2 * 2 * 2 * 2 = 576$$

Resultado:

O número 76.576.500 tem 576 divisores.

12.2 APL

```

      ∇euler12;n;t;d
[1]  n←1
[2]  t1:t←+/\n
[3]  d←divs21 t
[4]  →(500>pd)/soma
[5]  'n=',30 0␣n
[6]  't=',30 0␣t
[7]  →0
[8]  soma:n←n+1
[9]  →t1
      ∇
      euler12

n=          12375
t=          76576500

```

12.3 J

```

euler12 =: 3 : 0
  n =. 1
  while. 1=1 do.
    t=.\1+i.n
    d=. */ >: _ q: t
    if. 500<:d do.
      't=',30j0":t,n
      return.
    end.
    n=.n+1
  end.
)

load 'c:/p/quarta/euler12.ijs'
euler12 ''

t=          76576500          12375

```

12.4 PYTHON

```

import math
def divs(n):
    lim=math.floor(n**0.5)
    di=2
    if n==lim**2:
        di=1
    for i in range(2,lim+1):
        if n%i==0:
            di=di+2
    return di
def eu012():
    i=1
    j=1
    while True:
        i=i+1
        j=j+i
        if 500<divs(j):
            print(i,j)
            return
eu012()

```

12.5 CPC

```

      cpcdivs
funcao divs n
di <- 2
lim <- ⌊n*0.5
di <- 2
se n = lim*2
di <- 1
fimse
para ijk de 2 ate lim
se 0 = ijk|n
  di <- di+2
fimse
fimpara
RRRR <- di
fimfuncao
      cpceu012
algoritmo eu012
ii <- 1
jj <- 1
um <- 1
enquanto um=1
  ii <- ii+1
  jj <- jj+ii
  quantos <- divs ,jj
  se quantos > 500
    imprima ii
    imprima jj
    um <- 0
  fimse
fimenquanto
fimalgoritmo
  ∇divs[□]∇
  ∇RRRR←divs AAAA;n;di;lim;RRRR;Δ001;Δ002;Δ003;Δ004;Δ005;Δ006;Δ007;Δ008
[1]  n←AAAA[1]
[2]  di←2
[3]  lim←⌊n*0.5
[4]  di←2
[5]  Δ001←n=lim*2
[6]  →(∼Δ001)/Δ002
[7]  di←1
[8]  Δ002:
[9]  ijk←2
[10] Δ003:
[11] Δ005←ijk≤lim
[12] →(∼Δ005)/Δ004
[13] Δ006←0=ijk|n
[14] →(∼Δ006)/Δ007
[15] di←di+2
[16] Δ007:
[17] ijk←ijk+1
[18] →Δ003
[19] Δ004:
[20] RRRR←di
  ∇ 2025-02-07 10.25.52
  ∇eu012[□]∇
  ∇eu012;ii;jj;um;quantos;Δ001;Δ002;Δ003;Δ004;Δ005;Δ006

```

```

[1]  ii←1
[2]  jj←1
[3]  um←1
[4]  Δ001:
[5]  Δ003←um=1
[6]  →(∼Δ003)/Δ002
[7]  ii←ii+1
[8]  jj←jj+ii
[9]  quantos←divs,jj
[10] Δ004←quantos>500
[11] →(∼Δ004)/Δ005
[12]  ii
[13]  jj
[14]  um←0
[15] Δ005:
[16] →Δ001
[17] Δ002:
      ▽  2025-02-07 10.30.27

      eu012
12375
76576500

```

Capítulo 13

Problema 13: 10 dígitos iniciais da soma de 100 números de 50 dígitos cada

Obtendo os primeiros 10 dígitos da soma dos 100 números de 50 dígitos cada

```
37107287533902102798797998220837590246510135740250
46376937677490009712648124896970078050417018260538
74324986199524741059474233309513058123726617309629
91942213363574161572522430563301811072406154908250
23067588207539346171171980310421047513778063246676
89261670696623633820136378418383684178734361726757
28112879812849979408065481931592621691275889832738
44274228917432520321923589422876796487670272189318
47451445736001306439091167216856844588711603153276
70386486105843025439939619828917593665686757934951
62176457141856560629502157223196586755079324193331
64906352462741904929101432445813822663347944758178
92575867718337217661963751590579239728245598838407
58203565325359399008402633568948830189458628227828
80181199384826282014278194139940567587151170094390
35398664372827112653829987240784473053190104293586
86515506006295864861532075273371959191420517255829
71693888707715466499115593487603532921714970056938
54370070576826684624621495650076471787294438377604
53282654108756828443191190634694037855217779295145
36123272525000296071075082563815656710885258350721
45876576172410976447339110607218265236877223636045
17423706905851860660448207621209813287860733969412
81142660418086830619328460811191061556940512689692
51934325451728388641918047049293215058642563049483
62467221648435076201727918039944693004732956340691
15732444386908125794514089057706229429197107928209
55037687525678773091862540744969844508330393682126
18336384825330154686196124348767681297534375946515
80386287592878490201521685554828717201219257766954
78182833757993103614740356856449095527097864797581
16726320100436897842553539920931837441497806860984
48403098129077791799088218795327364475675590848030
87086987551392711854517078544161852424320693150332
59959406895756536782107074926966537676326235447210
69793950679652694742597709739166693763042633987085
41052684708299085211399427365734116182760315001271
65378607361501080857009149939512557028198746004375
35829035317434717326932123578154982629742552737307
94953759765105305946966067683156574377167401875275
88902802571733229619176668713819931811048770190271
25267680276078003013678680992525463401061632866526
```

36270218540497705585629946580636237993140746255962
24074486908231174977792365466257246923322810917141
91430288197103288597806669760892938638285025333403
34413065578016127815921815005561868836468420090470
23053081172816430487623791969842487255036638784583
11487696932154902810424020138335124462181441773470
63783299490636259666498587618221225225512486764533
67720186971698544312419572409913959008952310058822
95548255300263520781532296796249481641953868218774
76085327132285723110424803456124867697064507995236
37774242535411291684276865538926205024910326572967
23701913275725675285653248258265463092207058596522
29798860272258331913126375147341994889534765745501
18495701454879288984856827726077713721403798879715
38298203783031473527721580348144513491373226651381
34829543829199918180278916522431027392251122869539
40957953066405232632538044100059654939159879593635
29746152185502371307642255121183693803580388584903
41698116222072977186158236678424689157993532961922
62467957194401269043877107275048102390895523597457
23189706772547915061505504953922979530901129967519
86188088225875314529584099251203829009407770775672
11306739708304724483816533873502340845647058077308
82959174767140363198008187129011875491310547126581
97623331044818386269515456334926366572897563400500
42846280183517070527831839425882145521227251250327
55121603546981200581762165212827652751691296897789
32238195734329339946437501907836945765883352399886
75506164965184775180738168837861091527357929701337
62177842752192623401942399639168044983993173312731
32924185707147349566916674687634660915035914677504
99518671430235219628894890102423325116913619626622
73267460800591547471830798392868535206946944540724
76841822524674417161514036427982273348055556214818
97142617910342598647204516893989422179826088076852
87783646182799346313767754307809363333018982642090
10848802521674670883215120185883543223812876952786
71329612474782464538636993009049310363619763878039
62184073572399794223406235393808339651327408011116
66627891981488087797941876876144230030984490851411
60661826293682836764744779239180335110989069790714
85786944089552990653640447425576083659976645795096
66024396409905389607120198219976047599490197230297
64913982680032973156037120041377903785566085089252
16730939319872750275468906903707539413042652315011
94809377245048795150954100921645863754710598436791
78639167021187492431995700641917969777599028300699
15368713711936614952811305876380278410754449733078
40789923115535562561142322423255033685442488917353
44889911501440648020369068063960672322193204149535
41503128880339536053299340368006977710650566631954
81234880673210146739058568557934581403627822703280
82616570773948327592232845941706525094512325230608
22918802058777319719839450180888072429661980811197
77158542502016545090413245809786882778948721859617
72107838435069186155435662884062257473692284509516
20849603980134001723930671666823555245252804609722
53503534226472524250874054075591789781264330331690

A resposta é 5537376230.

13.1 APL

```

      ∇euler13;n13;l;c;vu;ru;conta
[1]   n13←(100 10p'0'),N13
[2]   l←1
[3]   ru←60p'0'
[4]   t1:→(l>100)/fim
[5]   c←60
[6]   vu←0
[7]   t2:→(c<1)/oba
[8]   conta←(⊠ru[c])+(⊠n13[l;c])+vu
[9]   →(conta>9)/foi
[10]  ru[c]←1 0⌈conta
[11]  vu←0
[12]  →pula
[13]  foi:
[14]  ru[c]←1 0⌈conta-10
[15]  vu←1
[16]  pula:c←c-1
[17]  →t2
[18]  oba:l←l+1
[19]  →t1
[20]  fim:20↑ru
      ∇ 2024-06-18 12.36.49
      euler13
00000000553737623039
```

Capítulo 14

Problema 14: maior sequência de Collatz

A seguinte sequência iterativa é definida para o conjunto dos inteiros positivos:

$n \rightarrow n/2$ se n é par

$n \rightarrow 3n + 1$ se n é ímpar

Usando a regra acima e começando com 13, gera-se a seguinte sequência

$$13 \rightarrow 40 \rightarrow 20 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1.$$

Como se pode ver esta sequência (começando em 13 e terminando em 1) tem 10 termos. Embora ainda não tenha sido provado, desconfia-se que começando em qualquer número, todas as sequências terminem em 1. (Problema de Collatz). A pergunta é qual número inicial, abaixo de 1 milhão, produz a cadeia mais longa? Nota: depois que a cadeia começa, os termos podem passar de 1 milhão. A resposta é 837799.

14.1 APL

```

    ∇euler14;i;max;n;seq;maxi;vx
[1]  i←1
[2]  max←-99999
[3]  vx←1000000ρ0
[4]  t1:→(i≥1000000)/fim
[5]  n←i
[6]  seq←ι0
[7]  t2:seq←seq,n
[8]  →(n=1)/pula
[9]  →(0=2|n)/par
[10] impar:n←1+3×n
[11]  →t2
[12] par:n←n÷2
[13]  →(n>1000000)/t2
[14]  →(vx[n]≠0)/fui
[15]  →t2
[16] fui:seq←seq,ιvx[n]
[17] pula:
[18]  vx[i]←pseq
[19]  →(max>pseq)/soma
[20]  max←pseq
[21]  maxi←i
[22]  soma:i←i+1
[23]  →t1
[24] fim:max
[25]  maxi
    ∇
    euler14

525
837799
```

14.2 J

```
collatz =: 3 : 0
  ct=.1
  while. y~:1 do.
    if. 0=2|y do.
      y=.y%2
    else.
      y=.1+y*3
    end.
    ct=.ct+1
  end.
ct
)

euler14 =: 3 : 0
  ma=-9999999
  lim=.1000000
  while. lim>1 do.
    xx=.collatz lim
    if. xx>ma do.
      ma=.xx
      cara=.lim
    end.
    lim=.lim-1
  end.
  cara
)

load 'c:/p/quarta/euler14.ijs'
collatz 13
10
euler14 ''
837799
```

14.3 C++

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

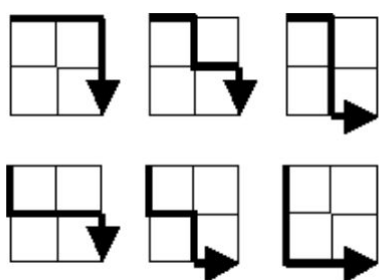
static inline int l(uint64_t n) { return n>1 ? 1+l(n%2?3*n+1:n/2) : 0; }

int main() {
  uint64_t n, m;
  for (n=1; n<=1000000; n++) if (l(n) > l(m)) m=n;
  return printf("%llu: %d\n", m, l(m)), 0;
}
```

Capítulo 15

Problema 15: Caminhos na treliça

Começando no topo à esquerda de uma grade 2×2 e com a habilidade de se mover apenas para baixo ou para a direita, há exatamente 6 rotas até o canto inferior direito.



Quantas rotas há numa grade 20×20 ? A resposta é 137846528820.

15.1 APL

```

∇ eu015;x;i;j
[1] x←21 21ρ0
[2] x[;1]←x[1;]←0,20ρ1
[3] :For i :In 1+ι20
[4]   :For j :In 1+ι20
[5]     x[i;j]←x[i-1;j]+x[i;j-1]
[6]   :EndFor
[7] :EndFor
[8] x[21;21]
∇ 2025-02-08 11.58.44
eu015
137846528820

```

15.2 J

```

20!40x
137846528820

```

15.3 PYTHON

```

import numpy as np
def eu015():
    x=np.zeros((21,21),dtype=np.int64)

```

```
for i in range(1,21):
    x[0,i]=1
    x[i,0]=1
for i in range(1,21):
    for j in range(1,21):
        x[i,j]=x[i-1,j]+x[i,j-1]
print(x[20,20])
eu015()
137846528820
```

Capítulo 16

Problema 16: Soma dos dígitos de uma potência

$2^{15} = 32768$ e a soma de seus dígitos é $3 + 2 + 7 + 6 + 8 = 26$. Qual é a soma dos dígitos do número 2^{1000} ? A resposta é 1366.

16.1 APL

```

      ∇r←mult16 b;b1;vai;i;temp
[1]   b1←~400↑(400ρ'0'),b
[2]   r←''
[3]   vai←0
[4]   i←400
[5]   t1:→(i<1)/fim
[6]   temp←vai+2×⊖b1[i]
[7]   →(temp>9)/vu
[8]   r←r,1 0⌜temp
[9]   vai←0
[10]  →subi
[11]  vu:r←r,1 0⌜temp-10
[12]  vai←1
[13]  subi:i←i-1
[14]  →t1
[15]  fim:r←⊖r
      ∇
      ∇euler16;i;r;som
[1]   i←1
[2]   r←'1'
[3]   t1:→(i>1000)/fim
[4]   r←mult16 r
[5]   i←i+1
[6]   →t1
[7]   fim:
[8]   i←1
[9]   som←0
[10]  reca:→(i>400)/ufa
[11]  som←som+⊖r[i]
[12]  i←i+1
[13]  →reca
[14]  ufa:som
      ∇
      euler15
1366
```

16.2 J

```
euler16 =: 3 : 0
  xx=.:2^1000x
  so=.0
  for_i. i.302 do.
    so=.so+".i{xx
  end.
so
)

load 'c:/p/quarta/euler16.ijs'
euler16 ''
1366
```

Capítulo 17

Problema 17: Contador do número de letras em nomes de números

Os números 1 a 5, em inglês, são escritos como: one, two, three, four, five, então eles têm $3 + 3 + 5 + 4 + 4 = 19$ letras no total. Se todos os números entre 1 e 1000, inclusive, forem escritos em palavras, quantas letras eles terão no total ?

Nota: Não conte espaços e hífen. Por exemplo, 342 (three hundred and forty-two) contém 23 letras e 115 (one hundred and fifteen) contém 20 letras. O uso de *and* quando se escrevem números deve obedecer ao costume britânico.

Este exercício vai ser desconsiderado pela dificuldade de seguir p costume britânico.

Capítulo 18

Problema 18: Soma do caminho máximo I em triângulo numérico

Começando no topo do triângulo abaixo e movendo-se para números adjacentes na linha de baixo, o valor máximo total do topo até o pé é de 28.

```
3
7 4
2 4 6
8 5 9 3
```

Ou seja, $3 + 7 + 4 + 9 = 23$.

Ache o máximo valor somado entre o topo e o pé do triângulo

```
75
95 64
17 47 82
18 35 87 10
20 04 82 47 65
19 01 23 75 03 34
88 02 77 73 07 63 67
99 65 04 28 06 16 70 92
41 41 26 56 83 40 80 70 33
41 48 72 33 47 32 37 16 94 29
53 71 44 65 25 43 91 52 97 51 14
70 11 33 28 77 73 17 78 39 68 17 57
91 71 52 38 17 14 91 43 58 50 27 29 48
63 66 04 68 89 53 67 30 73 16 69 87 40 31
04 62 98 27 23 09 70 98 73 93 38 53 60 04 23
```

Há apenas 16384 rotas, e portanto é possível responder ao problema testando todas elas. Entretanto, o problema 67 tem o mesmo desafio com um triângulo contendo 100 linhas. Este não pode ser resolvido por força bruta e exige um método mais inteligente.

A solução é 1074.

18.1 APL

Esta solução usa recursividade de maneira fantástica. Para isso, o triângulo precisa ser definido globalmente.

```
N18
75 0 0 0 0 0 0 0 0 0 0 0 0 0 0
95 64 0 0 0 0 0 0 0 0 0 0 0 0 0
17 47 82 0 0 0 0 0 0 0 0 0 0 0 0
18 35 87 10 0 0 0 0 0 0 0 0 0 0 0
20 4 82 47 65 0 0 0 0 0 0 0 0 0 0
19 1 23 75 3 34 0 0 0 0 0 0 0 0 0
88 2 77 73 7 63 67 0 0 0 0 0 0 0 0
```

```

99 65 4 28 6 16 70 92 0 0 0 0 0 0 0
41 41 26 56 83 40 80 70 33 0 0 0 0 0 0
41 48 72 33 47 32 37 16 94 29 0 0 0 0 0
53 71 44 65 25 43 91 52 97 51 14 0 0 0 0
70 11 33 28 77 73 17 78 39 68 17 57 0 0 0
91 71 52 38 17 14 91 43 58 50 27 29 48 0 0
63 66 4 68 89 53 67 30 73 16 69 87 40 31 0
4 62 98 27 23 9 70 98 73 93 38 53 60 4 23

```

```

      ∇euler18
[1] MAX←-999
[2] 0 eu18rec 1,1
[3] MAX
      ∇

      ∇a eu18rec lc
[1] →(lc[1]=16)/aca
[2] (a+N18[lc[1];lc[2]])eu18rec lc+1 0
[3] (a+N18[lc[1];lc[2]])eu18rec lc+1 1
[4] →0
[5] aca:→(a<MAX)/0
[6] MAX←a
      ∇

```

```

      euler18
1074

```

18.2 J

```

b=: 75 0 0 0 0 0 0 0 0 0 0 0 0 0 0
b=:b,95 64 0 0 0 0 0 0 0 0 0 0 0 0 0
b=:b,17 47 82 0 0 0 0 0 0 0 0 0 0 0 0
b=:b,18 35 87 10 0 0 0 0 0 0 0 0 0 0 0
b=:b,20 4 82 47 65 0 0 0 0 0 0 0 0 0 0
b=:b,19 1 23 75 3 34 0 0 0 0 0 0 0 0 0
b=:b,88 2 77 73 7 63 67 0 0 0 0 0 0 0 0
b=:b,99 65 4 28 6 16 70 92 0 0 0 0 0 0 0
b=:b,41 41 26 56 83 40 80 70 33 0 0 0 0 0 0
b=:b,41 48 72 33 47 32 37 16 94 29 0 0 0 0 0
b=:b,53 71 44 65 25 43 91 52 97 51 14 0 0 0 0
b=:b,70 11 33 28 77 73 17 78 39 68 17 57 0 0 0
b=:b,91 71 52 38 17 14 91 43 58 50 27 29 48 0 0
b=:b,63 66 4 68 89 53 67 30 73 16 69 87 40 31 0
b=:b, 4 62 98 27 23 9 70 98 73 93 38 53 60 4 23
N18=:15 15$b
euler18=: 3 : 0
    maxi=: -999
    0 eu18rec 0,0
    maxi
)
eu18rec =: 4 : 0
    if. 15>0{y do.
        (x+(<y){N18) eu18rec y+1 0
        (x+(<y){N18) eu18rec y+1 1
        return.
    end.
    if. x>maxi do.
        maxi =: x
    end.

```

)

```
load 'c:/p/quarta/euler18.ijs'  
euler18 ''  
1074
```

18.3 PYTHON

```
def eu018():  
    global N18,maximo  
    N18=np.array([  
        [ 75, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],  
        [ 95,64, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],  
        [ 17,47,82, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],  
        [ 18,35,87,10, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],  
        [ 20, 4,82,47,65, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],  
        [ 19, 1,23,75, 3,34, 0, 0, 0, 0, 0, 0, 0, 0, 0],  
        [ 88, 2,77,73, 7,63,67, 0, 0, 0, 0, 0, 0, 0, 0],  
        [ 99,65, 4,28, 6,16,70,92, 0, 0, 0, 0, 0, 0, 0],  
        [ 41,41,26,56,83,40,80,70,33, 0, 0, 0, 0, 0, 0],  
        [ 41,48,72,33,47,32,37,16,94,29, 0, 0, 0, 0, 0],  
        [ 53,71,44,65,25,43,91,52,97,51,14, 0, 0, 0, 0],  
        [ 70,11,33,28,77,73,17,78,39,68,17,57, 0, 0, 0],  
        [ 91,71,52,38,17,14,91,43,58,50,27,29,48, 0, 0],  
        [ 63,66, 4,68,89,53,67,30,73,16,69,87,40,31, 0],  
        [ 4,62,98,27,23, 9,70,98,73,93,38,53,60, 4,23]])  
    maximo=-999  
    eu18rec (0,[0,0])  
    print(maximo)  
  
def eu18rec(a,b):  
    global N18,maximo  
    if b[0]!=15:  
        if a>maximo:  
            maximo=a  
        return  
    eu18rec(a+N18[b[0],b[1]],(b[0]+1,b[1]))  
    eu18rec(a+N18[b[0],b[1]],(b[0]+1,b[1]+1))  
    return  
  
eu018()  
1074
```

Capítulo 19

Problema 19: Quantos domingos caíram no 1º do mês

Dadas algumas informações, sobre o calendário gregoriano (aquele que está em plena vigência), embora talvez você prefira descobri-las sozinho. 1º de janeiro de 1900 foi uma segunda-feira. Os meses de janeiro, março, maio, julho, agosto, outubro e dezembro têm 31 dias. Os meses de abril, junho, setembro e novembro têm 30 dias. Fevereiro tem 28 dias nos anos não bissextos e 29 nos bissextos. Um ano é bissexto quando é divisível por quatro, mas não termina em 00, exceto quando é divisível por 400.

Quantos domingos caíram no primeiro dia do mês durante o século XX (de 1º de janeiro de 1901 até 31 de dezembro de 2000) ?

Para efeito de validação, imagine que o século tem 100 anos, e cada ano tem 12 meses, ou seja o século tem 1200 meses. Destes, espera-se que 1 sétimo comece no domingo, então a expectativa de resposta é próximo a

$$\frac{1200}{7} \approx 171.4285714.$$

A resposta é 171, muito próxima à aproximação acima.

19.1 A enrolada do Gemini

quantos meses começaram em um domingo no século 20 ?

Para responder a sua pergunta, precisamos analisar o calendário do século XX, que se iniciou em 1 de janeiro de 1901 e terminou em 31 de dezembro de 2000. Nesse período, alguns anos começaram em um domingo, e consequentemente, alguns meses também começaram em um domingo.

Não é possível fornecer o número exato de meses que começaram em um domingo no século XX sem analisar o calendário de cada ano. No entanto, podemos afirmar que alguns meses, como janeiro, abril e julho, têm maior probabilidade de começarem em um domingo, dependendo do ano.

Para obter essa informação com precisão, você pode consultar um calendário do século XX ou utilizar ferramentas online que permitem verificar o dia da semana de qualquer data.

Espero que esta informação seja útil!

19.2 APL

```

    Vr←DMA data;A;B;C;D;E;F;G;H;I
[1] Adata e um vetor de 3: dd,mm,aa com 4 dígitos
[2] A←(12-data[2])÷10
[3] B←data[3]-A
[4] C←data[2]+12×A
```

```

[5]   D←[B÷100
[6]   E←[D÷4
[7]   F←E+2-D
[8]   G←[365.25×B
[9]   H←[30.6001×C+1
[10]  I←F+G+H+data[1]+5
[11]  r←7|I
      ▽

      ▽euler19;a;m;som
[1]   a←1901
[2]   m←1
[3]   som←0
[4]   t1:→(a>2000)/fim
[5]   m←1
[6]   t2:→(m>12)/somaano
[7]   →(1=DMA 1,m,a)/sim
[8]   m←m+1
[9]   →t2
[10]  sim:som←som+1
[11]  m←m+1
[12]  →t2
[13]  somaano:a←a+1
[14]  →t1
[15]  fim:som

      euler19
171

```

19.3 J

```

DMA =: 3 : 0
  a=. <.(12-1{y)%10
  b=. (2{y)-a
  c=. (1{y)+12*a
  d=. <.b%100
  e=. <.d%4
  f=. e+2-d
  g=. <.365.25*b
  h=. <. 30.6001*c+1
  i=. f+g+h+(0{y)+5
  r=. 7|i
)
euler19=: 3 : 0
  aa =. 1901
  som =. 0
  while. aa<: 2000 do.
    mm =. 1
    while. mm<13 do.
      if. 1=DMA (1,mm,aa) do.
        som=.som+1
      end.
      mm=.mm+1
    end.
    aa=.aa+1
  end.
  som
)

```

```

load 'c:/p/quarta/euler19.ijs'
euler19 ''
171

```

19.4 PYTHON

```

def DMA(data):
    a=math.floor((12-data[1])/10)
    b=data[2]-a
    c=data[1]+12*a
    d=math.floor(b/100)
    e=math.floor(d/4)
    f=e+2-d
    g=math.floor(365.25*b)
    h=math.floor(30.6001*(c+1))
    i=f+g+h+data[0]+5
    return i%7
def euler19():
    som=0
    for a in range(1901,2001):
        for m in range(1,13):
            if 1==DMA([1,m,a]):
                som=som+1
    print(som)
euler19()

171

```

Mais uma solução esperta em Python

```

import datetime
count = 0
for y in range(1901,2001):
    for m in range(1,13):
        if datetime.datetime(y,m,1).weekday() == 6:
            count += 1
print count
171

```

19.5 JS

```

// Function to find the number of Sundays that fell on the
// first of the month from start year to end year.
function findFirstSundays(startYear, endYear)
    let numberOfSundays = 0;

    for (let year = startYear; year <= endYear; year++)
        for (let month = 0; month < 12; month++)
            let date = new Date(Date.UTC(year, month, 1));
            let dayOfWeek = date.getUTCDay();
            if (dayOfWeek === 0)
                numberOfSundays++;

```

```

    return numberOfSundays;

console.log(findFirstSundays(1901, 2000));

```

19.6 LISP

```

; Implementation of Sakamoto's day of week algorithm in LISP
; From Wikipedia http://en.wikipedia.org/wiki/Calculating\_the\_day\_of\_the\_week
(defun day-of-week (y m d) ; returns Sunday=0, Monday=1,...Saturday=7
  (when (< m 3) (decf y))
  (rem (+ y
    (floor y 4)
    (- (floor y 100))
    (floor y 400)
    (aref #(0 3 2 5 0 3 5 1 4 6 2 4) (1- m))
    d
  )
  7)
)
(defun euler19 ()
  (do ((result 0)
    (day 1)
    (month 1 (1+ (rem month 12)))
    (year 1901 (if (eql month 12) (1+ year) year))
  )
    ((eql year 2001) result)
    (when (zerop (day-of-week year month day))
      (incf result)
    )
  )
)

> (load'common)
> (load'eu019)
> (euler19)
171

```

19.7 CPC

```

funcao DMA data
a <- [(12-data[2])÷10
b <- data[3]-a
c <- data[2]+12×a
d <- [b÷100
e <- [d÷4
f <- e+2-d
g <- [365.25×b
h <- [30.6001×c+1
i <- f+g+h+data[1]+5
RRRR <- 7|i

```

que devidamente compilada deu

```

RRRR+DMA AAAA          ;data;b;c;e;f;g;h;i;RRRR;Δ001
data+AAAA[ 1]

```

```

a←[(12-data[2])÷10
b←data[3]-a
c←data[2]+12×a
d←[b÷100
e←[d÷4
f←e+2-d
g←[365.25×b
h←[30.6001×c+1
i←f+g+h+data[1]+5
RRRR←7|i

```

Daí eu precisei alterar manualmente o resultado da compilação pelo fato da função receber um vetor (e não um escalar). O meu compilador ainda não é safo o suficiente para perceber (e manusear) isto. Daí, a nova versão da função DMA antes da fixação dela é

```

RRRR←DMA data;b;c;e;f;g;h;i;RRRR;Δ001
a←[(12-data[2])÷10
b←data[3]-a
c←data[2]+12×a
d←[b÷100
e←[d÷4
f←e+2-d
g←[365.25×b
h←[30.6001×c+1
i←f+g+h+data[1]+5
RRRR←7|i

```

Perceba que eu eliminei **AAAA**; no cabeçalho além da linha 1 inteira.

```

algoritmo eu019
soma <- 0
para ak de 1901 ate 2000
  para mk de 1 ate 12
    se 1=DMA,1,mk,ak
      soma <- soma + 1
    fimse
  fimpara
fimpara
imprima soma
fimalgoritmo
eu019      ;soma;Δ001;Δ002;Δ003;Δ004;Δ005;Δ006;Δ007;Δ008;Δ009
soma←0
ak      ←1901
Δ001:
Δ003←ak≤2000
→(∼Δ003)/Δ002
mk      ←1
Δ004:
Δ006←mk≤12
→(∼Δ006)/Δ005
Δ007←1=DMA,1,mk,ak
→(∼Δ007)/Δ008
soma←soma+1
Δ008:
mk      ←mk      + 1.00000
→Δ004
Δ005:
ak      ←ak      + 1.00000
→Δ001
Δ002:

```

soma

eu019

171

Capítulo 20

Problema 20: Soma dos dígitos do fatorial de 100

$n!$ significa $n \times (n-1) \times \dots \times 3 \times 2 \times 1$. Por exemplo, $!10 = 10 \times 9 \times \dots \times 3 \times 2 \times 1 = 3638800$, e a soma dos dígitos no número $!10$ é $3 + 6 + 2 + 8 + 8 + 0 + 0 = 27$. Ache a soma dos dígitos no número $!100$.

A resposta é 648.

20.1 J

```
+/(("#":!100x)#10)#:!100x
648
```

20.2 Python

Aproveitando a maravilhosa capacidade de Python para *bignum*.

```
def fac(n):
    if n==0: return 1
    else: return n*fac(n-1)
print (sum(int(n) for n in str(fac(100))))
648
```

Capítulo 21

Problema 21: Soma dos números amigos abaixo de 10000

No original conforme o site projecteuler.net

Let $d(n)$ be defined as the sum of proper divisors of n (numbers less than n which divide evenly into n). If $d(a) = b$ and $d(b) = a$, where $a \neq b$, then a and b are an amicable pair and each of a and b are called amicable numbers.

For example, the proper divisors of 220 are 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 and 110; therefore $d(220) = 284$. The proper divisors of 284 are 1, 2, 4, 71 and 142; so $d(284) = 220$.

Evaluate the sum of all the amicable numbers under 10000.

Answer: 31626

Números amigos Seja $d(n)$ a soma dos divisores próprios de n (os divisores próprios são os inteiros menores que n que dividem n sem deixar resto). Se $d(a) = b$ e $d(b) = a$, e $a \neq b$, então diz-se que a e b formam um par amigo e cada um deles a ou b são chamados números amigos. Por exemplo, os divisores próprios de 220 são 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 e 110. Assim $d(220) = 284$. Os divisores próprios de 284 são 1, 2, 4, 71 e 142, então $d(284) = 220$. Assim, a dupla (220,284) forma a primeira dupla de números amigos.

21.1 Solução em Gemini

quais as duplas de números amigos abaixo de 10000 ?

Números amigos são pares de números inteiros positivos em que a soma dos divisores próprios de um número é igual ao outro número, e vice-versa. Abaixo de 10.000, existem as seguintes duplas de números amigos:

220 e 284:

Os divisores próprios de 220 (excluindo o próprio 220) são: 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 e 110. A soma desses divisores é 284.

Os divisores próprios de 284 são: 1, 2, 4, 71 e 142. A soma desses divisores é 220.

1184 e 1210:

A soma dos divisores próprios de 1184 é 1210.

A soma dos divisores próprios de 1210 é 1184.

2620 e 2924:

A soma dos divisores próprios de 2620 é 2924.

A soma dos divisores próprios de 2924 é 2620.

5020 e 5564:

A soma dos divisores próprios de 5020 é 5564.

A soma dos divisores próprios de 5564 é 5020.

6232 e 6368:

A soma dos divisores próprios de 6232 é 6368.

A soma dos divisores próprios de 6368 é 6232.

Estes são os pares de números amigos que você encontrará abaixo de 10.000.

Usando a calculadora, eu somei estes números e achei 31626.

21.2 Solução em APL

```

      ∇euler21;i;ta;ia;soma
[1]   i←1
[2]   ta←10000p0
[3]   t1:→(i>9999)/fim
[4]   ta[i]←+/-1↓divs21 i
[5]   i←i+1
[6]   →t1
[7]   fim:soma←0
[8]   i←2
[9]   t3:→(i>9999)/mesmo
[10]  ia←ta[i]
[11]  →(ia>9999)/sss
[12]  →(ta[ia]=i)/sao
[13]  sss:i←i+1
[14]  →t3
[15]  sao:→(ta[i]=ta[ia])/sss
[16]  soma←soma+ta[i]+ta[ia]
[17]  i,ia
[18]  i←i+1
[19]  →t3
[20]  mesmo:soma÷2

```

```

      ∇r←divs21 x;di;som;qua
[1]   di←2
[2]   som←1
[3]   qua←,1
[4]   t0:→(di>(x*0.5))/fim
[5]   t1:→(0=di|x)/sim
[6]   di←di+1
[7]   →t0
[8]   sim:qua←qua,di
[9]   di←di+1
[10]  →t0
[11]  fim:→((di-1)=x*0.5)/qp
[12]  r←qua,0x÷qua
[13]  →ju
[14]  qp:r←qua,1↓0x÷qua
[15]  ju:

```

21.3 Solução em J

```

di =: 3 : 0
  lim =. y^0.5
  som =. 1
  if. lim = <. lim do.
    som =. som - lim
  end.
  ii=.2
  while. ii<: lim do.
    if. 0=ii|y do.
      som=.som+ii+y%ii
    end.
    ii=.ii+1
  end.
  if. y=0 do.
    0
  else.

```

```

        som
    end.
)
euler21 =: 3 : 0
    tab =. 10000 $ 0
    for_i. i.10000 do.
        k=.di i
        if. k<:10000 do.
            tab=. k i}tab
        end.
    end.
    som=.0
    for_i. i.10000 do.
        j =. i{tab
        if. (i{tab = j) *. (j{tab = i) *. (i~:j) do.
            som=.som+i+j
        end.
    end.
    som%2
)

```

```

    load '/p/quarta/euler21.ijs'
    euler21 ''
31626

```

Capítulo 22

Problema 22 - Escores de nomes

No original conforme o site projecteuler.net

Using names.txt (right click and 'Save Link/Target As...'), a 46K text file containing over five-thousand first names, begin by sorting it into alphabetical order. Then working out the alphabetical value for each name, multiply this value by its alphabetical position in the list to obtain a name score.

For example, when the list is sorted into alphabetical order, COLIN, which is worth $3 + 15 + 12 + 9 + 14 = 53$, is the 938th name in the list. So, COLIN would obtain a score of $938 \times 53 = 49714$.

What is the total of all the name scores in the file?

Answer: 871198282

Escores de nomes Leia o arquivo NOMES.TXT (46KB) formado por uma certa quantidade de primeiros nomes, em torno de 5000. A seguir ordene os demais em ordem alfabética crescente. Depois, multiplique o valor alfabético de cada nome pela sua posição na lista para obter a pontuação de cada nome. Pergunta-se qual a soma das pontuações de todos os nomes da lista? Por exemplo, COLIN, tem um valor de $3+15+12+9+14=53$. Se ele ocupar a posição 1001 na lista, sua pontuação será: $53 \times 1001 = 53053$. $N =$

22.1 Solução em APL

```

      r←euler22;x;alfa;tam;i;vets
[1]  x←HFM8 'c:/p/n/e05/names22.txt'  barras ao contrário
[2]  x←x
[3]  x←(' '≠x)/x
[4]  NO22←⌵(∼x∈',')⊂x
[5]  NO22←NO22[⌵AV⌵NO22;]
[6]  alfa←'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
[7]  tam←0
[8]  i←1
[9]  t1:→(i>1↑pNO22)/fim
[10]  vets←+/alfa⌵((NO22[i;]≠' ')/NO22[i;])
[11]  tam←tam+(vets×i)
[12]  i←i+1
[13]  →t1
[14]  fim:r←tam

      euler22
871198282
```

22.2 Solução em J

```

euler22 =: 3 : 0
  z =: >'m' fread <'c:\p\n\e05\names22.txt'
  z =: ,z
  z=:(' '~:z)#z    NB. tira as dupla aspas
```

```

  alfa=. 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
  N022=:,. /:~ dltb each ',,' cut z NB. copieie fa phrases/strings do J
  su=.0
  for_i. i.1{.$N022 do.
    pa=.,>i{N022
    su=.su+(i+1)* +/1+alfa i. pa
  end.
su
)

load '/p/quarta/euler22.ijs'
euler22 ''
871198282
```

Capítulo 23

Problema 23: Soma de não abundantes

No original conforme o site projecteuler.net

A perfect number is a number for which the sum of its proper divisors is exactly equal to the number. For example, the sum of the proper divisors of 28 would be $1 + 2 + 4 + 7 + 14 = 28$, which means that 28 is a perfect number.

A number n is called deficient if the sum of its proper divisors is less than n and it is called abundant if this sum exceeds n .

As 12 is the smallest abundant number, $1 + 2 + 3 + 4 + 6 = 16$, the smallest number that can be written as the sum of two abundant numbers is 24. By mathematical analysis, it can be shown that all integers greater than 28123 can be written as the sum of two abundant numbers. However, this upper limit cannot be reduced any further by analysis even though it is known that the greatest number that cannot be expressed as the sum of two abundant numbers is less than this limit.

Find the sum of all the positive integers which cannot be written as the sum of two abundant numbers.

Answer: 4179871

Somas não abundantes Um número perfeito é aquele para o qual a soma dos seus divisores próprios é exatamente igual ao número. Assim, a soma dos divisores próprios de 28 é $1 + 2 + 4 + 7 + 14 = 28$, logo 28 é um número perfeito. Um número n é chamado deficiente se a soma de seus divisores próprios é menor do que n e é chamado abundante se a soma excede n . O menor número abundante é 12 ($1+2+3+4+6=16$). Daqui, o menor número natural que pode ser escrito como a soma de dois abundantes é 24. Da análise matemática, qualquer inteiro superior a 28123 pode ser escrito como a soma de dois abundantes. No entanto, este limite superior não pode ser reduzido ainda mais pela análise, embora se saiba que o maior número que não pode ser expresso como a soma de dois números abundantes é menor que este limite.

Encontre a soma de todos os inteiros positivos que não podem ser escritos como a soma de dois números abundantes.

23.1 Solução em APL

```

    ∇r←euler23;x;i;abu;sebo;j;som
[1]  A achar os abundantes
[2]  x←28123p0
[3]  i←1
[4]  t1:→(i>28123)/fim
[5]  x[i]←+/divs23 i
[6]  i←i+1
[7]  →t1
[8]  fim:i←1
[9]  t2:→(i>28123)/mesmo
[10] →(x[i]≤i)/soma
[11] i←i+1
[12] →t2
[13] soma:
[14] x[i]←0
[15] i←i+1
[16] →t2
[17] mesmo:abu←(x≠0)/ι28123
[18]  sebo←ι28123
[19]  i←1
[20] t3:→(i>6965)/fim33
```

```

[21] j←1
[22] t4:→(j>6965)/fim44
[23] som←abu[i]+abu[j]
[24] →(som>28123)/som66
[25] sebo[som]←0
[26] som66:j←j+1
[27] →t4
[28] fim44:i←i+1
[29] →t3
[30] fim33:+/sebo

```

```

      ∇r←divs23 x;di;som;qua
[1] di←2
[2] som←1
[3] qua←,1
[4] t0:→(di>(x*0.5))/fim
[5] t1:→(0=di|x)/sim
[6] di←di+1
[7] →t0
[8] sim:qua←qua,di
[9] di←di+1
[10] →t0
[11] fim:→((di-1)=x*0.5)/qp
[12] r←~1↓qua,0x÷qua
[13] →ju
[14] qp:r←~1↓qua,1↓0x÷qua
[15] ju:

```

```

      euler23
4179871

```

23.2 Solução em J

```

di =: 3 : 0
  lim =. y^0.5
  som =. 1
  if. lim = <. lim do.
    som =. som - lim
  end.
  ii=.2
  while. ii<: lim do.
    if. 0=ii|y do.
      som=.som+ii+y%ii
    end.
    ii=.ii+1
  end.
  if. y=0 do.
    0
  else.
    som
  end.
)
euler23 =: 3 : 0
  xx=.28123$0
  for_i. i.28123 do.
    so=. di i
    if. so > i do.
      xx=. (di i) i}xx
    end.
  end.
end.

```

```
abu=.(xx~:0)#i.28123
sebo=.i.28123
for_i. i.6965 do.
  for_j. i.6965 do.
    so=.i{abu + j{abu
    if. so<28123 do.
      sebo=. 0 so }sebo
    end.
  end.
end.
+/sebo
)

load '/p/quarta/euler23.ijs'
euler23 ''
4179871
```

Capítulo 24

Problema 24 - Qual a milionésima permutação de 0..9

No original conforme o site projecteuler.net

A permutation is an ordered arrangement of objects. For example, 3124 is one possible permutation of the digits 1, 2, 3 and 4. If all of the permutations are listed numerically or alphabetically, we call it lexicographic order. The lexicographic permutations of 0, 1 and 2 are:

012021102120201210

What is the millionth lexicographic permutation of the digits 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9?

Answer: 2783915460 Completed on Mon, 25 Mar 2013, 17:12

Permutações lexicográficas A permutação é um arranjo ordenado de objetos. Por exemplo, 3124 é uma das permutações possíveis dos dígitos 1,2,3 e 4. Se todas as permutações são listadas numérica ou alfabeticamente diz-se que estão em ordem lexicográfica. As permutações lexicográficas de 0, 1 e 2 são:

012 021 102 120 201 210

Considere a permutação lexicográfica dos dígitos 0,1,2,3,4,5,6,7,8 e 9. Indique qual a milionésima permutação de 0..9 ?

24.1 Solução em APL

Tem uma solução no WS, mas parece ser meio lerdinha...

24.2 Solução em J

```
1000000 A. '0123456789'
2783915604 NB. p.q.p.
```

Capítulo 25

Problema 25 - qual o termo fibonacci a ter mais de 1000 dígitos

No original conforme o site projecteuler.net

The Fibonacci sequence is defined by the recurrence relation:

$F_n = F_{n-1} + F_{n-2}$, where $F_1 = 1$ and $F_2 = 1$. Hence the first 12 terms will be:

$F_1 = 1$

$F_2 = 1$

$F_3 = 2$

$F_4 = 3$

$F_5 = 5$

$F_6 = 8$

$F_7 = 13$

$F_8 = 21$

$F_9 = 34$

$F_{10} = 55$

$F_{11} = 89$

$F_{12} = 144$

The 12th term, F_{12} , is the first term to contain three digits.

What is the index of the first term in the Fibonacci sequence to contain 1000 digits?

Answer: 4782 Completed on Tue, 26 Mar 2013, 00:21

Fibonacci grande A sequência de Fibonacci é definida pela relação de recorrência:

$$F_n = F_{n-1} + F_{n-2}$$

onde $F_1 = F_2 = 1$. Desta maneira, os primeiros 12 termos são: $F_1 = 1$, $F_2 = 1$, $F_3 = 2$, $F_4 = 3$, $F_5 = 5$, $F_6 = 8$, $F_7 = 13$, $F_8 = 21$, $F_9 = 34$, $F_{10} = 55$, $F_{11} = 89$, $F_{12} = 144$. O *décimo segundo* termo, F_{12} é o primeiro termo a conter três dígitos. Qual o primeiro termo da sequência de Fibonacci a conter mais de 1000 dígitos ?

25.1 Solução em APL

```
∇euler25;a;b;n;r
[1] b←(999ρ'0'),'1'
[2] n←3
[3] r←(999ρ'0'),'2'
[4] vorta:→(r[2]≠'0')/fim
[5] n←n+1
[6] a←b
[7] b←r
[8] r←a soma25 b
[9] →vorta
[10] fim:n
∇ 2024-06-18 12.36.49
∇soma25[⍵]∇
∇r←a soma25 b;a1;b1;temp;vai;i
```

```

[1] a1←-1000†(1000ρ'0'),a
[2] b1←-1000†(1000ρ'0'),b
[3] r←1001ρ'0'
[4] vai←0
[5] i←1000
[6] vorta:→(i<1)/fim
[7] temp←vai+(‡a1[i])+‡b1[i]
[8] →(temp>9)/foi
[9] r[i+1]←1 0‡temp
[10] vai←0
[11] →soma
[12] foi:r[i+1]←1 0‡temp-10
[13] vai←1
[14] soma:i←i-1
[15] →vorta
[16] fim:→(vai=0)/0
[17] r[1]←'1'

```

euler25

4782

25.2 Solução em J

```

euler25 =: 3 : 0
    a=.1x
    b=.1x
    r=.2x
    c=.3
    r=.a+b
    while. 1=1 do.
        a=.b
        b=.r
        r=.a+b
        c=.c+1
        if. (#":r)>999 do.
            c
            return.
        end.
    end.
end.

    load '/p/quarta/euler25.ijs'
    euler25''
4782

```

Capítulo 26

Problema 26 - Maior ciclo recorrente de inversa

No original conforme o site projecteuler.net

A unit fraction contains 1 in the numerator. The decimal representation of the unit fractions with denominators 2 to 10 are given:

$1/2 = 0.5$
 $1/3 = 0.(3)$
 $1/4 = 0.25$
 $1/5 = 0.2$
 $1/6 = 0.1(6)$
 $1/7 = 0.(142857)$
 $1/8 = 0.125$
 $1/9 = 0.(1)$
 $1/10 = 0.1$

Where $0.1(6)$ means $0.166666\dots$, and has a 1-digit recurring cycle. It can be seen that $1/7$ has a 6-digit recurring cycle.

Find the value of $d < 1000$ for which $1/d$ contains the longest recurring cycle in its decimal fraction part.

Answer: 983 Completed on Thu, 28 Mar 2013, 12:47

Ciclos recíprocos Uma fração unitária contém 1 no numerador. A representação decimal das frações unitárias cujo denominador vai de 2 a 10 são é: $1/2 = 0.5$, $1/3 = 0.(3)$, $1/4 = 0.25$, $1/5 = 0.2$, $1/6 = 0.1(6)$, $1/7 = 0.(142857)$, $1/8 = 0.125$, $1/9 = 0.(1)$, $1/10 = 0.1$. Aqui, $0.1(6)$ significa $0.16666\dots$, e há aqui um ciclo de 1 dígito. Pode-se ver que $1/7$ tem um ciclo recorrente de 6 dígitos. Ache qual o valor de d , para o qual $1/d$ contém o mais longo ciclo recorrente na sua parte da fração decimal.

26.1 Solução em APL

```

    Veuler26;i;max;s;imax
[1] i←1
[2] max←-99999
[3] t1:→(i>999)/fim
[4] s←detectcicl26 i
[5] →(s≤max)/soma
[6] max←s
[7] imax←i
[8] soma:i←i+1
[9] →t1
[10] fim:
[11] 'max=',⍉max
[12] 'Resposta >>>>> imax=',⍉imax

    Vr←detectcicl26 n;aa;x;i;y
[1] aa←1
[2] vorta:x←(100×aa)divisao26 n
[3] →(x=0)/r0
[4] i←(10+?80)+15
[5] y←(x[i]⊖x)/1px
[6] →(3<py)/ok
```

```

[7] aa←aa×10
[8] →vorta
[9] ok:r←y[py]-y[-1+py]
[10] →0
[11] r0:r←0

    ∇r←lim divisao26 n;x;aa
[1] →((n=1)∨(n=10)∨(n=100))/acabou
[2] r←'0.'
[3] x←10
[4] t1:→(x>n)/fi
[5] x←x×10
[6] r←r,'0'
[7] →t1
[8] fi:r←r,1 0⊘aa←⌊x÷n
[9] →(lim<pr)/0
[10] x←10×x-n×aa
[11] →(x=0)/acabou
[12] →t1
[13] acabou:r←0
[14] →0
[15]

    euler26
max=982
Resposta >>>>> imax=983

```

26.2 Solução em J

```

divisao26 =: 4 : 0
  if. (y=1) +. (y=10) +. (y=100) do. goto_acabou. end.
  r=.'0.'
  xx=. 10
  label_t1.
    if. xx> y do. goto-fi. end.
    xx=.xx*10
    r=.r,'0'
    goto_t1.
  label-fi.
    r=.r,1j0":aa=.<.xx%y
    if. x<#r do. r
      return. end.
    xx=.10*xx-y*aa
    if. xx=0 do. goto_acabou. end.
  goto_t1.
  label_acabou.
  0
)
detectcicl26 =: 3 : 0
  aa=.1
  label_vorta.
  xx=.(100*aa) divisao26 y
  if. xx=0 do. goto_r0. end.
  i=.(10+?80)+i.5
  yy=. ((i{xx)E.xx)#i.#xx
  if. 3<#yy do. goto_ok. end.
  aa=.aa*10
  goto_vorta.
  label_ok.
  r=.((_1+#yy){yy)-(_2+#yy){yy

```

```
    return.
    label_r0.
    0
)

euler26=: 3 : 0
    ii=.1
    max=.-99999
    while. ii<:999 do.
        s=.detectcicl26 ii
        if. s>max do.
            max=.s
            imax=.ii
        end.
        ii=.ii+1
    end.
    max,imax
)

    load 'c:/p/quarta/euler26.ijs'
    euler26 ''
982 983
```

Capítulo 27

Problema 27 - Maior ciclo de primos quadráticos

No original conforme o site projecteuler.net
Euler discovered the remarkable quadratic formula:

$$n^2 + n + 41$$

It turns out that the formula will produce 40 primes for the consecutive values $n = 0$ to 39. However, when $n = 40$, $40^2 + 40 + 41 = 40(40 + 1) + 41$ is divisible by 41, and certainly when $n = 41$, $41^2 + 41 + 41$ is clearly divisible by 41.

The incredible formula $n^2 - 79n + 1601$ was discovered, which produces 80 primes for the consecutive values $n = 0$ to 79. The product of the coefficients, -79 and 1601 , is -126479 .

Considering quadratics of the form:

$$n^2 + an + b, \text{ where } |a| < 1000 \text{ and } |b| < 1000$$

where $|n|$ is the modulus/absolute value of n e.g. $|11| = 11$ and $|-4| = 4$ Find the product of the coefficients, a and b , for the quadratic expression that produces the maximum number of primes for consecutive values of n , starting with $n = 0$.

Answer: -59231, with $a=-61$ and $b=971$. $\text{max}=71$.

Primos quadráticos Euler publicou a fórmula quadrática notável

$$n^2 + n + 41$$

Esta fórmula produz 40 primos para os valores consecutivos de n de 0 a 39. Entretanto, quando $n = 40$, $40^2 + 40 + 41 = 40(40 + 1) + 41$ é divisível por 41 e certamente quando $n = 41$, $41^2 + 41 + 41$ é claramente divisível por 41. Usando computadores, a fórmula $n^2 - 79n + 1601$ foi descoberta. Ela produz 80 primos para os valores consecutivos de $n = 0..79$. O produto dos dois coeficientes -79 e 1601 é -126479 . Considere quadráticas na forma $n^2 + an + b$, onde $|a| < LIMITE$ e $|b| < LIMITE$, onde $|n|$ é o módulo/valor absoluto de n . Por exemplo, $|11| = 11$ e $|-4| = 4$. Ache o produto dos coeficientes a e b para a expressão quadrática que produz o maior número de primos para valores consecutivos de n , começando com $n = 0$, quando $LIMITE = 1000$

27.1 Solução em APL

```
⎕euler27 lim;a;b;ia;ib;imax;max;n;qtp;x
[1] a←-lim
[2] max←-9999999
[3] t1:→(a>lim)/fim0
[4] b←-lim
[5] t2:→(b>lim)/truco
[6] A →(~b∈PRIMO)/fui
[7] A →(~(a+b+1)∈PRIMO)/fui
[8] n←0
[9] qtp←0
[10] t3:→((n>|a)^(n>|b))/troca
[11] x←(n*2)+(a*n)+b
[12] →(~x∈PRIMO)/troca
[13] sim:qtp←qtp+1
```

```
[14]  n←n+1
[15]  →t3
[16]  troca:→(max>qtp)/fui
[17]  max←qtp
[18]  imax←a×b
[19]  ia←a
[20]  ib←b
[21]  fui:b←b+1
[22]  →t2
[23]  truco:a←a+1
[24]  →(0≠100|a)/t1
[25]  a
[26]  →t1
[27]  fim0:
[28]  'max=',␣max
[29]  'imax=',␣imax
[30]  'a=',␣ia
[31]  'b=',␣ib
```

Deve-se notar que PRIMO é uma variável no ws, na forma de vetor numérico com 148933 números primos menores que 2000000.

Capítulo 28

Problema 28 - Soma das diagonais

No original conforme o site projecteuler.net

Starting with the number 1 and moving to the right in a clockwise direction a 5 by 5 spiral is formed as follows:

```
21 22 23 24 25
20 7 8 9 10
19 6 1 2 11
18 5 4 3 12
17 16 15 14 13
```

It can be verified that the sum of the numbers on the diagonals is 101.

What is the sum of the numbers on the diagonals in a 1001 by 1001 spiral formed in the same way?

Answer: 669171001 Completed on Tue, 26 Mar 2013, 08:25

Diagonais em números espiralados Começando com o número 1 e movendo-se para a direita na direção do movimento dos ponteiros do relógio em uma matriz 5×5 , uma espiral é formada como se vê:

```
21 22 23 24 25
20 7 8 9 10
19 6 1 2 11
18 5 4 3 12
17 16 15 14 13
```

Neste exemplo, a soma dos números nas diagonais é 101. Qual é a soma dos números nas diagonais de uma espiral igualmente formada e contendo 1001 linhas por 1001 colunas ?

28.1 Solução em APL

```
∇r←euler28a x
[1] →(x<2)/um
[2] r←(4×x×2)+(-(6×(x-1)))+euler28a(x-2)
[3] →0
[4] um:r←1

euler28a 1001
669171001
```

28.2 Solução em J

```
euler28=: 3 : 0
  if. y>2 do.
    (4*y^2)+(-(6*(y-1)))+euler28 y-2
  else.
    1
  end.
)
```

20j0":euler28 1001
669171001

Capítulo 29

Problema 29 - Termos distintos em a^b e b^a

Considere todas as combinações de a^b para $2 \leq a \leq 5$ e $2 \leq b \leq 5$: $2^2 = 4$, $2^3 = 8$, $2^4 = 16$, $2^5 = 32$, $3^2 = 9$, $3^3 = 27$, $3^4 = 81$, $3^5 = 243$, $4^2 = 16$, $4^3 = 64$, $4^4 = 256$, $4^5 = 1024$, $5^2 = 25$, $5^3 = 125$, $5^4 = 625$, $5^5 = 3125$. Se os números forem colocados em ordem numérica crescente, removendo-se qualquer repetição obtém-se a seguinte sequência de 15 termos distintos:

4, 8, 9, 16, 25, 27, 32, 64, 81, 125, 243, 256, 625, 1024, 3125

Quantos termos diferentes há na sequência gerada por a^b para $2 \leq a \leq 100$ e $2 \leq b \leq 100$?

29.1 Solução em APL

primeiro, o tamanho padrão de ws (20MB) revelou-se insuficiente para rodar este programa. Daí precisou-se)CLEAR 100 MB e depois)COPY...

```

      ∇euler29[⍵]∇
      ∇r←euler29;a;b;j
[1]  x←9801 210ρ' '
[2]  a←2
[3]  j←1
[4]  t1:→(a>100)/fim
[5]  b←2
[6]  t2:
[7]  →(b>100)/trocaa
[8]  x[j;]←(⌘a)pot29⌘b
[9]  j←j+1
[10] b←b+1
[11] →t2
[12] trocaa:a←a+1
[13] a
[14] →t1
[15] fim:r←x
      ∇ 2025-03-04 20.28.35
      ∇pot29[⍵]∇
      ∇r←a pot29 b;ba
[1]  ba←⊥b
[2]  a←⌘a
[3]  r←'1'
[4]  t1:→(ba=0)/fim
[5]  r←a mult29 r
[6]  ba←ba-1
[7]  →t1
[8]  fim:
      ∇ 2025-03-04 20.28.35
      ∇mult29[⍵]∇
      ∇rr←a mult29 b;a1;b1;r;i;vai;temp
[1]  →(3=+/'100'=3↑a)/cem
[2]  a1←-2↑'00',a
```

```

[3]   b1←-210†(210ρ'0'),b
[4]   r←3 210ρ' '
[5]   i←210
[6]   vai←0
[7]   t1:→(i<2)/fim1
[8]   temp←vai+(a1[2])×b1[i]
[9]   →(temp>9)/foi1
[10]  r[1;i]←1 0temp
[11]  vai←0
[12]  i←i-1
[13]  →t1
[14]  foi1:
[15]  r[1;i]←1 010|temp
[16]  vai←|temp÷10
[17]  i←i-1
[18]  →t1
[19]  fim1:
[20]  i←210
[21]  r[2;210]←'0'
[22]  vai←0
[23]  t2:→(i<2)/fim2
[24]  temp←vai+(a1[1])×b1[i]
[25]  →(temp>9)/foi2
[26]  r[2;i-1]←1 0temp
[27]  vai←0
[28]  i←i-1
[29]  →t2
[30]  foi2:
[31]  r[2;i-1]←1 010|temp
[32]  vai←|temp÷10
[33]  i←i-1
[34]  →t2
[35]  fim2:i←210
[36]  vai←0
[37]  t3:→(i<2)/ufa
[38]  temp←vai+(r[1;i])+r[2;i]
[39]  →(temp>9)/cri
[40]  r[3;i]←1 0temp
[41]  vai←0
[42]  i←i-1
[43]  →t3
[44]  cri:
[45]  r[3;i]←1 0temp-10
[46]  vai←1
[47]  i←i-1
[48]  →t3
[49]  ufa:rr←r[3;]
[50]  →0
[51]  cem:
[52]  rr←(208†(210ρ'0'),b),'00'
      ∇ 2025-03-04 20.28.35
      ∇euler292p[ ]∇
      ∇euler292p x;ct;i
[1]   x←' 0' edit x
[2]   x←x[AVx;]
[3]   ct←0
[4]   i←2
[5]   t1:→(i>1†px)/fim
[6]   →(x[i;]igual29 x[i-1;])/iguais
[7]   diferentes:
[8]   ct←ct+1
[9]   i←i+1
[10]  →t1

```

```

[11] iguais:i←i+1
[12] →t1
[13] fim:ct←ct+1
[14] ct
      ▽ 2025-03-04 20.28.35
      ▽igual29[ ]▽
      ▽r←a igual29 b;i
[1] i←p,a
[2] r←0
[3] t1:→(i<3)/fim
[4] →(a[i]=b[i])/sim
[5] nao:→0
[6] sim:i←i-1
[7] →t1
[8] fim:r←1

      euler292p rr
9183

```

29.2 Solução em J

Perceba como a facilidade de número estendido do J simplifica este processamento.

```

euler29=: 3 : 0
  aa=:((99*99),210)$' '
  ind=:0
  for_i. 2+i.99 do.
    ii=:i*1x
    for_j. 2+i.99 do.
      jj=:j*1x
      bb=:ii^jj
      cc=:210j0 ": bb
      if. -. cc e. aa do.
        aa=:cc ind}aa
        ind=:ind+1
      end.
    end.
  end.
  ind
)

      load 'c:/p/quarta/euler29.ijs'
      euler29''
9183

```

Capítulo 30

Problema 30 - Números iguais à soma da potência quinta de seus dígitos

No original conforme o site projecteuler.net

Surprisingly there are only three numbers that can be written as the sum of fourth powers of their digits:

$$1634 = 1^4 + 6^4 + 3^4 + 4^4$$

$$8208 = 8^4 + 2^4 + 0^4 + 8^4$$

$$9474 = 9^4 + 4^4 + 7^4 + 4^4$$

As $1 = 1^4$ is not a sum it is not included.

The sum of these numbers is $1634 + 8208 + 9474 = 19316$.

Find the sum of all the numbers that can be written as the sum of fifth powers of their digits.

Answer: 443839 Completed on Fri, 29 Mar 2013, 18:45

Potências de 5 Surpreendentemente só há 3 números que podem ser escritos como a soma das quartas potências de seus dígitos. $1634 = 1^4 + 6^4 + 3^4 + 4^4$, $8208 = 8^4 + 2^4 + 0^4 + 8^4$ e $9474 = 9^4 + 4^4 + 7^4 + 4^4$. Note que $1^4 = 1$ não é uma soma e não é incluído. A soma destes números é $1634 + 8208 + 9474 = 19316$. Seguindo em frente, há apenas 6 números que podem ser escritos como a soma das potências 5 de seus dígitos. Ache a soma dos 6 números que são iguais à soma das potências quintas de seus dígitos.

30.1 Solução em APL

```
∇euler30; som; sp; a1; i; t; qt
[1] som←0
[2] i←2
[3] qt←0
[4] t1:→(qt≥6)/fim
[5] a1←⌊i
[6] t←1
[7] sp←0
[8] t2:→(t>pa1)/t3
[9] sp←sp+(a1[t])*5
[10] t←t+1
[11] →t2
[12] t3:→(i=sp)/sim
[13] nao:i←i+1
[14] →t1
[15] sim:
[16] qt←qt+1
[17] i
[18] som←som+i
[19] i←i+1
[20] →t1
[21] fim:'som=',⌈som
```

```

euler30
4150
4151
54748
92727
93084
194979
som=443839

```

30.2 Solução em J

A seguir uma resposta obtida no site [projecteuler](http://projecteuler.net) feita em J por um ucraniano de nome olegik

```

s5=: +/@:(^&5)@:( "."0)@": "0
+/(#~ (= s5))2+i.1e6
443839

```

A seguir, a minha solução em J, menos críptica

```

euler30 =: 3 : 0
    som=. 0
    i=.2
    qt=.0
    while. qt<6 do.
        a1=":i
        t=.0
        sp=.0
        while. t< $a1 do.
            sp=.sp+("t}a1)^5
            t=.t+1
        end.
        if. i=sp do.
            som=.som+i
            qt=.qt+1
        end.
        i=.i+1
    end.
    som
)

load 'c:/p/quarta/euler30.ijs'
euler30 ''
443839

```

Capítulo 31

Problema 31 - Quantas maneiras de obter 2 libras

No original conforme o site projecteuler.net

In England the currency is made up of pound, £, and pence, p, and there are eight coins in general circulation:

1p, 2p, 5p, 10p, 20p, 50p, £1 (100p) and £2 (200p).

It is possible to make £2 in the following way:

$1 \times £1 + 1 \times 50p + 2 \times 20p + 1 \times 5p + 1 \times 2p + 3 \times 1p$

How many different ways can £2 be made using any number of coins?

Answer: 73682

31.1 Solução em APL

Uma solução bem “força bruta”, em que todas as possibilidades são testadas. Não é eficiente, e se o valor aumentar, o tempo cresce muito mais. Vamos lá

```

    ∇euler31;b;c;d;e;f;g;h;ct
[1]   ct←1 A 1 moeda de 2L
[2]   b←0
[3]   t1:→(b>2)/fim
[4]   c←0
[5]   t2:→(c>4)/c4
[6]   d←0
[7]   t3:→(d>10)/d4
[8]   e←0
[9]   t4:→(e>20)/e4
[10]  f←0
[11]  t5:→(f>40)/f4
[12]  g←0
[13]  t6:→(g>100)/g4
[14]  h←0
[15]  t7:→(h>200)/h4
[16]  →(200<(b×100)+(c×50)+(d×20)+(e×10)+(f×5)+(g×2)+h)/h4
[17]  →(200≠(b×100)+(c×50)+(d×20)+(e×10)+(f×5)+(g×2)+h)/nex
[18]  ct,b,c,d,e,f,g,h
[19]  ct←ct+1
[20]
[21]  nex:
[22]  h←h+1
[23]  →t7
[24]  h4:
[25]  g←g+1
[26]  →t6
[27]  g4:
[28]  f←f+1
[29]  →t5
```

```

[30] f4:
[31]   e←e+1
[32]   →t4
[33] e4:
[34]   d←d+1
[35]   →t3
[36] d4:
[37]   c←c+1
[38]   →t2
[39] c4:
[40]   b←b+1
[41]   →t1
[42] fim:ct

```

```

      euler31
73682

```

31.2 Solução em J

Para a solução em J, consultei um texto no fórum da solução do problema 31 no site [projecteuler](http://projecteuler.net). Vamos lá

```

coins=:1 2 5 10 20 50 100 200
amount =: 200
ways=: 4 : 0
  if. y <:0 do.
    1
    return.
  end.
  res=.0
  while. x >: 0 do.
    res =. res+x ways y-1
    x =. x - y{coins
  end.
  res
)
euler31 =: 3 : 0
  amount ways 7
)

load 'c:/p/quarta/euler31.ijs'
euler31 ''
73682

```

Capítulo 32

Problema 32 - Produtos pandigitais

No original conforme o site projecteuler.net

We shall say that an n -digit number is pandigital if it makes use of all the digits 1 to n exactly once; for example, the 5-digit number, 15234, is 1 through 5 pandigital.

The product 7254 is unusual, as the identity, $39 \times 186 = 7254$, containing multiplicand, multiplier, and product is 1 through 9 pandigital.

Find the sum of all products whose multiplicand/multiplier/product identity can be written as a 1 through 9 pandigital.

HINT: Some products can be obtained in more than one way so be sure to only include it once in your sum.

Answer: 45228

Produtos pandigitais Diz-se que um número de n dígitos é pandigital se ele usa todos os dígitos entre 1.. n exatamente uma vez. Por exemplo, o número de 5 dígitos 15324 é pandigital de 1 a 5. O produto 7254 pode ser escrito como a identidade $39 \times 186 = 7254$. Aqui, contando o multiplicando, multiplicador e produto, tem-se uma expressão pandigital de 1 a 9. Considerando todos os produtos cujas identidades (envolvendo multiplicando, multiplicador e produto) são pandigitais de 1 a 9, ache a soma pedida. Note que alguns produtos podem ser obtidos em mais de uma expressão e devem ser contados apenas uma vez. No total são 7 produtos. O 7254 é o quinto. Ache a soma de todos os produtos cujo multiplicando, multiplicador e produto podem ser escritos como pandigitais de 1 a 9.

32.1 Solução em APL

Uma solução bem força bruta e portanto, muito demorada

```

    veuler32; i; quem; a1; b1; b2; c1
[1] i←123456789
[2] quem←10
[3] t1:→(i>987654321)/fim
[4] a1←⌈i
[5] →('0'∈a1)/nao
[6] →(9≠+/'123456789'∈⌈i)/nao
[7] b1←1 1 1 1 0 1 1 0 1 1 1\ a1
[8] b2←1 1 1 1 0 1 0 1 1 1 1\ a1
[9] b1[5]+b2[5]←'='
[10] b1[8]+b2[7]←'×'
[11] →(1=⌊b1)/sim
[12] →(1=⌊b2)/sim
[13] nao:i←i+1
[14] →t1
[15] sim:
[16] →(9≠+/'123456789'∈⌈b1)/nao
[17] →(9≠+/'123456789'∈⌈b2)/nao
[18] →((c1←⌊4↑b1)∈quem)/nao
[19] quem←quem,c1
[20] →nao
[21] fim:+/quem
```

euler32

45228

32.2 Solução em J

Primeiro uma solução de ninguém menos do que Roger Hui, que esta no forum de respostas do problema 32 no project euler.

```
p=: (10+i.90) (* / ,&.>& ,&.> /) 100+i. 900
p=: p,( 1+i. 9) (* / ,&.>& ,&.> /) 1000+i.9000
p=: p #~ 1e4 > {.&>p
p=: > p #~ ' 123456789' -: "1 /:~"1 " :>p
+ / ~. { ."1 p
45228
```

A seguir a minha resposta:

```
euler32 =: 3 : 0
i=:123456789
quem=: i.0
while. i<:987654321 do.
  a1=:":i
  if. (-.'0'e.a1) *. (9=+/'123456789'e.":i) do.
    b1=(0 1 2 3{a1),'=',(4 5{a1),'*',6 7 8{a1
    b2=(0 1 2 3{a1),'=',(4{a1),'*',5 6 7 8{a1
    if. ((1=" .b1)+.(1=" .b2))*((9=+/'123456789'e.":b1)+.(9=+/'123456789'e.":b2)) do.
      if. -. (c1=: ".4{.b1)e.quem do.
        quem=:quem,c1
      end.
    end.
  end.
  i=:i+1
end.
+ / quem
)

load 'c:/p/quarta/euler32.ijs'
euler32 ''
45228
```

Capítulo 33

Problema 33 - Frações com cancelamento de dígitos

No original conforme o site projecteuler.net

The fraction $49/98$ is a curious fraction, as an inexperienced mathematician in attempting to simplify it may incorrectly believe that $49/98 = 4/8$, which is correct, is obtained by cancelling the 9s.

We shall consider fractions like, $30/50 = 3/5$, to be trivial examples.

There are exactly four non-trivial examples of this type of fraction, less than one in value, and containing two digits in the numerator and denominator.

If the product of these four fractions is given in its lowest common terms, find the value of the denominator.

Answer: 100

Frações com cancelamento de dígitos A fração $\frac{49}{98}$ é uma fração curiosa, que um matemático inexperiente pode simplificar (erroneamente) fazendo $\frac{49}{98} = \frac{4}{8}$, mas cujo resultado está correto (cancelando os 9's). Consideraremos as frações do tipo $\frac{30}{50} = \frac{3}{5}$ como exemplos triviais. Existem exatamente 4 exemplos não triviais deste tipo de fração, menores do que 1 em valor, contendo dois dígitos no numerador e no denominador. Ache o produto dessas 4 frações, simplificado ao menor valor possível e responda com o valor do denominador.

As frações da resposta são $\frac{16}{64} \rightarrow \frac{1}{4}$, $\frac{19}{95} \rightarrow \frac{1}{5}$, $\frac{26}{65} \rightarrow \frac{2}{5}$ e $\frac{49}{98} \rightarrow \frac{4}{8}$.

33.1 Solução em APL

```
∇euler33;n;d;f;i;nx;dx;rn;rd
[1]  n←10
[2]  rn←rd←10
[3]  t1:→(n>99)/fim
[4]  d←10
[5]  t2:→(d>99)/zica
[6]  →(n>d)/ufa
[7]  f←n÷d
[8]  i←1
[9]  loop:→(i>9)/ufa
[10] nx←i retirar33 n
[11] dx←i retirar33 d
[12] →((dx=0)∨(nx=0))/inca
[13] →((n=nx)^(d=dx))/inca
[14] →(f=nx÷dx)/sim
[15] inca:i←i+1
[16] →loop
[17] ufa:d←d+1
[18] →t2
[19] zica:n←n+1
[20] →t1
[21] fim:
[22] →0
[23] sim:
[24] :If nx≠dx
[25]   rn←rn,nx
[26]   rd←rd,dx
```

```

[27]   :End
[28]   i←i+1
[29]   →loop
[30] fim:(×/rn),×/rd

      ∇r←i retirar33 n;a1;b1
[1]   a1←∅n
[2]   b1←~a1∈∅i
[3]   →(0=+/b1)/edois
[4]   r←±b1/a1
[5]   →0
[6]   edois:r←±1 0/a1

      euler33

8 800

```

33.2 Solução em J

```

retirar33 =: 4 : 0
  a1 =. ":y
  b1 =. -. a1 e. ":x
  if. 0=+/b1 do.
    ".1 0#a1
  else.
    ". b1 # a1
  end.
)
euler33 =: 3 : 0
  n=. 10
  rn=. i.0
  rd=. i.0
  while. n<100 do.
    d=. 10
    while. d<100 do.
      if. n<:d do.
        f=.n%d
        i=. 1
        while. i<:9 do.
          nx=. i retirar33 n
          dx=. i retirar33 d
          if. ((dx=0) +. (nx= 0)) do. goto_inca. end.
          if. ((n=nx)*.(d=dx)) do. goto_inca. end.
          if. f=nx%dx do.
            if. nx ~: dx do.
              rn=.rn,nx
              rd=.rd,dx
            end.
          end.
        end.
      end.
      i=.i+1
    end.
    d=.d+1
  end.
  n=.n+1
end.
linha=. ''
for_i. i.$rn do.
  linha=.linha,("i{i{rn),'/',"i{i{rd),'

```

```

end.
linha
)

load 'c:/p/quarta/euler33.ijs'
euler33 ''
1/4    1/5    2/5    4/8

```

Capítulo 34

Problema 34: Fatoriais dos dígitos

No original conforme o site projecteuler.net

145 is a curious number, as $1! + 4! + 5! = 1 + 24 + 120 = 145$. Find the sum of all numbers which are equal to the sum of the factorial of their digits. Note: as $1! = 1$ and $2! = 2$ are not sums they are not included.

Answer: 40730

Em português, 145 é um número curioso, já que $1! + 4! + 5! = 1 + 24 + 120 = 145$. Ache a soma de todos os números que são iguais à soma dos fatoriais dos seus dígitos. Note que $1! = 1$ e $2! = 2$ não são somas e portanto não estão incluídos.

34.1 Solução em APL

```
FAT34
1 1 2 6 24 120 720 5040 40320 362880

    veuler34;i;s;a1;v
[1] i←3
[2] s←0
[3] t1:→(i>999999)/fim
[4] a1←⌈i
[5] v←⌊((2×pa1)ρ1 0) BARRA DE EXPANSÃO a1
[6] →(i=+/FAT34[v+1])/sim
[7] nao:i←i+1
[8] →t1
[9] sim:
[10] i
[11] s←s+i
[12] i←i+1
[13] →t1
[14] fim:s
```

34.2 Solução em J

```
euler34=: 3 : 0
    FAT34=. 1 1 2 6 24 120 720 5040 40320 362880
    i=.3
    s=.0
    while. i<450000 do.
        a1=."i
        v=.>1".\a1 NB. cada dígito vira elemento de um vetor
        if. i=+/v{FAT34 do.
            s=.s+i
        end.
        i=.i+1
    end.
```

```
s
)

load 'c:/p/quarta/euler34.ijs'
euler34''
40730
```

Capítulo 35

Problema 35 - Primos circulares

No original conforme o site projecteuler.net

The number, 197, is called a circular prime because all rotations of the digits: 197, 971, and 719, are themselves prime.

There are thirteen such primes below 100: 2, 3, 5, 7, 11, 13, 17, 31, 37, 71, 73, 79, and 97.

How many circular primes are there below one million?

Answer: 55

Primos circulares O número 197 é chamado primo circular porque todas as rotações de seus dígitos: 197, 971 e 719 são também primos. Há 13 primos deste tipo abaixo de 100: 2, 3, 5, 7, 11, 13, 17, 31, 37, 71, 73, 79 e 97. Quantos primos circulares há abaixo de 1 milhão ?

35.1 Solução em APL

```

    ∇euler35;p;i;t;b;j;soma
[1]  p←(PRIMO<1000000)/PRIMO
[2]  soma←4
[3]  p←4↓p
[4]  i←1
[5]  t1:→(i>ρp)/fim
[6]  sim:t←ρb←⌊p[i]
[7]  j←1
[8]  t2:→(j≥t)/aca
[9]  →((⌊j⊖b)∈p)/sm
[10] →nao
[11] sm:j←j+1
[12] →t2
[13] aca:soma←soma+1
[14] ⌈ 'p=',⌊p[i]
[15] i←i+1
[16] →t1
[17] nao:
[18] i←i+1
[19] →t1
[20] fim:soma

```

euler35

55

35.2 Solução em J

```

euler35 =: 3 : 0
    i=.4
    s=.4

```

```

limite =. _1 p: 1000000    NB. ha 78498 primos < 1 mi
while. i<: limite do.
  b=.: p: i
  t=. $b
  j=. 1
  while. j<t do.
    if. 0 p: ". j|.b do.
      break.
    end.
    j=.j+1
  end.
  if. j>:t do.
    s=.s+1
  end.
  i=.i+1
end.
s
)

load 'c:/p/quarta/euler35.ijs'
euler35''
55

```

Capítulo 36

Problema 36 - Palíndromos em base dupla

No original conforme o site projecteuler.net

The decimal number, $585 = 10010010012$ (binary), is palindromic in both bases.

Find the sum of all numbers, less than one million, which are palindromic in base 10 and base 2.

(Please note that the palindromic number, in either base, may not include leading zeros.)

Answer: 872187

Palíndromos em duas bases O número decimal, $585 = 1001001001_2$ (em binário) é palindrômico nas duas bases. Ache a soma de todos os números, menores do que um milhão que são palindrômicos na base 10 e na base 2. Note que em um número palindrômico, em qualquer base não se devem incluir os zeros à esquerda.

36.1 Solução em APL

```

    Veuler36;x;som;xa;xb
[1]  x←1
[2]  som←0
[3]  t1:
[4]  →(x≥1000000)/fim
[5]  →(^(xa)=(Θxa←⌜x))/pal d
[6]  x←x+1
[7]  →t1
[8]  pal d:xb←((⌊1+2⊗x)ρ2)⌈x
[9]  →(^(xb=Θxb)/mesmo
[10] x←x+1
[11] →t1
[12] mesmo:som←som+x
[13] x←x+1
[14] →t1
[15] fim:som

    euler36
872187
```

36.2 Solução em J

```

palind=: 3 : 0
    y=".|." : y
)

palindvet=: 3 : 0
    resp=.1
    for_i. i.$y do.
        if. (i{y) ~: (($y)-i+1){y do.
```

```

        resp=.0
        break.
    end.
end.
resp
)

euler36 =: 3 : 0
    s=.0
    for_i. 1+i.999999 do.
        if. palind i do.
            if. palindvet #: i do.
                s=.s+i
            end.
        end.
    end.
    s
)

load 'c:/p/quarta/euler36.ijs'
euler36''
872187

```

Capítulo 37

Problema 37 - Primos truncáveis

No original conforme o site projecteuler.net

The number 3797 has an interesting property. Being prime itself, it is possible to continuously remove digits from left to right, and remain prime at each stage: 3797, 797, 97, and 7. Similarly we can work from right to left: 3797, 379, 37, and 3.

Find the sum of the only eleven primes that are both truncatable from left to right and right to left.

NOTE: 2, 3, 5, and 7 are not considered to be truncatable primes.

Answer: 748317

Primos truncáveis O número 3797 tem uma propriedade interessante. Além dele ser primo, é possível remover continuamente dígitos da esquerda para a direita, e ainda assim ele permanece primo a cada estágio: 3797, 797, 97 e 7. Similarmente pode-se trabalhar da direita para a esquerda: 3797, 379, 37 e 3. Há apenas 11 números primos que gozam desta propriedade. Note que os números 2, 3, 5 e 7 não devem ser considerados primos truncáveis. Ache a soma dos únicos 11 primos que são ambos truncáveis da direita para a esquerda e da esquerda para a direita.

37.1 Solução em APL

```

    veuler37;qt;i;cara;tam;carac;j;som
[1]  qt←0
[2]  som←0
[3]  i←5
[4]  t1:→(qt≥11)/fim
[5]  cara←PRIMO[i]
[6]  tam←pcarac←¯cara
[7]  j←1
[8]  t2:→(j≥tam)/ufa
[9]  →(((¯j↓carac)∈PRIMO)^(¯(¯j)↓carac)∈PRIMO))/sub
[10] i←i+1
[11] →t1
[12] sub:j←j+1
[13] →t2
[14] ufa:
[15] som←som+PRIMO[i]
[16] qt←qt+1
[17] i←i+1
[18] →t1
[19] fim:som

    euler37
748317
```

37.2 Solução em J

```

euler37 =: 3 : 0
  qt=.0
  so=.0
  i=.5
  while. qt < 11 do.
    cara=. p: i
    tam=.$carac=":cara
    j=.1
    while. j < tam do.
      if. (1 p: ". j}.carac) *. 1 p:"(-j)}.carac do.
        j=.j+1
      else.
        break.
      end.
    end.
    if. j>:tam do.
      qt=.qt+1
      so=.so+p: i
    end.
    i=.i+1
  end.
  so
)

```

```

  load 'c:/p/quarta/euler37.ijs'
  euler37''
748317

```

Capítulo 38

Problema 38 - Múltiplos pandigitais

No original conforme o site projecteuler.net

Take the number 192 and multiply it by each of 1, 2, and 3:

$$192 \times 1 = 192$$

$$192 \times 2 = 384$$

$$192 \times 3 = 576$$

By concatenating each product we get the 1 to 9 pandigital, 192384576. We will call 192384576 the concatenated product of 192 and (1,2,3)

The same can be achieved by starting with 9 and multiplying by 1, 2, 3, 4, and 5, giving the pandigital, 918273645, which is the concatenated product of 9 and (1,2,3,4,5).

What is the largest 1 to 9 pandigital 9-digit number that can be formed as the concatenated product of an integer with (1,2, ..., n) where $n > 1$?

Answer: 932718654

Múltiplos pandigitais Pegando o número 192 e multiplicando-o por 1, 2 e 3 :, obtem-se

$$192 \times 1 = 192$$

$$192 \times 2 = 384$$

$$192 \times 3 = 576$$

Pela concatenação de cada produto, obtém-se um número pandigital 1..9, 192384576. Pode-se chamar 192384576 do produto concatenado de 192 por (1, 2, 3).

O mesmo pode ser encontrado começando com 9 e multiplicando-o por 1, 2, 3, 4 e 5.

Qual é o maior número pandigital 1..9 de 9 dígitos que pode ser formado pelo produto concatenado de um inteiro com (1, 2, ..., n) onde $n > 1$?

38.1 Solução em APL

```
▽r←euler38;i;x;y;z
[1] i←1
[2] ufa:z←9 0≠DIG38[i]
[3] x←z[14]
[4] y←z[4+15]
[5] →((x×2)=y)/oba
[6] i←i+1
[7] →(0≠0|i)/ufa
[8] i
[9] →ufa
[10] oba:x,y
▽ 2024-06-18 12.37.19
DIG38[110]
987654321 987654312 987654231 987654213 987654132 987654123 987653421
987653412 987653241 987653214
ρDIG38
```

35900

euler38

9327 18654

38.2 Solução em J

```
euler38 =: 3 : 0
  for_j. i.35899 do.
    zz=.j A. '987654321'
    xx=. " . (i. 4){ zz
    yy=. " . (4+i.5){zz
    if. (2*xx)=yy do.
      xx,yy
      return.
    end.
  end.
end.

)

load 'c:/p/quarta/euler38.ijs'
euler38 ''
9327 18654

)
```

Capítulo 39

Problema 39 - Triângulos retângulos inteiros

no original:

If p is the perimeter of a right angle triangle with integral length sides, $\{a, b, c\}$, there are exactly three solutions for $p = 120$, $\{20, 48, 52\}$, $\{24, 45, 51\}$, $\{30, 40, 50\}$.

For which value of $p \leq 1000$, is the number of solutions maximised?

Traduzindo, Se p é o perímetro de um triângulo retângulo de lados inteiros, $\{a, b, c\}$, há exatamente 3 soluções para $p = 120$ que são $\{20, 48, 52\}$, $\{24, 45, 51\}$, $\{30, 40, 50\}$.

Para quais valores de $p \leq 1000$, é maximizado o número de soluções ?

39.1 Solução em J

Uma observação interessante. Este problema não tem (ainda) solução em APL, talvez porque eu não o tenha entendido muito bem. Mas, eu tinha a resposta e com ela acessei o fórum do problema. Além de uma solução em puro J que copiei

```
P=: ,(+/@, +/&.:*:)@,"0/~ 1+i.1000
P1=: (#~ 1000&>) (#~ (0: = 1&|))P
(I.@(] = >./)@:(+/"1)@= { ~.) P1
840
```

Acabei criando outra a partir de um algoritmo que achei de um usuário chamado euler, em PHP, devidamente traduzido para J

```
euler39=: 3 : 0
    max=.0
    for_p. 1+i.999 do.
        found=.0
        for_a. 1+i. >. p%4 do.
            for_b. (a+1)+ i.>. (p-a)%2 do.
                c=.((a^2)+(b^2))^0.5
                if. p=a+b+c do.
                    found=.found+1
                end.
            end.
        end.
    end.
    if. found>max do.
        max=.found
        value=.p
    end.
end.
max,value
)
```

```
load 'c:/p/quarta/euler39.ijs'
euler39''
7 840
```

Capítulo 40

Problema 40 - Constante de Champernowne

No original conforme o site projecteuler.net

An irrational decimal fraction is created by concatenating the positive integers:

0.123456789101112131415161718192021...

It can be seen that the 12th digit of the fractional part is 1.

If d_n represents the n^{th} digit of the fractional part, find the value of the following expression.

$$d_1 \times d_{10} \times d_{100} \times d_{1000} \times d_{10000} \times d_{100000} \times d_{1000000}$$

Answer: 210 Completed on Sat, 30 Mar 2013, 11:13

Constante do Jeca Tatu Uma fração decimal irracional é criada pela concatenação dos positivos inteiros:

0.123456789101112131415161718192021...

Pode-se ver que o 12º dígito da parte fracionária é 1. Se d_n representa o n-ésimo dígito da parte fracionária, ache o valor da seguinte expressão:

$$d_1 \times d_{10} \times d_{100} \times d_{1000} \times d_{10000} \times d_{100000} \times d_{1000000}$$

40.1 Solução em APL

```
⎕euler40; i; x
[1] x←''
[2] i←1
[3] t1:→(1000000<px)/fim
[4] x←x,⌜i
[5] i←i+1
[6] →t1
[7] fim:x[1],x[10],x[100],x[1000],x[10000],x[100000],x[1000000]
[8] ×/⌞x[1], ' ',x[10], ' ',x[100], ' ',x[1000], ' ',x[10000], ' ',x[100000], ' ',x[1000000]

euler40
1153721
210
```

40.2 Solução em J

```
euler40=: 3 : 0
  xx=: ''
  i=.1
  while. 1000000>$xx do.
    xx=:xx,"i
```

```
        i=.i+1
    end.
NB. (0{xx),(9{xx),(99{xx),(999{xx),(9999{xx),(99999{xx),(999999{xx)
    aa=(0{xx),'',(9{xx}','',(99{xx}','',(999{xx}','')
    aa=.aa,(9999{xx}','',(99999{xx}','',(999999{xx)
    */".aa
)
```

```
    load 'c:/p/quarta/euler40.ijs'
    euler40''
210
```

Capítulo 41

Problema 41 - Primo pandigital

No original conforme o site projecteuler.net

We shall say that an n -digit number is pandigital if it makes use of all the digits 1 to n exactly once. For example, 2143 is a 4-digit pandigital and is also prime.

What is the largest n -digit pandigital prime that exists?

Answer: 7652413

Primos pandigitais Diz-se que um número de n dígitos é pandigital se ele faz uso de todos os dígitos de 1 a n exatamente uma vez. Por exemplo, 2143 é um número pandigital de 4 dígitos e também é primo. Qual é o maior pandigital primo de n -dígitos que existe ?

Vamos estudar o valor de n . Se ele for 9, a soma dos dígitos é $1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 = 45$ que é divisível por 3, logo não é primo. Se ele for $n = 8$ a soma será 36 e novamente ele é divisível por 3, logo também não é primo. Assim, o primeiro candidato é $n = 7$ e a pesquisa vai começar em 7654321 e voltando na busca do primeiro primo pandigital ($n = 7$).

41.1 Solução em APL

```
[5]    x←7654321
[6]    t1:
[7]    →(0≠10000|x)/oba
[8]    x
[9]    oba:
[10]   →(x<1)/fim
[11]   a1←⌈x
[12]   u←~1↑a1
[13]   →((u='1')∨(u='3')∨(u='7'))/continua1
[14]   x←x-1
[15]   →t1
[16]   continua1:
[17]   →(('1'∈a1)^( '2'∈a1)^( '3'∈a1)^( '4'∈a1)^( '5'∈a1)^( '6'∈a1)^( '7'∈a1))/
      continua2
[18]   x←x-1
[19]   →t1
[20]   continua2:
[21]   →(eprimo x)/achei
[22]   x←x-1
[23]   →t1
[24]   achei:x
[25]   →0
[26]   fim:
```

41.2 Solução em J

```
euler41 =: 3 : 0
  xx=.7654321
  while. 1=1 do.
    while. 0=1 p: xx do.
      xx=.xx-1
    end.
    yy=." :xx
    if.('1'e.yy)*.('2'e.yy)*.('3'e.yy)*.('4'e.yy)*.('5'e.yy)*.('6'e.yy)*.('7'e.yy)do.
      xx
      return.
    end.
    xx=.xx-1
  end.
end.
)
```



```
load 'c:/p/quarta/euler41.ijs'
euler41''
7652413
```

Capítulo 42

Problema 42 - Números triangulares codificados

No original conforme o site projecteuler.net

The n^{th} term of the sequence of triangle numbers is given by, $tn = \frac{1}{2}n(n+1)$; so the first ten triangle numbers are:

1, 3, 6, 10, 15, 21, 28, 36, 45, 55, ...

By converting each letter in a word to a number corresponding to its alphabetical position and adding these values we form a word value. For example, the word value for SKY is $19 + 11 + 25 = 55 = t_{10}$. If the word value is a triangle number then we shall call the word a triangle word.

Using words.txt (right click and 'Save Link/Target As...'), a 16K text file containing nearly two-thousand common English words, how many are triangle words?

Answer: 162

Traduzindo, o n^{th} termo de uma sequência de números triangulares é dado por $t_n = \frac{1}{2}n(n+1)$, assim, os 10 primeiros números triangulares são

1, 3, 6, 10, 15, 21, 28, 36, 45, 55, ...

Convertendo cada letra de uma palavra em sua posição alfabética e somando esses valores forma-se o valor da palavra. Por exemplo, o valor da palavra SKY é $19 + 11 + 25 = 55 = t_{10}$. Se o valor da palavra é um número triangular, então pode-se chamar esta palavra de uma palavra triangular.

Usando o arquivo words.txt (busque-o no site do projeto euler) um arquivo de 16K contendo 2000 palavras comuns do inglês, quantas delas são palavras triangulares ?

42.1 Solução em APL

```

    ∇euler42;x;alfa;i;w;so;n;su
[1]  x←HFM8 'c:/p/quarta/0042_words.txt'
[2]  x←,x
[3]  su←0
[4]  x←(' '≠x)/x
[5]  NO42←⌊(⌈x⌉-1)/2⌋
[6]  alfa←'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
[7]  i←1
[8]  t1:→(i>1↑pNO42)/fim
[9]  w←NO42[i;]
[10] w←(w≠' ')/w
[11] so←+/alfa⌈w
[12] n←0.5×~1+(1+8×so)×0.5
[13] →(n=⌊n)/sim
[14] nao:i←i+1
[15] →t1
[16] sim:su←su+1
[17] i←i+1
[18] →t1
[19] fim:su

```

euler42

162

42.2 Solução em J

```
euler42 =: 3 : 0
  xx=. fread 'c:\p\quarta\0042_words.txt'
  xx=.,xx
  i=.0
  som=.0
  alfa=: 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
  aspa=: ''''
  while. i<$xx do.
    if. (i{xx)='"' do.
      xx=.aspa i}xx
    end.
    if. (i{xx)=';' do.
      xx=. ' ' i}xx
    end.
    i=.i+1
  end.
  pal=: ".xx
  tri=: 100$0
  for_i. 1+i.100 do.
    tri=: (0.5*i+i^2) (i-1)}tri
  end.
  i=.0
  while. i<$pal do.
    so=.,+/1+alfa i. >i{pal
    if. so e. tri do.
      som=.som+1
    end.
    i=.i+1
  end.
  som
)
```

```
load 'c:/p/quarta/euler42.ijs'
euler42''
162
```

Capítulo 43

Problema 43 - Divisibilidade de substrings

No original conforme o site projecteuler.net

The number, 1406357289, is a 0 to 9 pandigital number because it is made up of each of the digits 0 to 9 in some order, but it also has a rather interesting sub-string divisibility property.

Let d_1 be the 1st digit, d_2 be the 2nd digit, and so on. In this way, we note the following:

$d_2d_3d_4=406$ is divisible by 2

$d_3d_4d_5=063$ is divisible by 3

$d_4d_5d_6=635$ is divisible by 5

$d_5d_6d_7=357$ is divisible by 7

$d_6d_7d_8=572$ is divisible by 11

$d_7d_8d_9=728$ is divisible by 13

$d_8d_9d_{10}=289$ is divisible by 17

Find the sum of all 0 to 9 pandigital numbers with this property.

Answer: 16695334890

Divisibilidade de substrings O número 1406357289 é um número pandigital de 0 a 9 porque ele é construído com todos os dígitos de 0 a 9, mas ele tem uma propriedade interessante de divisibilidade dos substrings. Seja d_1 o primeiro dígito, d_2 o segundo e assim por diante, pode-se escrever: $d_2d_3d_4 = 406$ é divisível por 2; $d_3d_4d_5 = 063$ é divisível por 3; $d_4d_5d_6 = 635$ é divisível por 5; $d_5d_6d_7 = 357$ é divisível por 7; $d_6d_7d_8 = 572$ é divisível por 11; $d_7d_8d_9 = 728$ é divisível por 13; $d_8d_9d_{10} = 289$ é divisível por 17. Ache a soma de todos os números pandigitais de 0 a 9 com esta propriedade.

43.1 Solução em APL

```
euler43;i;quem;a1;b1;b2;c1
i←3460000000
quem←ι0
t1:
→(0≠10000000|i)/ikj
i,quem
ikj:
→(i>9876543210)/fim
a1←⌊i
→((a1[6]≠'0')^(a1[6]≠'5'))/soma
→(0≠2|⌊a1[4])/soma
→(~'0'∈a1)/soma
→(~'1'∈a1)/soma
→(~'2'∈a1)/soma
→(~'3'∈a1)/soma
→(~'4'∈a1)/soma
→(~'5'∈a1)/soma
→(~'6'∈a1)/soma
→(~'7'∈a1)/soma
→(~'8'∈a1)/soma
→(~'9'∈a1)/soma
→(0≠17|⌊a1[8 9 10])/soma
→(0≠13|⌊a1[7 8 9])/soma
```

```

→(0≠11|a1[6 7 8])/soma
→(0≠7|a1[5 6 7])/soma
→(0≠5|a1[4 5 6])/soma
→(0≠3|a1[3 4 5])/soma
→(0≠2|a1[2 3 4])/soma
quem←quem,i
soma:i←i+1
→t1
fim:quem
A sao 6 os numeros:
A +/1406357289 1430952867 1460357289
A +/ 4106357289 4130952867 4160357289
A resposta 16695334890

```

desempenho MEDONHO

43.2 Solução em J

```

euler43=: 3 : 0 NB. Usuario u56 no forum do problema
r=.10 10 10#:(0{y)*>:i.<.1000%0{y
while. 0<#y=.}.y do.
    r=. ((10*#r)$i.10),.10#r
    r=. (0=({.y)|10#. |:0 1 2{ |:r)#r
end.
+/10x#..((i.10)-."1 r),.r=.([] -: ~.)"1 r)#r
)

load 'c:/p/quarta/euler43.ijs'
euler43 17 13 11 7 5 3 2
16695334890

```

Capítulo 44

Problema 44 - Números pentagonais

No original conforme o site projecteuler.net

Pentagonal numbers are generated by the formula, $P_n = n(3n-1)/2$. The first ten pentagonal numbers are:

1, 5, 12, 22, 35, 51, 70, 92, 117, 145, ...

It can be seen that $P_4 + P_7 = 22 + 70 = 92 = P_8$. However, their difference, $70 - 22 = 48$, is not pentagonal.

Find the pair of pentagonal numbers, P_j and P_k , for which their sum and difference are pentagonal and $D = |P_k - P_j|$ is minimised; what is the value of D ?

Answer: 5482660

Número Pentagonal Números pentagonais são gerados pela fórmula $P_n = n(3n - 1)/2$. Os primeiros dez números pentagonais são:

1, 5, 12, 22, 35, 51, 70, 92, 117, 145, ...

Pode-se ver que $P_4 + P_7 = 22 + 70 = P_8$. Entretanto, a sua diferença $70 - 22 = 48$ não é um número pentagonal. Ache o par de números pentagonais P_j e P_k para os quais sua soma e sua diferença sejam números pentagonais e $D = |P_k - P_j|$ seja minimizada. Qual o valor de D ?

44.1 Solução em APL

```
∇euler44;n;sp;p;j;k;dif;som
[1]  n←6
[2]  sp←((1×(3×1)-1)÷2),((2×(3×2)-1)÷2),((3×(3×3)-1)÷2),((4×(3×4)-1)÷2)
[3]  sp←sp,((5×(3×5)-1)÷2)
[4]  t1:→(0≠100|n)/ufa
[5]  n
[6]  ufa:
[7]  p←((n×(3×n)-1)÷2),(((n+1)×(3×n+1)-1)÷2)
[8]  sp←sp,p
[9]  j←⌊0.5×psp
[10] at2:→(j<1)/fi1
[11]  k←j-1
[12] t3:→(k<1)/fi2
[13]  dif←sp[j]-sp[k]
[14]  som←sp[j]+sp[k]
[15]  →((dif÷sp)^(som÷sp))/achei
[16]  k←k-1
[17]  →t3
[18] fi2:aj←j-1
[19] a →t2
[20] fi1:n←n+2
[21]  →t1
[22] achei:dif,j,k

euler44
5482660 2167 1020
```

44.2 Solução em J

```
euler44=: 3 : 0 NB. Usuario edushka no forum do problema
pn=:[:-:]~3:**:
-/x: pn >: (([,{~) 1: i.~ >&0)([:(+./*(i.&1)){:(-(*.&(e.&(pn i.3000)))})\ pn >: i.3000
)

    load 'c:/p/quarta/euler44.ijs'
    euler44''
5482660
```

Capítulo 45

Problema 45 - Triangular, pentagonal e hexagonal

No original conforme o site projecteuler.net

Triangle, pentagonal, and hexagonal numbers are generated by the following formulae:

Triangle $T_n = n(n+1)/2$ 1, 3, 6, 10, 15, ...

Pentagonal $P_n = n(3n-1)/2$ 1, 5, 12, 22, 35, ...

Hexagonal $H_n = n(2n-1)$ 1, 6, 15, 28, 45, ...

It can be verified that $T_{285} = P_{165} = H_{143} = 40755$.

Find the next triangle number that is also pentagonal and hexagonal.

Answer: 1533776805

Número triangular, pentagonal e hexagonal Números triangulares são gerados por $T_n = n(n+1)/2$. A série é 1,3,6,10,15,... Números pentagonais são gerados por $P_n = n(3n-1)/2$. A série é 1,5,12,22, 35,... . Os hexagonais são gerados por $H_n = n(2n-1)$. A série é 1,6,15, 28,45,... . Pode-se verificar que $T_{285} = P_{165} = H_{143} = 40755$. Pede-se que você encontre o próximo número triangular que é também pentagonal e hexagonal.

45.1 Solução em APL

```
⍝euler45; i; h; p; t
[1] i←144
[2] t1:
[3] h←i×(2×i)-1
[4] p←(1+(1+24×h)×0.5)÷6
[5] →(p≠⌊p)/soma
[6] t←(1+(1+8×h)×0.5)÷2
[7] →(t≠⌊t)/soma
[8] h
[9] →0
[10] soma:i←i+1
[11] →t1

euler45
1533776805
```

45.2 Solução em J

```
euler45=: 3 : 0
  i=.144
  while. 1=1 do.
    h=.i*(2*i)-1
    p=.(1+(1+24*h)^0.5)%6
    t=.(1+(1+8*h)^0.5)%2
```

```
    if. (p=<.p)*.t=<.t do.  
      h  
      return.  
    end.  
    i=.i+1  
  end.  
)  
  
load 'c:/p/quarta/euler45.ijs'  
euler45''  
1533776805
```

Capítulo 46

Problema 46 - Outra conjectura de Goldbach

No original conforme o site projecteuler.net

It was proposed by Christian Goldbach that every odd composite number can be written as the sum of a prime and twice a square.

$9 = 7 + 2 \times 1^2$
 $15 = 7 + 2 \times 2^2$
 $21 = 3 + 2 \times 3^2$
 $25 = 7 + 2 \times 3^2$
 $27 = 19 + 2 \times 2^2$
 $33 = 31 + 2 \times 1^2$

It turns out that the conjecture was false.

What is the smallest odd composite that cannot be written as the sum of a prime and twice a square?

Answer: 5777

Outra conjectura de Goldbach Christian Goldbach propôs que todo número ímpar composto pode ser escrito como a soma de um primo e o dobro de um quadrado.

$9 = 7 + 2 \times 1^2$,
 $15 = 7 + 2 \times 2^2$,
 $21 = 3 + 2 \times 3^2$,
 $25 = 7 + 2 \times 3^2$,
 $27 = 19 + 2 \times 2^2$,
 $33 = 31 + 2 \times 1^2$.

Já se demonstrou que a conjectura é falsa. Abaixo de 100.000 há uma certa quantidade de números ímpares compostos que não podem ser escritos como a soma de um primo e o dobro de um quadrado. Qual é o menor ímpar composto que não pode ser escrito como a soma de um primo e duas vezes um quadrado ?

46.1 Solução em APL

```
∇euler46;n;p;q;j;k
[1] QUA←(ι500)*2
[2] n←9
[3] t1:→(n>pPRIMO)/fim
[4] q←(QUA<n)/QUA
[5] p←(PRIMO<n)/PRIMO
[6] →(n∈PRIMO)/achou
[7] j←pq
[8] t2:→(j<1)/ula
[9] k←pp
[10] t3:→(k<1)/xoc
[11] →(n=(2×q[j])+p[k])/achou
[12] k←k-1
[13] →t3
[14] xoc:
[15] j←j-1
[16] →t2
```

```

[17] achou:n←n+2
[18] →t1
[19] ula:n
[20] →0
[21] fim:'deu xabu'

```

```

euler46
5777

```

46.2 Solução em J

```

euler46=: 3 : 0
    qua=.(1+i.6000)^2
    pri=.p: i.6000
    n=.9
    label_comeco.
    while. n<$pri do.
        if. n e. pri do.
            n=.n+2
            continue.
        end.
        q=.(qua<n)#qua
        p=.(pri<n)#pri
        j=._1+$q
        while. j>:0 do.
            k=._1+$p
            while. k>:0 do.
                if. n=(2*j{q)+k{p do.
                    n=.n+2
                    goto_comeco.
                end.
                k=.k-1
            end.
            j=.j-1
        end.
        n
        return.
    end.
)

load 'c:/p/quarta/euler46.ijs'
euler46''
5777

```

Capítulo 47

Problema 47 - Fatores primos distintos

No original conforme o site projecteuler.net

The first two consecutive numbers to have two distinct prime factors are:

$$14 = 2 \times 7$$

$$15 = 3 \times 5$$

The first three consecutive numbers to have three distinct prime factors are:

$$644 = 2^2 \times 7 \times 23$$

$$645 = 3 \times 5 \times 43$$

$$646 = 2 \times 17 \times 19.$$

Find the first four consecutive integers to have four distinct prime factors. What is the first of these numbers?

Answer: 134043

Fatores primos diferentes Os primeiros dois números consecutivos que tem dois fatores primos diferentes são: $14 = 2 \times 7$ e $15 = 3 \times 5$. Os primeiros três números consecutivos que têm três fatores primos diferentes são: $644 = 2^2 \times 7 \times 23$, $645 = 3 \times 5 \times 43$ e $646 = 2 \times 17 \times 19$. Ache o primeiro conjunto de 4 números inteiros consecutivos que tenham quatro fatores primos diferentes. Qual o primeiro desses números ?

47.1 Solução em APL

```

    veuler47;k;difs
[1]  tb←4 4p0
[2]  tb[1;]←4↑fatprim47 640
[3]  tb[2;]←4↑fatprim47 641
[4]  tb[3;]←4↑fatprim47 642
[5]  tb[4;]←4↑fatprim47 643
[6]  k←644
[7]  t1:→(k>2000000)/fim
[8]  tb[1+4|k;]←4↑fatprim47 k
[9]  difs←,tb
[10] difs←difs[⍴difs]
[11] →((0=+/difs=0)^(0=+/difs=1⊖difs))/achou
[12] k←k+1
[13] →t1
[14] fim:
[15] →0
[16] achou:k
[17]
```

```

    ∇r←fatprim47 c;c1;x;i;r;j;p
[1]  c1←c
[2]  x←(c≥PRIMO)/PRIMO
[3]  r←''
[4]  i←1
```

```

[5]  t1:
[6]  p←1
[7]  t2:→(1=c1)/acabou
[8]  →(0=x[i]|c1)/sim
[9]  →(p≠1)/voltou
[10] i←i+1
[11] →t1
[12] voltou:r←r,p
[13] i←i+1
[14] →t1
[15] sim:
[16] c1←c1÷x[i]
[17] p←p×x[i]
[18] →t2
[19] acabou:
[20] →(p=1)/0
[21] r←r,p

      euler47
134046 -- esqueci de subtrair 3

```

47.2 Solução em J

```

euler47=: 3 : 0
  tb=:4 4 $0
  ini=:644
  tb=: (4{.~.q: (ini+1)) 0}tb
  tb=: (4{.~.q: (ini+2)) 1}tb
  tb=: (4{.~.q: (ini+3)) 2}tb
  tb=: (4{.~.q: (ini+4)) 3}tb
  k=:ini+5
  while. k<140000 do.
    tb=: (4{.~.q: k) (4|k)}tb
    pp1=: 4=$~.(0{tb)
    pp1=: pp1 *. 4=$~.(1{tb)
    pp1=: pp1 *. 4=$~.(2{tb)
    pp1=: pp1 *. 4=$~.(3{tb)
    pp2=: 0=+/0 e. ,tb
    if. pp1 *. pp2 do.
      k-3
      return.
    end.
    k=:k+1
  end.
)

      load 'c:/p/quarta/euler47.ijs'
      euler47''
134043

```

Capítulo 48

Problema 48 - Auto potências

No original conforme o site projecteuler.net

The series, $1^1 + 2^2 + 3^3 + \dots + 10^{10} = 10405071317$.

Find the last ten digits of the series,

$$1^1 + 2^2 + 3^3 + \dots + 1000^{1000}$$

Answer: 9110846700

Potências somadas A série $1^2 + 2^2 + 3^3 + \dots + 10^{10} = 10405071317$.

Ache os últimos 10 dígitos da série

$$1^1 + 2^2 + 3^3 + \dots + 1000^{1000}$$

48.1 Solução em APL

```
∇euler48;i
[1] x←999 12ρ' '
[2] i←1
[3] t1:→(i>999)/fim
[4] x[i;]←i pot48(⌈i)
[5] i←i+1
[6] →(0≠10|i)/t1
[7] i
[8] →t1
[9] fim:
```

```
∇r←a pot48 b;ba
[1] ba←⊖b
[2] a←⌈a
[3] r←'1'
[4] t1:→(ba=0)/fim
[5] r←a mult48 r
[6] ba←ba-1
[7] →t1
[8] fim:
```

```
∇rr←a mult48 b;a1;b1;r;i;vai;temp
[1] a1←3↑'00',a
[2] b1←12↑(12ρ'0'),b
[3] r←4 12ρ' '
[4] i←12
[5] vai←0
[6] t1:→(i<2)/fim1
[7] temp←vai+(⊖a1[3])×⊖b1[i]
[8] →(temp>9)/foi1
```

```

[9]   r[1;i]←1 0⊘temp
[10]  vai←0
[11]  i←i-1
[12]  →t1
[13]  foi1:
[14]  r[1;i]←1 0⊘10|temp
[15]  vai←⌊temp÷10
[16]  i←i-1
[17]  →t1
[18]  fim1:
[19]  i←12
[20]  r[2;12]←'0'
[21]  vai←0
[22]  t2:→(i<2)/fim2
[23]  temp←vai+(⊘a1[2])×⊘b1[i]
[24]  →(temp>9)/foi2
[25]  r[2;i-1]←1 0⊘temp
[26]  vai←0
[27]  i←i-1
[28]  →t2
[29]  foi2:
[30]  r[2;i-1]←1 0⊘10|temp
[31]  vai←⌊temp÷10
[32]  i←i-1
[33]  →t2
[34]  fim2:
[35]  i←12
[36]  r[3;11 12]←'0'
[37]  vai←0
[38]  t3:→(i<3)/fim3
[39]  temp←vai+(⊘a1[1])×⊘b1[i]
[40]  →(temp>9)/foi3
[41]  r[3;i-2]←1 0⊘temp
[42]  vai←0
[43]  i←i-1
[44]  →t3
[45]  foi3:
[46]  r[3;i-2]←1 0⊘10|temp
[47]  vai←⌊temp÷10
[48]  i←i-1
[49]  →t3
[50]  fim3:
[52]  i←12
[53]  vai←0
[54]  t31:→(i<2)/ufa
[55]  temp←vai+(⊘r[1;i])+(⊘r[2;i])+⊘r[3;i]
[56]  →(temp>9)/cri
[57]  r[4;i]←1 0⊘temp
[58]  vai←0
[59]  i←i-1
[60]  →t31
[61]  cri:→(temp>19)/gru
[62]  r[4;i]←1 0⊘temp-10
[63]  vai←1
[64]  i←i-1
[65]  →t31
[66]  gru:
[67]  r[4;i]←1 0⊘temp-20
[68]  vai←2
[69]  i←i-1
[70]  →t31
[71]  ufa:rr←r[4;]
[72]  →0

```

```

[1] x1←((1↑px),-10)↑x
[2] i←1
[3] r←'0000000000'
[4] t1:→(i>999)/aca
[5] r←r soma48 x1[i;]
[6] i←i+1
[7] →t1
[8] aca:r

```

```

      euler48_2 x
9110846700

```

48.2 Solução em J

```

euler48=: 3 : 0
  so=.1x
  for_i. 2+i.999 do.
    el=.i*1x
    ca=.el^el
    so=.so+ca
  end.
  _10{." :so
)

```

```

      load 'c:/p/quarta/euler48.ijs'
      euler48''
9110846700

```

Capítulo 49

Problema 49 - Permutações de primos

No original conforme o site projecteuler.net

The arithmetic sequence, 1487, 4817, 8147, in which each of the terms increases by 3330, is unusual in two ways: (i) each of the three terms are prime, and, (ii) each of the 4-digit numbers are permutations of one another.

There are no arithmetic sequences made up of three 1-, 2-, or 3-digit primes, exhibiting this property, but there is one other 4-digit increasing sequence.

What 12-digit number do you form by concatenating the three terms in this sequence?

Answer: 296962999629

Permutações entre primos A sequência aritmética 1487, 4817, 8147 na qual, cada termo é igual ao anterior mais 3330, é interessante por duas coisas: i. cada um dos 3 termos é primo e, ii. cada um dos números de 4 dígitos é uma permutação de qualquer outro. Não existem sequências feitas de primos de 1, 2 ou 3 dígitos exibindo esta propriedade. Mas há uma outra de números de 4 dígitos, primos com incremento constante. Ache os 3 números de 4 dígitos cada um e responda os doze dígitos do menor para o maior.

49.1 Solução em APL

Uma observação importante é que a programação APL não está funcionando legal. Estou sem saco para procurar o erro e consertá-lo. Até porque este não é o foco aqui. Então, sem mais delongas, vamos para a solução J.

49.2 Solução em J

```
euler49=: 3 : 0
  ii=:_1 p: 10000
  jj=:0
  pr=:i.0
  while. jj<:ii do.
    if. 999< p: jj do.
      pr=:pr,p: jj
    end.
    jj=:jj+1
  end.
  jj=:0
  r=:''
  while. jj<$pr do.
    cara=:jj{pr
    cara1 =: cara+3330
    cara2 =: cara1+3330
    if. (cara1 e.pr )*. cara2 e.pr do.
      c1=:(/:c1){c1=:~.":cara
      c2=:(/:c2){c2=:~.":cara1
      c3=:(/:c3){c3=:~.":cara2
      if. (($c1)=$c2) *. ($c2)=$c3 do.
        if. (($c1)=+/c1=c2) *. ($c2)=+/c2=c3 do.
```

```
        r=:r,' ',(":"cara),(":"cara1),(":"cara2
    end.
end.
end.
jj=:jj+1
end.
r
)
```

```
load 'c:/p/quarta/euler49.ijs'
euler49''
148748178147 296962999629
```

Capítulo 50

Problema 50 - Soma de primos consecutivos

No original conforme o site projecteuler.net

The prime 41, can be written as the sum of six consecutive primes:

$$41 = 2 + 3 + 5 + 7 + 11 + 13$$

This is the longest sum of consecutive primes that adds to a prime below one-hundred.

The longest sum of consecutive primes below one-thousand that adds to a prime, contains 21 terms, and is equal to 953.

Which prime, below one-million, can be written as the sum of the most consecutive primes?

Answer: 997651 Completed on Thu, 11 Apr 2013, 22:53

Soma de primos consecutivos O primo 41, pode ser escrito como a soma de seis primos consecutivos:

$$41 = 2 + 3 + 5 + 7 + 11 + 13$$

Esta é a soma de primos consecutivos mais longa que somados dão um primo abaixo de 100. A maior sequência de primos consecutivos somados dando outro primo abaixo de 1000 é igual a 953 e contém 21 termos. Qual primo, abaixo de N pode ser escrito como a soma da sequência mais longa de primos consecutivos ?

50.1 Solução em APL

```

    Veuler50;x;i;k;som;j;MAI;pmai
[1] x←PRIMO[168+⌈78330] A primos entre 1000 e 1000000
[2] A x←PRIMO[⌈168]
[3] i←px
[4] MAI←-999999
[5] t1:
[6] →(i<1)/fim
[7] k←1
[8] t15:
[9] j←k
[10] som←0
[11] t2:
[12] som←som+PRIMO[j]
[13] →(som=x[i])/chegou
[14] →(som>x[i])/mais
[15] j←j+1
[16] →t2
[17] chegou:
[18] →((1+j-k)<MAI)/denovo
[19] MAI←1+j-k
[20] pmai←x[i]
[21] denovo:i←i-1
[22] →t1
[23] mais:k←k+1
[24] A →(PRIMO[k]>x[i]÷2)/denovo
```

```
[25] →(k>20)/denovo
[26] →t15
[27] fim:MAI,pmai
```

```
euler50
543 997651
```

50.2 Solução em J

Como eu já sei a resposta, dada pelo APL, eis um truque de mágica

```
+/p:3+i.543
997651
```

Usuário olegyk no fórum do problema em J.

```
P=. p: i.&.(-&71274) 78498
t=. (+/\)"1 p:50(+/&i.)546
{:(#~ e.&P)({~([:(i. >./)(+./ * i:&1)@(e.&P)"1)) t
997651
```

50.3 Solução em MatLab

Solução do usuário mahogny no fórum do problema.

```
>> tic
>> p=primes(1e6);
>> pc=cumsum(p);
>> num=length(p);
>> toc
Elapsed time is 0.244699 seconds.
>> tic
>> bestd=0;
>> for i=1:num
    s=0;
    for j=(i+bestd):num
        s=pc(j)-pc(i);
        if s>p(num)
            break;
        end
        m=[];
        m=find(p==s);
        if length(m)==1
            d=j-i;
            if d>bestd
                besti=i;
                bestj=j;
                bestd=d;
                bestm=m;
            end
        end
    end
end
>> toc
Elapsed time is 2.00868 seconds.
>> p(bestm)
ans = 997651
>> end
```

Capítulo 51

Problema 51 - Troca de dígitos em primos

No original conforme o site projecteuler.net

By replacing the 1st digit of the 2-digit number $*3$, it turns out that six of the nine possible values: 13, 23, 43, 53, 73, and 83, are all prime.

By replacing the 3rd and 4th digits of $56**3$ with the same digit, this 5-digit number is the first example having seven primes among the ten generated numbers, yielding the family: 56003, 56113, 56333, 56443, 56663, 56773, and 56993. Consequently 56003, being the first member of this family, is the smallest prime with this property.

Find the smallest prime which, by replacing part of the number (not necessarily adjacent digits) with the same digit, is part of an eight prime value family.

Traduzindo Ao substituir o 1º dígito do número de 2 dígitos $*3$, verifica-se que seis dos nove valores possíveis: 13, 23, 43, 53, 73 e 83 são todos primos.

Ao substituir o 3º e o 4º dígitos de $56**3$ pelo mesmo dígito, esse número de 5 dígitos é o primeiro exemplo com sete primos entre os dez números gerados, resultando na família: 56003, 56113, 56333, 56443, 56663, 56773 e 56993. Consequentemente, 56003, sendo o primeiro membro dessa família, é o menor primo com essa propriedade.

Encontre o menor número primo que, ao substituir parte do número (não necessariamente dígitos adjacentes) pelo mesmo dígito, faça parte de uma família de oito valores primos.

51.1 Solução em Python

Solução de Zach Denton na internet.

```
from itertools import *
def primes(n):
    if n==2: return [2]
    elif n<2: return []
    s=list(range(3,n+1,2))
    mroot = n ** 0.5
    half=(n+1)//2-1
    i=0
    m=3
    while m <= mroot:
        if s[i]:
            j=(m*m-3)//2
            s[j]=0
            while j<half:
                s[j]=0
                j+=m
            i=i+1
            m=2*i+3
    return [2]+[x for x in s if x]

from bisect import bisect_left
# sqrt(1000000000) = 31622
__primes = primes(31622)
def is_prime(n):
    # if prime is already in the list, just pick it
```

```

    if n <= 31622:
        i = bisect_left(__primes, n)
        return i != len(__primes) and __primes[i] == n
    # Divide by each known prime
    limit = int(n ** .5)
    for p in __primes:
        if p > limit: return True
        if n % p == 0: return False
    # fall back on trial division if n > 1 billion
    for f in range(31627, limit, 6): # 31627 is the next prime
        if n % f == 0 or n % (f + 4) == 0:
            return False
    return True

def number_families(num):
    digits = [d for d in str(num)]
    products = list(product((True, False), repeat=len(digits)))[1:-1]
    for p in products:
        pattern = ''
        for i, x in enumerate(p):
            if x:
                pattern += digits[i]
            else:
                pattern += 'X'
        yield [int(pattern.replace('X', str(n))) for n in range(10)]

def main():
    for prime in primes(10000000):
        for number_family in number_families(prime):
            prime_family = [n for n in number_family if is_prime(n) and len(str(n)) == len(str(
            if len(prime_family) == 8 and prime in prime_family:
                print(prime)
                return

if __name__ == "__main__":
    main()

```

121313

Deve-se notar que nesta família $2^3 \cdot 3^3$, os primos são 121313, 222323, 323333, 424343, 525353, 626363, 828383 e 929393. O único que não é primo é 727373.

Capítulo 52

Problema 52 - Múltiplos permutados

No original conforme o site projecteuler.net

It can be seen that the number, 125874, and its double, 251748, contain exactly the same digits, but in a different order.

Find the smallest positive integer, x , such that $2x$, $3x$, $4x$, $5x$, and $6x$, contain the same digits.

Answer: 142857

Múltiplos permutados Pode-se ver que o número 125874 e o dobro dele, 251748 contém exatamente os mesmos dígitos, mas em ordem diferente. Ache o menor positivo inteiro x , tal que $2x$, $3x$, $4x$, $5x$, e $6x$, tenham os mesmos dígitos.

52.1 Solução em APL

```

    ∇euler52;i;i2;i3;i4;i5;i6
[1]   i←1000
[2]   t1:i2←i×2
[3]   →(~i contmesmdig i2)/soma
[4]   i3←i×3
[5]   →(~i contmesmdig i3)/soma
[6]   i4←i×4
[7]   →(~i contmesmdig i4)/soma
[8]   i5←i×5
[9]   →(~i contmesmdig i5)/soma
[10]  i6←i×6
[11]  →(~i contmesmdig i6)/soma
[12]  i
[13]  →0
[14]  soma:i←i+1
[15]  →t1

```

```

    ∇r←a contmesmdig b;a1;b1
[1]   a1←⌊a
[2]   b1←⌊b
[3]   r←(^/a1∈b1)^(^/b1∈a1)

```

```

    euler52
142857

```

52.2 Solução em J

```

contmesmdig=: 4 : 0
    a1=." :x
    b1=." :y
    (*./a1 e. b1) *. *./b1 e. a1
)

```

```
euler52=: 3 : 0
  i=:1000
  label_t1. i2=:i*2
  if. -. i contmesmdig i2 do. goto_soma. end.
  i3=:i*3
  if. -. i contmesmdig i3 do. goto_soma. end.
  i4=:i*4
  if. -. i contmesmdig i4 do. goto_soma. end.
  i5=:i*5
  if. -. i contmesmdig i5 do. goto_soma. end.
  i6=:i*6
  if. -. i contmesmdig i6 do. goto_soma. end.
  i
  return.
  label_soma. i=:i+1
  goto_t1.
)
```

```
    load 'c:/p/quarta/euler52.ijs'
    euler52''
142857
```

Capítulo 53

Problema 53 - Seleções combinatoriais

No original conforme o site projecteuler.net

There are exactly ten ways of selecting three from five, 12345:

123, 124, 125, 134, 135, 145, 234, 235, 245, and 345

In combinatorics, we use the notation, $\binom{5}{3} = 10$

In general, $\binom{n}{r} = \frac{n!}{r!(n-r)!}$ where $r \leq n$, $n! = n \times (n-1) \times \dots \times 3 \times 2 \times 1$, and $0! = 1$

It is not until $n = 23$, that a value exceeds one-million: $\binom{23}{10} = 1144066$.

How many, not necessarily distinct, values of $\binom{n}{r}$ for $1 \leq n \leq 100$, are greater than one-million?

Seleções combinatoriais Há exatamente 10 modos de selecionar 3 objetos de 5: 12345:

123, 124, 125, 134, 135, 145, 234, 235, 245, e 345

Em combinatória, usa-se a notação $\binom{5}{3} = 10$

Em geral, $\binom{n}{r} = \frac{n!}{r!(n-r)!}$ onde $r \leq n$, $n! = n \times (n-1) \times \dots \times 3 \times 2 \times 1$, e $0! = 1$

É só quando $n = 23$, que o valor excede 1 milhão: $\binom{23}{10} = 1144066$.

De quantas maneiras (não necessariamente distintas), que o valor de $\binom{n}{r}$ for $1 \leq n \leq 100$, é maior do que 1 milhão?

53.1 Solução em J

```
euler53=: 3 : 0
  tf=:101$0x
  tf=:1 (0)}tf
  for_i. 1+i.100 do.
    tf=:(((i-1){tf) * i) i}tf
  end.
  ct=:0
  for_i. 1+i.100 do.
    for_r. 1+i.100 do.
      if. (i{tf % (r{tf * (i-r){tf)) > 1000000 do.
        ct=:ct+1
      end.
    end.
  end.
  ct
```

)

```
load 'c:/p/quarta/euler53.ijs'  
euler53''  
4075
```

Capítulo 54

Problema 54 - Mãos de poker

No original conforme o site projecteuler.net

In the card game poker, a hand consists of five cards and are ranked, from lowest to highest, in the following way:

High Card: Highest value card.
One Pair: Two cards of the same value.
Two Pairs: Two different pairs.
Three of a Kind: Three cards of the same value.
Straight: All cards are consecutive values.
Flush: All cards of the same suit.
Full House: Three of a kind and a pair.
Four of a Kind: Four cards of the same value.
Straight Flush: All cards are consecutive values of same suit.
Royal Flush: Ten, Jack, Queen, King, Ace, in same suit.
The cards are valued in the order:
2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King, Ace.

If two players have the same ranked hands then the rank made up of the highest value wins; for example, a pair of eights beats a pair of fives (see example 1 below). But if two ranks tie, for example, both players have a pair of queens, then highest cards in each hand are compared (see example 4 below); if the highest cards tie then the next highest cards are compared, and so on.

Consider the following five hands dealt to two players:

Hand	Player 1	Player 2	Winner
1	5H 5C 6S 7S KD Pair of Fives	2C 3S 8S 8D TD Pair of Eights	Player 2
2	5D 8C 9S JS AC Highest card Ace	2C 5C 7D 8S QH Highest card Queen	Player 1
3	2D 9C AS AH AC Three Aces	3D 6D 7D TD QD Flush with Diamonds	Player 2
4	4D 6S 9H QH QC Pair of Queens Highest card Nine	3D 6D 7H QD QS Pair of Queens Highest card Seven	Player 1
5	2H 2D 4C 4D 4S Full House With Three Fours	3C 3D 3S 9S 9D Full House with Three Threes	Player 1

The file, `poker.txt`, contains one-thousand random hands dealt to two players. Each line of the file contains ten cards (separated by a single space): the first five are Player 1's cards and the last five are Player 2's cards. You can assume that all hands are valid (no invalid characters or repeated cards), each player's hand is in no specific order, and in each hand there is a clear winner.

How many hands does Player 1 win?

376

Mãos de poker No jogo de cartas **pôquer**, uma mão consiste em cinco cartas e elas são classificadas, da menor para a maior, da seguinte maneira:

Carta alta carta de maior valor.

Um par duas cartas do mesmo valor.

Dois pares dois pares diferentes.

Trinca três cartas do mesmo valor.

Straight todas as cartas têm valores consecutivos.

Flush todas as cartas do mesmo naipe.

Full House trinca e um par.

Quadra quatro cartas do mesmo valor.

Straight Flush todas as cartas são valores consecutivos do mesmo naipe.

Royal Flush Dez, Valete, Dama, Rei, Ás do mesmo naipe.

As cartas são avaliadas na seguinte ordem: 2, 3, 4, 5, 6, 7, 8, 9, 10, Valete, Dama, Rei, Ás.

Se dois jogadores tiverem as mesmas mãos classificadas, então a classificação composta pelo valor mais alto vence; por exemplo, um par de oitos vence um par de cincos (veja o exemplo 1 abaixo). Mas se duas classificações empatarem, por exemplo, ambos os jogadores tiverem um par de rainhas, então as cartas mais altas em cada mão são comparadas (veja o exemplo 4 abaixo); se as cartas mais altas empatarem, então as próximas cartas mais altas são comparadas, e assim por diante.

Neste jogo, os naipes recebem os nomes em inglês:

C=clubs=paus=♣

S=spaces=espadas=♠

H=hearts=copas=♥

D=diamonds=ouros=♦

Os naipes são importantes nas análises do flush (5 cartas do mesmo naipe), straight flush (5 cartas do mesmo naipe consecutivas) e royal flush (um straight flush que começa em 10 e acaba em A).

Além disso, o naipe é irrelevante no poker. Ele não desempata nada, quem faz isso é o valor de face da carta em questão. Por exemplo, se um jogador tem uma trinca de K e outro uma trinca de 9, vence a trinca da carta mais alta (o rei). E, o que acontece se 2 jogadores fizerem straight flush (obviamente de naipes diferentes) ? Embora regulamentos locais possam determinar diferentemente, no geral a partida é empatada e as apostas divididas.

Considere as cinco mãos a seguir distribuídas a dois jogadores:

M	Jogador 1	Jogador 2	G
1	5H 5C 6S 7S KD Par de cincos	2C 3S 8S 8D TD par de oitos	2
2	5D 8C 9S JS CA Carta + alta As	2C 5C 7D 8S QH Carta + alta Rainha	1
3	2D 9C AS AH AC Três Ases	3D 6D 7D TD QD Flush de ouros	2
4	4D 6S 9H QH QC Par de Rainhas + Alta: Nove	3D 6D 7H QD QS Par de Rainhas + Alta: Sete	1
5	2H 2D 4C 4D 4S Full House com três quatros	3C 3D 3S 9S 9D Full House com três três	1

54.1 Solução em APL

```

∇r←geraumh95
[1] xx←52 2p' '
[2] xx[;2]←52p'SHCD'
[3] xx[;1]←52p'A23456789TJQK'
[4] xx←xx[52?52;]
[5] r←29↑,xx[10?52;], ' '

∇avalia2j[[]]∇
∇r←avalia2j x;j1;j2;a1;a2;i
[1] j1←convertea2n x[14]
[2] j2←convertea2n x[15+14]
[3] a1←royalflush j1
[4] a2←royalflush j2
[5] :If (a1[1]=1)^(a2[1]≠1)
[6]   r←1
[7]   →0
[8] :EndIf
[9] :If (a1[1]≠1)^(a2[1]=1)
[10]   r←2
[11]   →0
[12] :EndIf
[13] :If (a1[1]=1)^(a2[1]=2)
[14]   r←3
[15]   →0
[16] :EndIf
[17] a1←straightflush j1
[18] a2←straightflush j2
[19] :If (a1[1]=1)^(a2[1]≠1)
[20]   r←1
[21]   →0
[22] :EndIf
[23] :If (a1[1]≠1)^(a2[1]=1)
[24]   r←2
[25]   →0
[26] :EndIf
[27] :If (a1[1]=1)^(a2[1]=1)
[28]   :For i :In 1+i5
[29]     :If a1[i]>a2[i]
[30]       r←1
[31]       →0
[32]     :EndIf
[33]     :If a1[i]<a2[i]
[34]       r←2
[35]       →0
[36]     :EndIf
[37]   :EndFor
[38]   r←3
[39] :EndIf
[40] a1←quadra j1
[41] a2←quadra j2
[42] :If (a1[1]=1)^(a2[1]≠1)
[43]   r←1
[44]   →0
[45] :EndIf
[46] :If (a1[1]≠1)^(a2[1]=1)
[47]   r←2
[48]   →0
[49] :EndIf
[50] :If (a1[1]=1)^(a2[1]=1)
[51]   :If a1[2]>a2[2]
[52]     r←1
[53]   :Else
[54]     r←2
[55]   :EndIf
[56]   →0
[57] :EndIf
[58] a1←fullhs j1
[59] a2←fullhs j2
[60] :If (a1[1]=1)^(a2[1]≠1)
[61]   r←1
[62]   →0
[63] :EndIf
[64] :If (a1[1]≠1)^(a2[1]=1)
[65]   r←2
[66]   →0

```

```

[67] :EndIf
[68] :If (a1[1]=1)^(a2[1]=1)
[69] :If a1[2]>a2[2]
[70] r←1
[71] :Else
[72] r←2
[73] :EndIf
[74] →0
[75] :EndIf
[76] a1←flush j1
[77] a2←flush j2
[78] :If (a1[1]=1)^(a2[1]≠1)
[79] r←1
[80] →0
[81] :EndIf
[82] :If (a1[1]≠1)^(a2[1]=1)
[83] r←2
[84] →0
[85] :EndIf
[86] :If (a1[1]=1)^(a2[1]=1)
[87] :If a1[2]>a2[2]
[88] r←1
[89] →0
[90] :EndIf
[91] :If a1[2]<a2[2]
[92] r←2
[93] →0
[94] :EndIf
[95] r←3
[96] →0
[97] :EndIf
[98] a1←straight j1
[99] a2←straight j2
[100] :If (a1[1]=1)^(a2[1]≠1)
[101] r←1
[102] →0
[103] :EndIf
[104] :If (a1[1]≠1)^(a2[1]=1)
[105] r←2
[106] →0
[107] :EndIf
[108] :If (a1[1]=1)^(a2[1]=1)
[109] :If a1[2]>a2[2]
[110] r←1
[111] →0
[112] :EndIf
[113] :If a1[2]<a2[2]
[114] r←2
[115] →0
[116] :EndIf
[117] r←3
[118] →0
[119] :EndIf
[120] a1←trinca j1
[121] a2←trinca j2
[122] :If (a1[1]=1)^(a2[1]≠1)
[123] r←1
[124] →0
[125] :EndIf
[126] :If (a1[1]≠1)^(a2[1]=1)
[127] r←2
[128] →0
[129] :EndIf
[130] :If (a1[1]=1)^(a2[1]=1)
[131] :If a1[2]>a2[2]
[132] r←1
[133] :Else
[134] r←2
[135] :EndIf
[136] →0
[137] :EndIf
[138] a1←dois pares j1
[139] a2←dois pares j2
[140] :If (a1[1]=1)^(a2[1]≠1)
[141] r←1
[142] →0
[143] :EndIf
[144] :If (a1[1]≠1)^(a2[1]=1)
[145] r←2
[146] →0
[147] :EndIf
[148] :If (a1[1]=1)^(a2[1]=1)
[149] :For i :In 1+13
[150] :If a1[i]>a2[i]
[151] r←1
[152] →0
[153] :EndIf
[154] :If a1[i]<a2[i]
[155] r←2
[156] →0
[157] :EndIf
[158] :EndFor
[159] r←3
[160] →0
[161] :EndIf
[162] a1←um par j1
[163] a2←um par j2
[164] :If (a1[1]=1)^(a2[1]≠1)
[165] r←1
[166] →0
[167] :EndIf
[168] :If (a1[1]≠1)^(a2[1]=1)
[169] r←2
[170] →0
[171] :EndIf
[172] :If (a1[1]=1)^(a2[1]=1)
[173] :For i :In 1+14
[174] :If a1[i]>a2[i]
[175] r←1
[176] →0
[177] :EndIf
[178] :If a1[i]<a2[i]
[179] r←2
[180] →0
[181] :EndIf
[182] :EndFor
[183] r←3
[184] →0
[185] :EndIf
[186] a1←cartaalta j1
[187] a2←cartaalta j2
[188] :For i :In 1+15
[189] :If a1[i]>a2[i]
[190] r←1
[191] →0
[192] :EndIf
[193] :If a1[i]<a2[i]
[194] r←2
[195] →0
[196] :EndIf
[197] :EndFor
[198] r←3
[199] r←royal flush x;z
[200] z←straight flush x
[201] :If (z[1]=1)^z[2]=14
[202] [3] r←1
[203] [4] :Else
[204] [5] r←0
[205] [6] :EndIf

```

54.2 Solução em Javascript

```

<html> <head> <title> apoio a h95 </title>
<meta charset="UTF-8">
<style type="text/css" rel="stylesheet">
h1 color:red;
</style>
<script type="text/javascript">

function convertea2n(a14)
    nai='DSHC'
    car='23456789TJQKA'
    let res=[0,0,0,0,0]
    j=0
    for (i=0;i<14;i=i+3)
        aux=(1+nai.indexOf(a14[i+1]))*100
        res[j]=aux+2+car.indexOf(a14[i])
        j++

```

```

        return res

function straightflush(v)
    li=[0,0,0,0,0]
    na=[0,0,0,0,0]
    for (i=0;i<5;i++)
        li[i]=v[i]%100
        na[i]=Math.floor(v[i]/100)

    li.sort((a, b) => a - b)
    if ((na[0]==na[1])&&(na[1]==na[2])&&
        (na[2]==na[3])&&(na[4]==na[3])&&
        ((li[4]==li[3]+1)&&(li[3]==li[2]+1)&&
        (li[2]==li[1]+1)&&(li[1]==li[0]+1)))
        li.push(1)

```

```

        return (li.reverse())
    else return [0,0,0,0,0,0]

function royalflush(v)
    z=straightflush(v)
    if ((z[0]==1)&&(z[1]==14))
        return [1]
    else return [0]

function quadra(v)
    li=[0,0,0,0,0]
    for (i=0;i<5;i++)
        li[i]=v[i]%100

    li.sort((a, b) => a - b)
    if ((li[0]==li[1])&&(li[1]==li[2])&&
        (li[2]==li[3]))

        nums='1,'+li[0]
        return nums.split(',').map(Number)

    if ((li[1]==li[2])&&(li[2]==li[3])&&
        (li[3]==li[4]))

        nums='1,'+li[1]
        return nums.split(',').map(Number)

    return [0,0]

function fullhs(v)
    li=[0,0,0,0,0]
    for (i=0;i<5;i++)
        li[i]=v[i]%100

    li.sort((a, b) => a - b)
    if ((li[0]==li[1])&&(li[1]==li[2])&&
        (li[3]==li[4]))

        nums='1,'+li[0]
        return nums.split(',').map(Number)

    if ((li[0]==li[1])&&(li[2]==li[3])&&
        (li[3]==li[4]))

        nums='1,'+li[4]
        return nums.split(',').map(Number)

    return [0,0]

function flush(v)
    li=[0,0,0,0,0]
    na=[0,0,0,0,0]
    for (i=0;i<5;i++)
        li[i]=v[i]%100
        na[i]=Math.floor(v[i]/100)

    li.sort((a, b) => a - b)
    if ((na[0]==na[1])&&(na[1]==na[2])&&
        (na[2]==na[3])&&(na[3]==na[4]))

        nums='1,'+li[4]
        return nums.split(',').map(Number)

    else return [0,0]

function straight(v)
    li=[0,0,0,0,0]
    for (i=0;i<5;i++)
        li[i]=v[i]%100

    li.sort((a, b) => a - b)
    if ((li[4]==li[3]+1)&&(li[3]==li[2]+1)&&
        (li[2]==li[1]+1)&&(li[1]==li[0]+1))

        nums='1,'+li[4]
        return nums.split(',').map(Number)

    else return [0,0]

function trinca(v)
    li=[0,0,0,0,0]
    for (i=0;i<5;i++)
        li[i]=v[i]%100

    li.sort((a, b) => a - b)
    if ((li[0]==li[1])&&(li[1]==li[2])&&
        (li[3]!=li[4]))

        nums='1,'+li[2]
        return nums.split(',').map(Number)

    if ((li[1]==li[2])&&(li[2]==li[3])&&
        (li[0]!=li[4]))

        nums='1,'+li[3]
        return nums.split(',').map(Number)

    return [0,0]

    nums='1,'+li[3]
    return nums.split(',').map(Number)

    if ((li[0]!=li[1])&&(li[2]==li[3])&&
        (li[3]!=li[4]))

        nums='1,'+li[4]
        return nums.split(',').map(Number)

    return [0,0]

function dois pares(v)
    li=[0,0,0,0,0]
    for (i=0;i<5;i++)
        li[i]=v[i]%100

    li.sort((a, b) => a - b)
    if ((li[0]==li[1])&&(li[2]==li[3])&&
        (li[0]!=li[2]))

        nums='1,'+li[3]+' '+li[1]+' '+li[4]
        return nums.split(',').map(Number)

    if ((li[1]==li[2])&&(li[3]==li[4])&&
        (li[1]!=li[4]))

        nums='1,'+li[4]+' '+li[2]+' '+li[0]
        return nums.split(',').map(Number)

    if ((li[0]==li[1])&&(li[3]==li[4])&&
        (li[0]!=li[4]))

        nums='1,'+li[4]+' '+li[1]+' '+li[2]
        return nums.split(',').map(Number)

    return [0,0,0,0]

function umpar(v)
    li=[0,0,0,0,0]
    for (i=0;i<5;i++)
        li[i]=v[i]%100

    li.sort((a, b) => a - b)
    if ((li[0]==li[1])&&(li[0]!=li[2])&&
        (li[0]!=li[3])&&(li[0]!=li[4]))

        nums='1,'+li[1]+' '+li[4]+' '+li[3]+' '+li[2]
        return nums.split(',').map(Number)

    if ((li[1]==li[2])&&(li[0]!=li[2])&&
        (li[3]!=li[2])&&(li[4]!=li[2]))

        nums='1,'+li[2]+' '+li[4]+' '+li[3]+' '+li[0]
        return nums.split(',').map(Number)

    if ((li[2]==li[3])&&(li[0]!=li[2])&&
        (li[1]!=li[2])&&(li[4]!=li[2]))

        nums='1,'+li[3]+' '+li[4]+' '+li[1]+' '+li[0]
        return nums.split(',').map(Number)

    if ((li[3]==li[4])&&(li[0]!=li[3])&&
        (li[1]!=li[3])&&(li[2]!=li[3]))

        nums='1,'+li[4]+' '+li[2]+' '+li[1]+' '+li[0]
        return nums.split(',').map(Number)

    return [0,0,0,0,0]

function carta alta(v)
    li=[0,0,0,0,0]
    for (i=0;i<5;i++)
        li[i]=v[i]%100

    li.sort((a, b) => a - b)
    if ((li[0]!=li[1])&&(li[1]!=li[2])&&
        (li[2]!=li[3])&&(li[3]!=li[4]))

        nums='1,'+li[4]+' '+li[3]+' '+li[2]+' '+
            li[1]+' '+li[0]
        return nums.split(',').map(Number)

    return [0,0,0,0,0,0]

function avalia 2j(x)
    j1=converttea2n(x.slice(0,15))
    j2=converttea2n(x.slice(15))
    a1=royalflush(j1)
    a2=royalflush(j2)
    if ((a1[0]==1)&&(a2[0]!=1))return 1
    if ((a1[0]!=1)&&(a2[0]==1))return 2
    if ((a1[0]==1)&&(a2[0]==1))return 3
    a1=straightflush(j1)
    a2=straightflush(j2)

```

```

if ((a1[0]==1)&&(a2[0]!=1))return 1
if ((a1[0]!=1)&&(a2[0]==1))return 2
if ((a1[0]==1)&&(a2[0]==1))
    for (j=1;j<5;j++)
        if (a1[j]>a2[j])return 1
if (a1[j]<a2[j])return 2

return 3

a1=quadra(j1)
a2=quadra(j2)
if ((a1[0]==1)&&(a2[0]!=1))return 1
if ((a1[0]!=1)&&(a2[0]==1))return 2
if ((a1[0]==1)&&(a2[0]==1))
    if (a1[1]>a2[1])return 1
    if (a1[1]<a2[1])return 2

a1=fullhs(j1)
a2=fullhs(j2)
if ((a1[0]==1)&&(a2[0]!=1))return 1
if ((a1[0]!=1)&&(a2[0]==1))return 2
if ((a1[0]==1)&&(a2[0]==1))
    if (a1[1]>a2[1])return 1
    if (a1[1]<a2[1])return 2

a1=flush(j1)
a2=flush(j2)
if ((a1[0]==1)&&(a2[0]!=1))return 1
if ((a1[0]!=1)&&(a2[0]==1))return 2
if ((a1[0]==1)&&(a2[0]==1))
    if (a1[1]>a2[1])return 1
    if (a1[1]<a2[1])return 2
return 3
a1=straight(j1)
a2=straight(j2)
if ((a1[0]==1)&&(a2[0]!=1))return 1
if ((a1[0]!=1)&&(a2[0]==1))return 2
if ((a1[0]==1)&&(a2[0]==1))
    if (a1[1]>a2[1])return 1
    if (a1[1]<a2[1])return 2
return 3
a1=trinca(j1)
a2=trinca(j2)
if ((a1[0]==1)&&(a2[0]!=1))return 1
if ((a1[0]!=1)&&(a2[0]==1))return 2
if ((a1[0]==1)&&(a2[0]==1))
    if (a1[1]>a2[1])return 1
    if (a1[1]<a2[1])return 2

a1=doispares(j1)
a2=doispares(j2)
if ((a1[0]==1)&&(a2[0]!=1))return 1
if ((a1[0]!=1)&&(a2[0]==1))return 2
if ((a1[0]==1)&&(a2[0]==1))
    for (j=1;j<4;j++)
        if (a1[j]>a2[j])return 1
        if (a1[j]<a2[j])return 2

return 3

a1=umpar(j1)
a2=umpar(j2)
if ((a1[0]==1)&&(a2[0]!=1))return 1
if ((a1[0]!=1)&&(a2[0]==1))return 2
if ((a1[0]==1)&&(a2[0]==1))
    for (j=1;j<5;j++)
        if (a1[j]>a2[j])return 1
        if (a1[j]<a2[j])return 2

return 3

a1=cartaalta(j1)
a2=cartaalta(j2)
for (j=1;j<6;j++)
    if (a1[j]>a2[j])return 1
    if (a1[j]<a2[j])return 2

let AA=[
'KS 9C 5D 3H 3S AS AH 2D AC QC',
'JS 9D JH 4D QH 5C JD KS 9S 8H',
...
'8S 4S KS AH 2H 4C 4D 6C JD 5S']

function execu()
ct=0
for (k=0;k<200;k++)
if ((avalia2j(AA[k]))==1)ct++

document.getElementById('resp').value=ct

</script>
</head> <body>
<h1>Apoio à folha h95</h1>
<h2>Leitura de arquivo contendo 200 mãos de poker</h2>
<ul> Procure o arquivo citado na sua folha (pendrive do professor ?)
<li> Descubra quantas vezes a mão 1 ganha
</li>
<h3> Vamos lá </h3>
<form>
Já incluiu as cartas no seu programa fonte ?<br>
<button type="button" id="exe" onclick="execu()">Calcular
</button><br><br>
Resposta às perguntas: <input type="text" id="resp" size="30"
readonly>
</form>
<h4></h4><br>
<br>
<br>
<br>
<br>
<br>
<br>
</body>
</html>

```

54.3 Solução em Python

```

global aa
aa=1
def compute():
    ans = sum(1 for handpair in HANDS if is_player1_win(handpair))
    return str(ans)
def is_player1_win(handpair):
    global aa
    cards = [parse_card(item) for item in handpair.split(" ")]
    assert len(cards) == 10
    player1 = cards[: 5]
    player2 = cards[5 : ]
    aa=aa+1
    return get_score(player1) > get_score(player2)
def get_score(hand):
    assert len(hand) == 5
    rankcounts = [sum(1 for (rank, _) in hand if rank == i) for i in range(13)]
    rankcounthist = [rankcounts.count(i) for i in range(6)]
    minsuit = min(suit for (_, suit) in hand)
    maxsuit = max(suit for (_, suit) in hand)
    flushsuit = minsuit if minsuit == maxsuit else -1
    bestcards = get_5_frequent_highest_cards(rankcounts, rankcounthist)
    straighthighrank = get_straight_high_rank(rankcounts)
    if straighthighrank != -1 and flushsuit != -1 : return 8 << 20 | straighthighrank # Straight flush
    elif rankcounthist[4] == 1 : return 7 << 20 | bestcards # Four of a kind
    elif rankcounthist[3] == 1 and rankcounthist[2] == 1: return 6 << 20 | bestcards # Full house
    elif flushsuit != -1 : return 5 << 20 | bestcards # Flush
    elif straighthighrank != -1 : return 4 << 20 | straighthighrank # Straight
    elif rankcounthist[3] == 1 : return 3 << 20 | bestcards # Three of a kind

```

```

        elif rankcounthist[2] == 2:                : return 2 << 20 | bestcards      # Two pairs
        elif rankcounthist[2] == 1:                : return 1 << 20 | bestcards      # One pair
        else:                                       : return 0 << 20 | bestcards      # High card

def get_5_frequent_highest_cards(ranks, rankshist):
    result = 0
    count = 0
    for i in reversed(range(len(rankshist))):
        for j in reversed(range(len(ranks))):
            if ranks[j] == i:
                for k in range(i):
                    if count >= 5:
                        break
                    result = result << 4 | j
                    count += 1
    if count != 5:
        raise ValueError()
    return result

def get_straight_high_rank(ranks):
    for i in reversed(range(3, len(ranks))):
        for j in range(5):
            if ranks[(i - j + 13) % 13] == 0:
                break
        else:
            return i
    return -1

def parse_card(card):
    return (RANKS.index(card[0]), SUITS.index(card[1]))

RANKS = "23456789TJQKA"
SUITS = "SHCD"

HANDS = [
    "8C TS KC 9H 4S 7D 2S 5D 3S AC",
    "5C AD 5D AC 9C 7C 5H 8D TD KS",
    "3H 7H 6S KC JS QH TD JC 2D 8S",
    "TH 8H 5..."

... aqui entram as cartas de seu arquivo...

print(compute()) # AQUI ELE IMPRIME A QTD DE VITORIAS DO 1

```

Capítulo 55

Problema 55 - Números de Lychrel

No original conforme o site projecteuler.net

If we take 47, reverse and add, $47 + 74 = 121$, which is palindromic.

Not all numbers produce palindromes so quickly. For example, $349 + 943 = 1292$

$1292 + 2921 = 4213$

$4213 + 3124 = 7337$

That is, 349 took three iterations to arrive at a palindrome.

Although no one has proved it yet, it is thought that some numbers, like 196, never produce a palindrome. A number that never forms a palindrome through the reverse and add process is called a Lychrel number. Due to the theoretical nature of these numbers, and for the purpose of this problem, we shall assume that a number is Lychrel until proven otherwise. In addition you are given that for every number below ten-thousand, it will either (i) become a palindrome in less than fifty iterations, or, (ii) no one, with all the computing power that exists, has managed so far to map it to a palindrome. In fact, 10677 is the first number to be shown to require over fifty iterations before producing a palindrome: 4668731596684224866951378664, 53 iterations, 28-digits.

Surprisingly, there are palindromic numbers that are themselves Lychrel numbers; the first example is 4994

How many Lychrel numbers are there below ten-thousand?

NOTE: Wording was modified slightly on 24 April 2007 to emphasise the theoretical nature of Lychrel numbers

Answer: 249

Números de Lychrel Se tomarmos 47, reverter e adicionar, $47 + 74 = 121$, que é palíndromo.

Nem todos os números produzem palíndromos tão rapidamente. Por exemplo, $349 + 943 = 1292$

$1292 + 2921 = 4213$

$4213 + 3124 = 7337$

Foram necessárias três iterações para chegar a um palíndromo.

Embora ninguém o tenha provado ainda, pensa-se que alguns números, como 196, nunca produzem um palíndromo. Um número que nunca forma um palíndromo através do processo de reversão e adição é chamado de número de Lychrel. Devido à natureza teórica desses números, e para o propósito deste problema, assumiremos que um número é Lychrel até que se prove o contrário. Além disso, é dado a você que para cada número abaixo de dez mil, ele (i) se tornará um palíndromo em menos de cinquenta iterações, ou, (ii) ninguém, com todo o poder de computação que existe, conseguiu até agora mapeá-lo para um palíndromo. Na verdade, 10677 é o primeiro número que demonstrou exigir mais de cinquenta iterações antes de produzir um palíndromo: 4668731596684224866951378664, (53 iterações, 28-dígitos).

Surpreendentemente, existem números palíndromos que são eles próprios números de Lychrel; o primeiro exemplo é 4994. Quantos números de Lychrel existem abaixo de dez mil?

NOTA: A redação foi ligeiramente modificada em 24 de abril de 2007 para enfatizar a natureza teórica dos números de Lychrel.

55.1 Solução em APL

```
veuler55;i;ct;wt
[1] i←12
[2] ct←0
[3] t1:→(i>10000)/fim
[4] wt←ccicl55 i A volta o numero de ciclos ate o palindromo ou -1
[5] →(wt>0)/pula
[6] ct←ct+1
```

```

[7] pula:i←i+1
[8]   →t1
[9] fim:ct

    ∇r←ccicl55 i;cic;j;ii
[1]  cic←1
[2]  j←-80↑'0'
[3]  ii←80 0¶i
[4]  t1:→(cic≥100)/neg
[5]  j←0ii
[6]  j←trim j
[7]  ii←trim ii
[8]  ii←ii soma55 j
[9]  A 'ii=',ii,'   j=',j
[10] →(ii iguala550ii)/pos
[11]
[12] cic←cic+1
[13] →t1
[14] neg:r←-1
[15] →0
[16] pos:r←cic

```

```

    ∇r←a soma55 b;rr;aa;bb;i;vaium;temp
[1]  rr←80p' '
[2]  aa←' 0' edit -80↑a
[3]  bb←' 0' edit -80↑b
[4]  i←80
[5]  vaium←0
[6]  t1:→(i<1)/fim
[7]  temp←vaium+(↑aa[i])+↑bb[i]
[8]  →(temp<10)/ok
[9]  temp←temp-10
[10] vaium←1
[11] →junta
[12] ok:vaium←0
[13] junta:rr[i]←1 0¶temp
[14] i←i-1
[15] →t1
[16] fim:r←(+/^\\rr='0')↓rr

```

```

    ∇r←a iguala55 b;aa;bb;igual;i
[1]  aa←' 0' edit -80↑a
[2]  bb←' 0' edit -80↑b
[3]  igual←1
[4]  i←80
[5]  t1:→(i<1)/fim
[6]  →(aa[i]=bb[i])/soma
[7]  igual←0
[8]  i←0
[9]  →t1
[10] soma:
[11] i←i-1
[12] →t1
[13] fim:r←igual

```

euler55

249

Capítulo 56

Problema 56 - Potência dos dígitos somada

No original conforme o site projecteuler.net

A googol (10^{100}) is a massive number: one followed by one-hundred zeros; 100100 is almost unimaginably large: one followed by two-hundred zeros. Despite their size, the sum of the digits in each number is only 1.

Considering natural numbers of the form, ab , where $a, b < 100$, what is the maximum digital sum?

Answer: 972

Soma dos dígitos da potência Um googol (10^{100}) é um número grande: um seguido por 100 zeros. Já 100^{100} é ainda mais formidavelmente maior: um seguido por 200 zeros. Apesar de seus tamanhos, em ambos os casos a soma dos dígitos em cada número é apenas 1. Considere os números naturais na forma a^b , onde a e $b < 100$, e responda qual a máxima soma dos dígitos.

56.1 Solução em APL

```

    ∇euler56;i;j;aa;ct;k;max
[1]  max←-999999
[2]  i←1
[3]  t1:→(i>99)/fim
[4]  i
[5]  j←1
[6]  t2:→(j>99)/incra
[7]  aa←1↓(⌈i)pot56⌈j
[8]  ct←0
[9]  k←1
[10] t3:→(k>paa)/ufa
[11] ct←ct+⌈aa[k]
[12] k←k+1
[13] →t3
[14] ufa:→(ct<max)/volta
[15] max←ct
[16] volta:j←j+1
[17] →t2
[18] incra:i←i+1
[19] →t1
[20] fim:max

    euler56
972
```

56.2 Minha solução em J

```

euler56=: 3 : 0
    max=: -999999
```

```

for_i. 1+i.99x do.
  for_j. 1+i.99x do.
    num=:i^j
    atu=:0
    for_k. i.$num do.
      atu=:atu+"."k}num
    end.
    if. atu>max do.
      max=:atu
    end.
  end.
end.
max
)

load 'c:/p/quarta/euler56.ijs'
euler56''
972

```

Uma solução mandrake no fórum da solução do problema. (Usuário u56)

```

>./,+/|:((#":>./,x)#10)#:"1 0 x=: ^/~60+i.40x
972

```

Capítulo 57

Problema 57 - Convergência da raiz quadrada

No original conforme o site projecteuler.net

It is possible to show that the square root of two can be expressed as an infinite continued fraction.

$$\sqrt{2} = 1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \dots}}}$$

By expanding this for the first four iterations, we get:

$$1 + \frac{1}{2} = \frac{3}{2} = 1.5$$

$$1 + \frac{1}{2 + \frac{1}{2}} = \frac{7}{5} = 1.4$$

$$1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2}}} = \frac{17}{12} = 1.41666\dots$$

$$1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2}}}} = \frac{41}{29} = 1.41379\dots$$

The next three expansions are $\frac{99}{70}$, $\frac{239}{169}$ and $\frac{577}{408}$, but the eighth expansion, $\frac{1393}{985}$, is the first example where the number of digits in the numerator exceeds the number of digits in the denominator.

In the first one-thousand expansions, how many fractions contain a numerator with more digits than the denominator?

Convergência de raiz quadrada É possível mostrar que a raiz quadrada de dois pode ser expressa como uma fração contínua infinita.

$$\sqrt{2} = 1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \dots}}}$$

Expandindo isso para as quatro primeiras iterações, obtemos:

$$1 + \frac{1}{2} = \frac{3}{2} = 1.5$$

$$1 + \frac{1}{2 + \frac{1}{2}} = \frac{7}{5} = 1.4$$

$$1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2}}} = \frac{17}{12} = 1.41666\dots$$

$$1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2}}}} = \frac{41}{29} = 1.41379\dots$$

As próximas três expansões são $\frac{99}{70}$, $\frac{239}{169}$ e $\frac{577}{408}$, mas a oitava expansão, $\frac{1393}{985}$, é o primeiro exemplo em que o número de dígitos no numerador excede o número de dígitos no denominador. Nas primeiras expansões de mil, quantas frações contêm um numerador com mais dígitos que o denominador?

57.1 Solução em Python

```
p = 1
q = 1
counter = 0
for i in range(1000):
    a1 = p + 2*q
    b1 = p + q
    if len(str(a1)) > len(str(b1)):
        counter = counter + 1
    p = a1
    q = b1
```

```
print (counter)
```

```
153
```

57.2 Solução em J

```
euler57=: 3 : 0
  ctd=:0
  p=:1x
  q=:1x
  for_i. i.1000 do.
    a1=:p+2*q
    b1=:p+q
    if. ($":a1)>$":b1 do.
      ctd=:ctd+1
    end.
    p=:a1
    q=:b1
  end.
  ctd
)

load 'c:/p/quarta/euler57.ijs'
euler57''
74 NB. Este resultado foi obtido sem o 'x' na inicialização de p e q
load 'c:/p/quarta/euler57.ijs'
euler57''
153 NB. com 'x' na inicialização
```

A seguir uma solução do usuário olegyk no fórum das respostas

```
f=: >:@%>:
+/->&(#@":)/@(&2&x:)"0 f^:(<1000) 1x
153
```

Capítulo 58

Problema 58 - Primos espirais

No original conforme o site projecteuler.net

Starting with 1 and spiralling anticlockwise in the following way, a square spiral with side length 7 is formed.

```
37 36 35 34 33 32 31
38 17 16 15 14 13 30
39 18 5 4 3 12 29
40 19 6 1 2 11 28
41 20 7 8 9 10 27
42 21 22 23 24 25 26
43 44 45 46 47 48 49
```

It is interesting to note that the odd squares lie along the bottom right diagonal, but what is more interesting is that 8 out of the 13 numbers lying along both diagonals are prime; that is, a ratio of $8/13 \approx 62\%$.

If one complete new layer is wrapped around the spiral above, a square spiral with side length 9 will be formed. If this process is continued, what is the side length of the square spiral for which the ratio of primes along both diagonals first falls below 10%?

Answer: 26241

Primos espirais Começando com 1 e espiralando no sentido anti-horário do jeito abaixo, uma espiral quadrada de lado 7 é formada:

```
37 36 35 34 33 32 31
38 17 16 15 14 13 30
39 18 5 4 3 12 29
40 19 6 1 2 11 28
41 20 7 8 9 10 27
42 21 22 23 24 25 26
43 44 45 46 47 48 49
```

É interessante notar que os quadrados ímpares estão na diagonal inferior direita, mas é ainda mais interessante que 8 dos 13 números das duas diagonais são primos, ou seja, há uma proporção de $8/13 \approx 62\%$. Se enrolarmos uma nova camada nessa espiral, vai-se formar uma espiral quadrada de lado 9. Se este processo continuar, qual o comprimento do lado da espiral quadrada para a qual a relação dos primos nas diagonais é inferior a 10% ?

58.1 Solução em APL

Minha solução em APL é terrivelmente ineficiente, mas deve estar certa porque eu já havia resolvido o problema lá em 2013. De qualquer jeito vai aí:

```
)clear 100M
)copy 1 euler711
euler58
...
v←euler58;x;mat;zz
[1] x←37 36 35 34 33 32 31 38 17 16 15 14 13 30 39 18 5 4 3 12 29 40
[2] x←x,19 6 1 2 11 28 41 20 7 8 9 10 27 42 21 22 23 24 25 26 43 44
[3] x←x,45 46 47 48 49
[4] R←mat←7 7px
```

```

[5] testa:zz←achaprim58 mat
[6]   zz,1↑pmat
[7]   →(zz<0.1)/achou
[8]   mat←maisuma58 mat
[9]   →testa
[10] achou:1↑pmat

      ∇r←achaprim58 x;i;j;ct
[1]   i←1
[2]   ct←10
[3]   t1:→(i>1↑px)/fim
[4]   j←1
[5]   t2:→(j>1↓px)/sofr
[6]   →((i≠j)^((i+j)≠1+1↑px))/soma
[7]   ct←ct,x[i;j]
[8]   soma:j←j+1
[9]   →t2
[10]  sofr:i←i+1
[11]  →t1
[12]  fim:
[13]  r←(+/ct∈PRIMO)÷pct

      ∇r←maisuma58 x;z;w;i;j;k
[1]   z←(,x[1↑px;1↓px])+14+4×1↑px
[2]   w←(2+px)p0
[3]   w[1+1↑px;1+1↓px]←x
[4]   k←1
[5]   i←1+1↑pw
[6]   j←1↓pw
[7]   t2:→(i<1)/ufa1
[8]   w[i;j]←z[k]
[9]   k←k+1
[10]  i←i-1
[11]  →t2
[12]  ufa1:
[13]  i←1
[14]  j←j-1
[15]  t3:→(j<1)/ufa2
[16]  w[i;j]←z[k]
[17]  k←k+1
[18]  j←j-1
[19]  →t3
[20]  ufa2:
[21]  i←2
[22]  j←1
[23]  t4:→(i>1↑pw)/ufa3
[24]  w[i;j]←z[k]
[25]  k←k+1
[26]  i←i+1
[27]  →t4
[28]  ufa3:
[29]  i←1↑pw
[30]  j←2
[31]  t5:→(j>1↓pw)/ufa9
[32]  w[i;j]←z[k]
[33]  k←k+1
[34]  j←j+1
[35]  →t5
[36]  ufa9:r←w

```

Uma versão bem melhor:

```

      ∇euler58_2;d1;d2;d3;d4;tam;nump;tp;pd1;pd2;pd3;pd4;cic;quais;x1;x2;x3;x4
[1]   d1←1 3 13 31

```

```

[2]    x1←x2←x3←x4←0
[3]    d2←5 17 37
[4]    d3←7 21 43
[5]    d4←9 25 49
[6]    tam←7
[7]    nump←8
[8]    tp←13
[9]    t1:→((nump÷tp)<0.1)/fim
[10]   tam,nump,tp,(nump÷tp),x1,x2,x3,x4
[11]   pd1←8+d1[pd1]-d1[-1+pd1]
[12]   d1←d1,x1←d1[pd1]+pd1
[13]   pd2←8+d2[pd2]-d2[-1+pd2]
[14]   d2←d2,x2←d2[pd2]+pd2
[15]   pd3←8+d3[pd3]-d3[-1+pd3]
[16]   d3←d3,x3←d3[pd3]+pd3
[17]   pd4←8+d4[pd4]-d4[-1+pd4]
[18]   d4←d4,x4←d4[pd4]+pd4
[19]   nump←nump+epri mo x1
[20]   nump←nump+epri mo x2
[21]   nump←nump+epri mo x3
[22]   nump←nump+epri mo x4
[23]   tp←tp+4
[24]   tam←tam+2
[25]   →t1
[26]   fim:tam
[27]   D1←d1
[28]   D2←d2
[29]   D3←d3
[30]   D4←d4
[31]

```

58.2 Solução em J

```

euler58=: 3 : 0
  r=.0
  d1=.2
  d2=.4
  d3=.6
  d=.0
  'a b c' =. 1 1 1
  for_i. i.100000 do.
    a=.a+d1+i*8
    b=.b+d2+i*8
    c=.c+d3+i*8
    if. 1 p: a do. r=.r+1 end.
    if. 1 p: b do. r=.r+1 end.
    if. 1 p: c do. r=.r+1 end.
    d=.d+4
    if. 0.1>r%d do.
      ((i*2)+1),r%d
      return.
    end.
  end.
end.
)

load 'c:/p/quarta/euler58.ijs'
euler58''
26241 0.0999924

```

Capítulo 59

Problema 59 - Decriptação XOR

No original conforme o site projecteuler.net

Each character on a computer is assigned a unique code and the preferred standard is ASCII (American Standard Code for Information Interchange). For example, uppercase A = 65, asterisk (*) = 42, and lowercase k = 107.

A modern encryption method is to take a text file, convert the bytes to ASCII, then XOR each byte with a given value, taken from a secret key. The advantage with the XOR function is that using the same encryption key on the cipher text, restores the plain text; for example, 65 XOR 42 = 107, then 107 XOR 42 = 65.

For unbreakable encryption, the key is the same length as the plain text message, and the key is made up of random bytes. The user would keep the encrypted message and the encryption key in different locations, and without both "halves", it is impossible to decrypt the message.

Unfortunately, this method is impractical for most users, so the modified method is to use a password as a key. If the password is shorter than the message, which is likely, the key is repeated cyclically throughout the message. The balance for this method is using a sufficiently long password key for security, but short enough to be memorable.

Your task has been made easy, as the encryption key consists of three lower case characters. Using cipher.txt (right click and 'Save Link/Target As...'), a file containing the encrypted ASCII codes, and the knowledge that the plain text must contain common English words, decrypt the message and find the sum of the ASCII values in the original text.

Answer: 107359

Cujo enunciado é: Cada caracter em um computador tem um código único e o padrão preferido é o ASCII (American Standard Code for Information Interchange). Por exemplo, em maiúsculas, A=65, *=42 e em minúsculas k=107. Um método criptográfico moderno é pegar um arquivo de texto, converter os bytes em ASCII e depois realizar a operação XOR a cada byte com um valor determinado extraído de uma chave secreta. A vantagem da operação XOR é que ao usar a mesma chave no texto cifrado restaura-se o texto original. Por exemplo, 65 XOR 42 = 107 e 107 XOR 42 = 65.

Para uma criptografia segura a chave deve ter o mesmo comprimento da mensagem e deve estar composta de bytes aleatórios. O usuário deve manter a mensagem cifrada e a chave em lugares diferentes. Sem ambas as "metades" é impossível decifrar a mensagem.

Entretanto, este método não é prático para a maioria dos usuários. O método modificado é usar uma contrasenha como chave. Se a contrasenha for mais curta do que a mensagem ela se repete ciclicamente até o fim da mensagem. O equilíbrio do método consiste em usar uma contrasenha suficientemente comprida para ser segura, mas suficientemente curta para ser lembrada facilmente.

Sua tarefa é fácil, já que a chave de criptografia se compõe de 3 caracteres em minúscula. Usando o arquivo anexo, um arquivo com códigos ASCII encriptados, e sabendo que o texto possui palavras comuns em inglês, decifre a mensagem e ache a soma dos valores ASCII do texto original.

59.1 Solução em APL

Para achar a senha, eu usei o programa abaixo, e eu NÃO tenho idéia do que eu fiz

```
[1] A considera-se o texto lido em texto59
[2] A e a chave foi encontrada por tentativa e erro usando
[3] A xor59, trad509 e decif59.
[4] A Com isso chegamos a senha=103,111,100 (cfn)
[5] A dai resp←1 decif59 103 111 100 (incluir manualmente o .)
[6]   i←1
[7]   ct←0
[8]   t1:→(i>1201)/fim
[9]   ct←ct+~1+⌊AViresp[i]
```

```
[10] i←i+1
[11] →t1
[12] fim:ct
```

De qualquer maneira este problema foi modificado em 2019, veja a explicação Observe que o texto cifrado para este problema foi substituído na terça-feira, 5 de fevereiro de 2019. Muitas das postagens permanentes fazem referência ao texto original, mas as discussões ainda permanecem úteis e relevantes.

Quando o texto original foi escolhido, havia três critérios: 1. Usa palavras simples em inglês. 2. Sem restrições de direitos autorais. 3. Contém uma distribuição típica de palavras comuns e frequências de letras. Por esse motivo, o texto original foi um extrato retirado do Novo Testamento e a chave era "deus". Nunca foi a intenção causar ofensa ou fazer com que qualquer membro se sentisse desconfortável. Estamos muito orgulhosos de que o Projeto Euler abrange muitas culturas e é bom lembrar do nosso propósito principal:

O Projeto Euler existe para incentivar, desafiar e desenvolver as habilidades e o prazer de qualquer pessoa interessada no fascinante mundo da matemática.

Embora esse problema tenha permanecido inalterado desde seu início em 2003, não queríamos que membros de religiões diferentes se sentissem desconfortáveis ao serem expostos a um texto de outra religião e, da mesma forma, não gostaríamos que os cristãos se sentissem ofendidos pelo fato de seu texto sagrado estar sendo usado em um contexto menos do que reverente. Consequentemente, a equipe decidiu substituí-lo por algo mais apropriado. Esperamos que você tenha gostado de decifrar e finalmente conseguir ler o novo texto. Para referência, o novo texto decifrado pode ser encontrado abaixo.

An extract taken from the introduction of one of Euler's most celebrated papers, "De summis serierum reciprocarum"[On the sums of series of reciprocals]: I have recently found, quite unexpectedly, an elegant expression for the entire sum of this series $1 + 1/4 + 1/9 + 1/16 + \text{etc.}$, which depends on the quadrature of the circle, so that if the true sum of this series is obtained, from it at once the quadrature of the circle follows. Namely, I have found that the sum of this series is a sixth part of the square of the perimeter of the circle whose diameter is 1; or by putting the sum of this series equal to s , it has the ratio $\sqrt{6}$ multiplied by s to 1 of the perimeter to the diameter. I will soon show that the sum of this series to be approximately 1.644934066842264364; and from multiplying this number by six, and then taking the square root, the number 3.141592653589793238 is indeed produced, which expresses the perimeter of a circle whose diameter is 1. Following again the same steps by which I had arrived at this sum, I have discovered that the sum of the series $1 + 1/16 + 1/81 + 1/256 + 1/625 + \text{etc.}$ also depends on the quadrature of the circle. Namely, the sum of this multiplied by 90 gives the biquadrate (fourth power) of the circumference of the perimeter of a circle whose diameter is 1. And by similar reasoning I have likewise been able to determine the sums of the subsequent series in which the exponents are even numbers.

Capítulo 60

Problema 60 - Conjunto de pares de primos

No original conforme o site projecteuler.net

The primes 3, 7, 109, and 673, are quite remarkable. By taking any two primes and concatenating them in any order the result will always be prime. For example, taking 7 and 109, both 7109 and 1097 are prime. The sum of these four primes, 792, represents the lowest sum for a set of four primes with this property.

Find the lowest sum for a set of five primes for which any two primes concatenate to produce another prime.

Answer: 26033

conjuntos de pares de primos Os primos 3, 7, 109 e 673 são bastante notáveis. Ao tomar quaisquer dois primos e concatená-los em qualquer ordem, o resultado será sempre primo. Por exemplo, tomando 7 e 109 ambos 7109 e 1097 são primos. A soma desses quatro primos, $3 + 7 + 109 + 673 = 792$, representa a menor soma para um conjunto de quatro primos com esta propriedade.

Encontre a menor soma para um conjunto de cinco primos para os quais quaisquer dois primos se concatenam para produzir outro primo.

```

    ∇euler60;x;y;z;x1;y1;z1;i;j;ctd;prox;k;x2;y2;z2;prox1;prox2;l;x3;y3;z3;aa
    ;som;a;b;c;d;e
[1] A procura aneis de cardinalidade 4 ou maior em PRIMCAS
[2] i←1
[3] t1:→(i>pPRIMCAS60)/fim
[4] x←PRIMCAS60[i]
[5] y←[x
[6] z←±(x-[x)×10000
[7] prox←(z=[PRIMCAS60)/PRIMCAS60
[8] j←1
[9] t2:→(j>pprox)/ufa
[10] x1←prox[j]
[11] y1←[x1
[12] z1←±(x1-[x1)×10000
[13] →((y+z1÷10000)∈PRIMCAS60)/tem
[14] naotem:j←j+1
[15] →t2
[16] tem:prox1←(z1=[PRIMCAS60)/PRIMCAS60
[17] k←1
[18] t3:→(k>pprox1)/mesmo
[19] x2←prox1[k]
[20] y2←[x2
[21] z2←±(x2-[x2)×10000
[22] →(((y+z2÷10000)∈PRIMCAS60)^(y1+z2÷10000)∈PRIMCAS60))/boa
[23] arrica:
[24] k←k+1
[25] →t3
[26] mesmo:
[27] →naotem
[28]
[29]
[30]
```

```

[31] boa:
[32] prox2←(z2=[PRIMCAS60])/PRIMCAS60
[33] l←1
[34] t4:→(l>pprox2)/yhu
[35] x3←prox2[l]
[36] y3←x3
[37] z3←⊕(x3-[x3])×10000
[38] →(((y+z2÷10000)∈PRIMCAS60)^(y1+z2÷10000)∈PRIMCAS60)^(y2+z3÷10000)∈
    PRIMCAS60))/boazuda
[39] hifa:
[40] l←l+1
[41] →t4
[42] yhu:
[43] →arrica
[44] boazuda:
[45]
[46] a i,j,PRIMCAS[i],prox[j],(y+z1÷10000),(y+z2÷10000),(y1+z2÷10000),(y2+z3÷100
    00)
[47] aa←PRIMO[[PRIMCAS60[i]],PRIMO[[prox[j]],PRIMO[z1],PRIMO[z2],PRIMO[z3]
[48] a←⊗aa[1]
[49] b←⊗aa[2]
[50] c←⊗aa[3]
[51] d←⊗aa[4]
[52] e←⊗aa[5]
[53] som←0
[54] som←som+eprimo⊕a,b
[55] som←som+eprimo⊕a,c
[56] som←som+eprimo⊕a,d
[57] som←som+eprimo⊕a,e
[58] som←som+eprimo⊕b,a
[59] som←som+eprimo⊕c,a
[60] som←som+eprimo⊕d,a
[61] som←som+eprimo⊕e,a
[62]
[63] som←som+eprimo⊕b,c
[64] som←som+eprimo⊕b,d
[65] som←som+eprimo⊕b,e
[66] som←som+eprimo⊕c,b
[67] som←som+eprimo⊕d,b
[68] som←som+eprimo⊕e,b
[69]
[70] som←som+eprimo⊕c,d
[71] som←som+eprimo⊕c,e
[72] som←som+eprimo⊕d,c
[73] som←som+eprimo⊕e,c
[74]
[75] som←som+eprimo⊕d,e
[76] som←som+eprimo⊕e,d
[77]
[78] →(som≥20)/esim22
[79] →pula77
[80] esim22:
[81] (+/aa),aa
[82] pula77:
[83] →arrica
[84] →naotem
[85] ufa:i←i+1
[86] →t1
[87] fim:

    euler60
26033 13 5197 5701 6733 8389

```

60.1 Uma solução em Python

Do usuário tolstopuz do fórum do problema

```
mx = 10000

primes = [2]

def isprime(p):
    for i in primes:
        if i * i > p:
            return True
        if p % i == 0:
            return False
    return True

x = 3
while x < mx:
    while not isprime(x):
        x = x + 2
    primes = primes + [x]
    x = x + 2

ml = 10 ** 20

pairs = {a:{b for b in primes if a < b and isprime(int(str(a)+str(b))) and
            isprime(int(str(b)+str(a)))} for a in primes}

for a in primes:
    xa = pairs[a]
    for b in xa:
        xb = xa & pairs[b]
        for c in xb:
            xc = xb & pairs[c]
            for d in xc:
                xd = xc & pairs[d]
                for e in xd:
                    l = a + b + c + d + e
                    if l < ml:
                        ml = l

print(ml)
print(isprime(101))
print(isprime(100))
```

Capítulo 61

Problema 61 - Números figurativos cíclicos

No original conforme o site projecteuler.net

Triangle, square, pentagonal, hexagonal, heptagonal, and octagonal numbers are all figurate (polygonal) numbers and are generated by the following formulae:

Triangle	$P_{3,n} = n(n+1)/2$	1, 3, 6, 10, 15, ...
Square	$P_{4,n} = n^2$	1, 4, 9, 16, 25, ...
Pentagonal	$P_{5,n} = n(3n-1)/2$	1, 5, 12, 22, 35, ...
Hexagonal	$P_{6,n} = n(2n-1)$	1, 6, 15, 28, 45, ...
Heptagonal	$P_{7,n} = n(5n-3)/2$	1, 7, 18, 34, 55, ...
Octagonal	$P_{8,n} = n(3n-2)$	1, 8, 21, 40, 65, ...

The ordered set of three 4-digit numbers: 8128, 2882, 8281, has three interesting properties.

The set is cyclic, in that the last two digits of each number is the first two digits of the next number (including the last number with the first). Each polygonal type: triangle ($P_{3,127}=8128$), square ($P_{4,91}=8281$), and pentagonal ($P_{5,44}=2882$), is represented by a different number in the set. This is the only set of 4-digit numbers with this property. Find the sum of the only ordered set of six cyclic 4-digit numbers for which each polygonal type: triangle, square, pentagonal, hexagonal, heptagonal, and octagonal, is represented by a different number in the set.

Answer: 28684

Números especiais cíclicos Números triangulares, quadrados, pentagonais, hexagonais, heptagonais e octogonais, são números poligonais e cada um é gerado pelas fórmulas:

Triangular:	$P_{3,n} = n(n+1)/2,$	1, 3, 6, 10, 15, ...
Quadrado:	$P_{4,n} = n^2,$	1, 4, 9, 16, 25, ...
Pentagonal:	$P_{5,n} = n(3n-1)/2,$	1, 5, 12, 22, 35, ...
Hexagonal:	$P_{6,n} = n(2n-1),$	1, 6, 15, 28, 45, ...
Heptagonal:	$P_{7,n} = n(5n-3)/2,$	1, 7, 18, 34, 55, ...
Octagonal:	$P_{8,n} = n(3n-2),$	1, 8, 21, 40, 65, ...

O conjunto ordenado de números de 4 dígitos 8128, 2882, 8281 tem 3 interessantes propriedades: i. O conjunto é cíclico já que os últimos 2 dígitos de um número são os 2 primeiros do próximo (incluindo o último e o primeiro); ii. Cada tipo polinomial: triângulo ($P_{3,127} = 8128$), quadrado ($P_{4,91} = 8281$) e pentagonal ($P_{5,44} = 2882$) é representado por um número diferente no conjunto. iii. Este é o único conjunto de números de 4 dígitos com esta propriedade. Ache a soma do único conjunto ordenado de 6 números cíclicos de 4 dígitos, um para cada tipo polinomial e onde cada um dos tipos é representado por um número diferente no conjunto.

61.1 Solução em APL

Bom, eu resolvi o problema, mas estou com dificuldade de recuperar a solução por causa da modificada posterior que fiz para o gerador de exercícios diferentes (série VIVOE, eu acho). Já dei tratos à bola com os programas em EULER711, mas não consegui achar uma resposta. Vou por a resposta aqui, mas com muita dificuldade de entendimento.

```
∇euler61;x;n;octa;hepta;hexa;penta;quad;tria
[1] x←19
[2] octa←10
[3]
[4] t1:
[5] n←x×(3×x)-2
```

```

[6]   →(n>9999)/fim
[7]   octa←octa,n
[8]   x←x+1
[9]   →t1
[10] fim:
[11]   x←1
[12]   hepta←ι0
[13] t2:
[14]   n←(x×(5×x)-3)÷2
[15]   →(n>9999)/fim2
[16]   →(n<1000)/soma1
[17]   hepta←hepta,n
[18] soma1:x←x+1
[19]   →t2
[20] fim2:
[21]   hexa←ι0
[22]   x←1
[23] t3:
[24]   n←x×(2×x)-1
[25]   →(n>9999)/fim3
[26]   →(n<1000)/soma2
[27]   hexa←hexa,n
[28] soma2:x←x+1
[29]   →t3
[30] fim3:
[31]   penta←ι0
[32]   x←1
[33] t4:
[34]   n←(x×(3×x)-1)÷2
[35]   →(n>9999)/fim4
[36]   →(n<1000)/soma3
[37]   penta←penta,n
[38] soma3:
[39]   x←x+1
[40]   →t4
[41] fim4:
[42]   quad←ι0
[43]   x←1
[44] t5:
[45]   n←x×2
[46]   →(n>9999)/fim5
[47]   →(n<1000)/soma4
[48]   quad←quad,n
[49] soma4:
[50]   x←x+1
[51]   →t5
[52] fim5:
[53]   tria←ι0
[54]   x←1
[55] t6:
[56]   n←(x×x+1)÷2
[57]   →(n>9999)/fim6
[58]   →(n<1000)/soma5
[59]   tria←tria,n
[60] soma5:
[61]   x←x+1
[62]   →t6
[63] fim6:
[64]   TRIA←4 0⌘((ptria),1)ptria
[65]   QUAD←4 0⌘((pquad),1)pquad
[66]   PENTA←4 0⌘((ppenta),1)ppenta
[67]   HEXA←4 0⌘((phexa),1)phexa
[68]   HEPTA←4 0⌘((phepta),1)phepta

```

```
[69] OCTA←4 0ᄡ((pocᄡa),1)pocᄡa
[70]
```

```

ᄡr←euler61_2 ind;f;v5;v4;i;j;v6;v3;v2;t1;t2;t3;t4;t5;t6;t7;v1;var
[1] r←0 40ᄡ' '
[2] var←6 5ᄡ'TRIA QUAD PENTAHEXA HEPTAOCTA '
[3] ᄡ 123456789012345678901234567890
[4] ᄡ'v1←',var[ind[1];]
[5] ᄡ'v2←',var[ind[2];]
[6] ᄡ'v3←',var[ind[3];]
[7] ᄡ'v4←',var[ind[4];]
[8] ᄡ'v5←',var[ind[5];]
[9] ᄡ'v6←',var[ind[6];]
[10] inicio:
[11] t1←0 8ᄡ' '
[12] i←1
[13] m1:→(i>1ᄡpv6)/aca1
[14] f←ᄡv6[i;3 4]
[15] j←1
[16] w1:→(j>1ᄡpv1)/ufa1
[17] →(f≠ᄡv1[j;1 2])/ss1
[18] t1←t1,[1]1 8ᄡv6[i;],v1[j;]
[19] ss1:j←j+1
[20] →w1
[21] ufa1:
[22] i←i+1
[23] →m1
[24] aca1:
[25] ᄡ 't1='
[26] ᄡ t1
[27] i←1
[28] t2←0 12ᄡ' '
[29] m2:→(i>1ᄡpt1)/aca2
[30] f←ᄡt1[i;7 8]
[31] j←1
[32] w2:→(j>1ᄡpv2)/ufa2
[33] →(f≠ᄡv2[j;1 2])/ss2
[34] t2←t2,[1]1 12ᄡt1[i;],v2[j;]
[35] ss2:j←j+1
[36] →w2
[37] ufa2:i←i+1
[38] →m2
[39] aca2:
[40] ᄡ 't2='
[41] ᄡ t2
[42] t3←0 16ᄡ0
[43] i←1
[44] m3:→(i>1ᄡpt2)/aca3
[45] f←ᄡt2[i;11 12]
[46] j←1
[47] w3:→(j>1ᄡpv3)/ufa3
[48] →(f≠ᄡv3[j;1 2])/ss3
[49] t3←t3,[1]1 16ᄡt2[i;],v3[j;]
[50] ss3:j←j+1
[51] →w3
[52] ufa3:i←i+1
[53] →m3
[54] aca3:
[55] ᄡ 't3='
[56] ᄡ t3
[57] t4←0 20ᄡ0
[58] i←1
[59] m4:→(i>1ᄡpt3)/aca4
```

```

[60] f←t3[i;15 16]
[61] j←1
[62] w4:→(j>1tpv4)/ufa4
[63] →(f≠v4[j;1 2])/ss4
[64] t4←t4,[1]1 20pt3[i;],v4[j;]
[65] ss4:j←j+1
[66] →w4
[67] ufa4:i←i+1
[68] →m4
[69] aca4:
[70] A 't4='
[71] A t4
[72] t5←0 24p0
[73] i←1
[74] m5:→(i>1tp4)/aca5
[75] f←t4[i;19 20]
[76] j←1
[77] w5:→(j>1tpv5)/ufa5
[78] →(f≠v5[j;1 2])/ss5
[79] t5←t5,[1]1 24pt4[i;],v5[j;]
[80] ss5:j←j+1
[81] →w5
[82] ufa5:i←i+1
[83] →m5
[84] aca5:
[85] A 't5='
[86] A t5
[87] i←1
[88] t6←0 28p0
[89] m6:→(i>1tp5)/aca6
[90] f←t5[i;23 24]
[91] j←1
[92] w6:→(j>1tpv6)/ufa6
[93] →(f≠v6[j;1 2])/ss6
[94] t6←t6,[1]1 28pt5[i;],v6[j;]
[95] ss6:j←j+1
[96] →w6
[97] ufa6:i←i+1
[98] →m6
[99] aca6:
[100]A 't6='
[101]A t6
[102] i←1
[103]t88:→(i>1tp6)/fik
[104] →((t6[i;1 2 3 4])≠t6[i;25 26 27 28])/sok
[105] r←r,[1]1 40pt6[i;],2 0ind
[106]sok:i←i+1
[107] →t88
[108]fik:
[109]

```

61.2 Solução em Python

Do usuário the_gav no fórum de soluções do problema.

```
numbers=set(range(1000,10000))
```

```

triangles=[int(n*(n+1)/2) for n in range(300) if n*(n+1)/2 in numbers]
squares=[int(n**2) for n in range(300) if n**2 in numbers]
pentagons=[int(n*(3*n-1)/2) for n in range(300) if n*(3*n-1)/2 in numbers]
hexagons=[int(n*(2*n-1)) for n in range(300) if n*(2*n-1) in numbers]
heptagons=[int(n*(5*n-3)/2) for n in range(300) if n*(5*n-3)/2 in numbers]

```

```

octagons= [int(n*(3*n-2)) for n in range(300) if n*(3*n-2) in numbers]

all_nums=list(set(triangles+squares+pentagons+hexagons+heptagons+octagons))

nums_dict={'triangles':triangles, 'squares':squares, 'pentagons':pentagons,
'hexagons':hexagons, 'heptagons':heptagons,'octagons':octagons}
reverse_lookup={}
for key, numbers in nums_dict.items():
    for number in numbers:
        reverse_lookup[number] = key

pairs=[[a,b] for a in all_nums for b in all_nums if str(a)[2:] == str(b)[2:]
and reverse_lookup[a] != reverse_lookup[b]]

trios=[[a,b,c] for a,b in pairs for c in all_nums if str(b)[2:] == str(c)[2:]
and reverse_lookup[c] != reverse_lookup[a] and reverse_lookup[c] != reverse_lookup[b]]

quartets=[[a,b,c,d] for a,b,c in trios for d in all_nums if str(c)[2:] ==
str(d)[2:] and reverse_lookup[d] != reverse_lookup[c] and
reverse_lookup[d] != reverse_lookup[b] and reverse_lookup[d] != reverse_lookup[a]]

quintets=[[a,b,c,d,e] for a,b,c,d in quartets for e in all_nums
if str(d)[2:] == str(e)[2:] and reverse_lookup[e] != reverse_lookup[d] and
reverse_lookup[e] != reverse_lookup[c] and reverse_lookup[e] != reverse_lookup[b]
and reverse_lookup[e] != reverse_lookup[a]]

sextets= [[a,b,c,d,e,f] for a,b,c,d,e in quintets for f in all_nums if str(e)[2:] ==
str(f)[2:] and str(f)[2:] == str(a)[2:] and reverse_lookup[f] != reverse_lookup[e]
and reverse_lookup[f] != reverse_lookup[d] and reverse_lookup[f] != reverse_lookup[c]
and reverse_lookup[f] != reverse_lookup[b] and reverse_lookup[f] != reverse_lookup[a]]

print(sextets)
print([[reverse_lookup[x] for x in i] for i in sextets])
print([sum(i) for i in sextets])

[[8256, 5625, 2512, 1281, 8128, 2882], [1281, 8128, 2882, 8256, 5625, 2512], [2882, 8256, 5625, 2512, 1281, 8128], [8128, 2882,
8256, 5625, 2512, 1281], [2512, 1281, 8128, 2882, 8256, 5625], [5625, 2512, 1281, 8128, 2882, 8256]] [['triangles', 'squares',
'heptagons', 'octagons', 'hexagons', 'pentagons'], ['octagons', 'hexagons', 'pentagons', 'triangles', 'squares', 'heptagons'],
['pentagons', 'triangles', 'squares', 'heptagons', 'octagons', 'hexagons'], ['hexagons', 'pentagons', 'triangles', 'squares',
'heptagons', 'octagons'], ['heptagons', 'octagons', 'hexagons', 'pentagons', 'triangles', 'squares'], ['squares', 'heptagons',
'octagons', 'hexagons', 'pentagons', 'triangles']] [28684, 28684, 28684, 28684, 28684, 28684]

```

Capítulo 62

Problema 62 - Permutações cúbicas

No original conforme o site projecteuler.net

The cube, 41063625 (345^3), can be permuted to produce two other cubes: 56623104 (384^3) and 66430125 (405^3). In fact, 41063625 is the smallest cube which has exactly three permutations of its digits which are also cube.

Find the smallest cube for which exactly five permutations of its digits are cube.

Answer: 127035954683

Permutações cúbicas O cubo, 41063625 (345^3), pode ser permutado para produzir dois outros cubos: 56623104 (384^3) e 66430125 (405^3). Na verdade, 41063625 é o menor cubo que tem exatamente três permutações de seus dígitos que também são cubo.

Encontre o menor cubo para o qual exatamente cinco permutações de seus dígitos são cúbicas.

62.1 Solução em APL

```
[1] x←1
[2] C62←9999ρ0
[3] t1:→(x>9999)/fim
[4] C62[x]←x*3
[5] x←x+1
[6] →t1
[7] fim:
[8] T62←9999ρ0
[9] i←1
[10] t2:→(i>9999)/fi3
[11] x←⌊C62[i]
[12] a←(1 0⌈+/'0'=x),(1 0⌈+/'1'=x),(1 0⌈+/'2'=x),(1 0⌈+/'3'=x),(1 0⌈+/'4'=x)
[13] a←a,(1 0⌈+/'5'=x),(1 0⌈+/'6'=x),(1 0⌈+/'7'=x),(1 0⌈+/'8'=x),(1 0⌈+/'9'=x)
[14] T62[i]←⌊a
[15] i←i+1
[16] →t2
[17] fi3:i←1
[18] t3:→(i>9999)/fi7
[19] →(5>+/b←T62∈T62[i])/pula
[20] i,(b/⌊9999)
[21] pula:i←i+1
[22] →t3
[23] fi7:A a resposta |e 5027 * 3 = 127035954683
[24]
```

euler62

5027 5027 7061 7202 8288 8384 ...

62.2 Solução em J

Uma solução elegante do usuário olegyk no fórum de soluções

```
({~ (i.>./)@:({."1)) (/::~@":"0 (#,<./)/. ])3^~i.8600x
5 127035954683
```

Capítulo 63

Problema 63 - Contagens de dígitos poderosas

No original conforme o site projecteuler.net

The 5-digit number, $16807 = 7^5$, is also a fifth power. Similarly, the 9-digit number, $134217728 = 8^9$, is a ninth power.

How many n -digit positive integers exist which are also an n th power?

Answer: 49

Potências de dígitos O número de 5 dígitos $16807 = 7^5$ é também uma potência quinta. Similarmente o número de 9 dígitos $134217728 = 8^9$ é uma potência nona. Há 49 inteiros positivos que, sendo números de n -dígitos são também potências n -ésimas.

63.1 Solução em APL

```
⎕euler63;x;y;z;k;w;ct
[1] ct←1
[2] x←1
[3] t1:→(x>1000)/fim
[4] y←1
[5] t2:
[6] z←40 0⌞k←x*y
[7] w←(z≠' ')/z
[8] →(y≠pw)/volta
[9] A
[10]
[11] ct,x,y,w,pw
[12] ct←ct+1
[13]
[14] y←y+1
[15] →t2
[16] volta:
[17] x←x+1
[18] →t1
[19]
[20] fim:

    euler63
...
49 9 21 109418989131512000000 21
```

63.2 Solução em J

Elegante solução do usuário olegyk no fórum de soluções

```
#~. (#~&, (#@": "0 ="1 0 >:@i.@#)) | : ^/~1+i.24x
49
```

63.3 Solução em Python

solução do usuário hamsterboy do fórum de soluções

```
for maxpow in xrange(5,100):
    if len(str(9**maxpow)) > maxpow:
        break
nums = []
for n in range(1,maxpow):
    for i in range(1,10):
        x = i**n
        if len(str(x)) == n:
            nums.append((i,n))
print len(nums)

> 49
```

Capítulo 64

Problema 64 - Período ímpar de raízes quadradas

Não vou por o original aqui, pela imensa dificuldade de transcrever as expressões matemáticas em formato Latex. Veja lá no site se quiser o original em inglês.

Toda raiz quadrada é periódica quando escrita como frações contínuas e pode ser escrita na forma

$$\sqrt{N} = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \dots}}}$$

Por exemplo, vai-se considerar $\sqrt{23}$:

$$\sqrt{23} = 4 + \sqrt{23} - 4 = 4 + \frac{1}{\frac{1}{\sqrt{23}-4}} = 4 + \frac{1}{1 + \frac{\sqrt{23}-3}{7}}$$

Se continuar com a expansão vai se obter:

$$\sqrt{23} = 4 + \frac{1}{1 + \frac{1}{3 + \frac{1}{1 + \frac{1}{3 + \dots}}}}$$

o processo pode ser sumarizado como:

$$\begin{aligned} a_0 &= 4, \frac{1}{\sqrt{23}-4} = \frac{\sqrt{23}+4}{7} = 1 + \frac{\sqrt{23}-3}{7} \\ a_1 &= 4, \frac{7}{\sqrt{23}-3} = \frac{7(\sqrt{23}+3)}{14} = 3 + \frac{\sqrt{23}-3}{2} \\ a_2 &= 3, \frac{2}{\sqrt{23}-3} = \frac{2(\sqrt{23}+3)}{14} = 1 + \frac{\sqrt{23}-4}{7} \\ a_3 &= 1, \frac{7}{\sqrt{23}-4} = \frac{7(\sqrt{23}+4)}{7} = 8 + \sqrt{23} - 4 \\ a_4 &= 8, \frac{1}{\sqrt{23}-4} = \frac{\sqrt{23}+4}{7} = 1 + \frac{\sqrt{23}-3}{7} \\ a_5 &= 1, \frac{7}{\sqrt{23}-3} = \frac{7(\sqrt{23}+3)}{14} = 3 + \frac{\sqrt{23}-3}{2} \\ a_6 &= 3, \frac{2}{\sqrt{23}-3} = \frac{2(\sqrt{23}+3)}{14} = 1 + \frac{\sqrt{23}-4}{7} \\ a_7 &= 1, \frac{7}{\sqrt{23}-4} = \frac{7(\sqrt{23}+4)}{7} = 8 + \sqrt{23} - 4 \end{aligned}$$

Pode-se ver como a sequência se repete. Por brevidade, nota-se $\sqrt{23} = [4; (1, 3, 1, 8)]$, para indicar que o bloco $(1, 3, 1, 8)$ repete-se indefinidamente. As primeiras 10 frações continuadas representando raízes quadradas irracionais são:

$$\begin{aligned} \sqrt{2} &= [1; (2)], \text{ período } 1 \\ \sqrt{3} &= [1; (1, 2)], \text{ período } 2 \\ \sqrt{5} &= [2; (4)], \text{ período } 1 \\ \sqrt{6} &= [2; (2, 4)], \text{ período } 2 \\ \sqrt{7} &= [2; (1, 1, 1, 4)], \text{ período } 4 \\ \sqrt{8} &= [2; (1, 4)], \text{ período } 2 \\ \sqrt{10} &= [3; (6)], \text{ período } 1 \\ \sqrt{11} &= [3; (3, 6)], \text{ período } 2 \\ \sqrt{12} &= [3; (2, 6)], \text{ período } 2 \\ \sqrt{13} &= [3; (1, 1, 1, 1, 6)], \text{ período } 5 \end{aligned}$$

Para $N \leq 13$ exatamente 4 frações contínuas têm período ímpar. Para $N \leq 10000$ quantas frações contínuas têm período ímpar ?

Resposta: 1322

64.1 Solução em J

```
comprimento_peri:=: 3 : 0
m=:<.y^0.5
if. y=m*m do.
    0
    return.
end.
a=:m
d=:y-m*m
peri=:1
while. d>1 do.
    m=: ((<.(a+m)%d)*d)-m
    d=: <.(y-m*m)%d
    peri=:peri+1
end.
peri
)
euler64=: 3 : 0
ctd=:0
for_i. 2+i.9999 do.
    aa=:comprimento_periodo i
    if. 1=2|aa do.
        ctd=:ctd+1
    end.
end.
ctd
)

load 'c:/p/quarta/euler64.ijs'
euler64 ''
1322
```

64.2 Solução em Python

uma solução bem esperta baseada naquela do usuário

<https://betaprojects.com/solutions/project-euler/project-euler-problem-064-solution/>

Uma observação empírica que me ocorre: quanto mais complexa é a definição do problema, mais simples é a sua solução. Será ???

```
import math
def comprimento_periodo(n):
    m = math.isqrt(n)
    if m*m == n:
        return 0
    a = m
    d = n-m*m
    peri = 1
    while d > 1:
        m = ((a+m) // d)*d - m
        d = (n - m*m) // d
        peri = peri + 1
    return peri
ctd=0
for i in range(2,10001):
    aa=comprimento_periodo(i)
    if aa%2==1:
        ctd=ctd+1
```

```
print(ctd)
```

```
1322
```

Capítulo 65

Problema 65 - Convergência de e

A raiz quadrada de 2 pode ser escrita como uma fração continuada infinita

$$\sqrt{2} = 1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \dots}}}}$$

Esta fração infinita pode ser escrita como $\sqrt{2} = [1; (2)]$, (2) indica que o 2 se repete *ao infinito*. Igualmente, $\sqrt{23} = [4; (1, 3, 1, 8)]$. Acontece que a sequência dos valores parciais das frações contínuas da raiz quadrada provê as melhores aproximações racionais. Consideremos a convergência de $\sqrt{2}$

$$1 + \frac{1}{2} = \frac{3}{2}$$

$$1 + \frac{1}{2 + \frac{1}{2}} = \frac{7}{5}$$

$$1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2}}} = \frac{17}{12}$$

$$1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2}}}} = \frac{41}{29}$$

Aqui, a sequência das primeiras 10 convergências de $\sqrt{2}$ é

$$1, \frac{3}{2}, \frac{7}{5}, \frac{17}{12}, \frac{41}{29}, \frac{99}{70}, \frac{239}{169}, \frac{577}{408}, \frac{1393}{985}, \frac{3363}{2378}$$

O que é mais surpreendente é que a importante constante matemática e é

$$e = [2; 1, 2, 1, 1, 4, 1, 1, 6, 1, \dots, 1, 2k, 1, \dots]$$

Os primeiros dez termos da sequência de convergência são:

$$2, 3, \frac{8}{3}, \frac{11}{4}, \frac{19}{7}, \frac{87}{32}, \frac{106}{39}, \frac{193}{71}, \frac{1264}{465}, \frac{1457}{536}, \dots$$

A soma dos dígitos no numerador da 10ª convergência é $1 + 4 + 5 + 7 = 17$. Ache a soma dos dígitos do numerador da 100ª convergência da fração continuada para e .

65.1 Solução em Python

Tive muita dificuldade em entender o enunciado da questão. Ainda não sei direito como passar da sequência $e = [2; 1, 2, 1, 1, 4, 1, 1, 6, 1, \dots, 1, 2k, 1, \dots]$ para o cálculo das frações. Pedi ajuda para o Gemini e ele foi muito prestativo. Deu

uma explicação maravilhosa, mas eu não consegui que ele a reproduzisse. Daí pedi para ele a mesma explicação na forma algorítmica e ele me deu um código python para a solução. Mas, ele só ia até a 10 fração, e o problema do euler pedia a soma do 100º numerador. Precisei adaptar o código e *voil* a resposta deu certo na primeira

```
def calcular_convergentes_e(n):
    termos = [2, 1, 2]
    for i in range(3,n):
        if i%3==0 or i%3==1:
            termos=termos+[1]
        else:
            termos=termos+[((i//3)+1)*2]
    numeradores = [termos[0], termos[0] * termos[1] + 1]
    denominadores = [1, termos[1]]
    for i in range(2, n):
        numeradores.append(termos[i] * numeradores[i-1] + numeradores[i-2])
        denominadores.append(termos[i] * denominadores[i-1] + denominadores[i-2])
    return numeradores[n-1], denominadores[n-1]

numerador, denominador = calcular_convergentes_e(100)
nu=str(numerador)
so=0
for j in range(len(nu)):
    so=so+int(nu[j])
print(so)
print(numerador,denominador)

272
6963524437876961749120273824619538346438023188214475670667
2561737478789858711161539537921323010415623148113041714756
```

Após a solução deste problema (65) ocupo a posição 109 no Brasil.

65.2 Solução em J

```
calcular_convergentes =: 4 : 0
    termos=:2 1 2
    for_i. 3+i.y do.
        if. (0=3|i)+.1=3|i do.
            termos=:termos,1
        else.
            termos=:termos,2*1+<.i%3
        end.
    end.
    numeradores=: (0}termos), (1+0}termos*1}termos)
    denominadores=: 1,1}termos
    for_i. 2+i.y do.
        numeradores=: 1x*numeradores, ((i-2)}numeradores)+(i}termos)*(i-1)}numeradores
        denominadores=: 1x*denominadores, ((i-2)}denominadores)+(i}termos)*((i-1)}denominadores)
    end.
    if. x=1 do.
        (y-1)}numeradores
    else.
        (y-1)}denominadores
    end.
)

euler65=: 3 : 0
    num=:1 calcular_convergentes 100
    den=:2 calcular_convergentes 100
    nu=:":num
    so=:0
    for_j. i.$nu do.
```

```
        so=:so+ ".j}nu
end.
so
)

load 'c:/p/quarta/euler65.ijs'
euler65 ''
272
```

Capítulo 66

Problema 66 - Equação diofantina

Considere equações diofantinas (só envolvendo inteiros) na forma

$$x^2 - Dy^2 = 1$$

Por exemplo, quando $D = 13$, a solução mínima em x é $649^2 - 13 \times 180^2 = 1$. Assume-se que não há soluções em inteiros positivos quando D é quadrado. Pela busca de soluções mínimas em x para $D = \{2, 3, 5, 6, 7\}$ obtem-se o seguinte

$$3^2 - 2 \times 2^2 = 1$$

$$2^2 - 3 \times 1^2 = 1$$

$$9^2 - 5 \times 4^2 = 1$$

$$5^2 - 6 \times 2^2 = 1$$

$$8^2 - 7 \times 3^2 = 1$$

Daqui, considerando as soluções mínimas em x para $D \leq 7$, o maior valor de x é obtido quando $D = 5$. Ache o valor de $D \leq 1000$ no qual a solução mínima em x obtém o maior valor de x possível.

66.1 Solução em Python

Solução recuperada "as is" do usuário

<https://radiusofcircle.blogspot.com/2017/01/project-euler-problem-66-solution-with-python.html>.

```
# http://radiusofcircle.blogspot.com
```

```
#import time
from math import sqrt
# function to calculate the continued fraction
def cf(n):
    mn = 0.0
    dn = 1.0
    a0 = int(sqrt(n))
    an = int(sqrt(n))
    convergents = [a0]
    period = 0
    if a0 != sqrt(n):
        while an != 2*a0:
            mn = dn*an - mn
            dn = (n - mn**2)/dn
            an = int((a0 + mn)/dn)
            convergents.append(an)
    return convergents[:-1]

def cf_inv(cf):
    numerator = 1
    denominator = cf.pop()
    while cf:
        denominator, numerator = denominator*cf.pop() + numerator, denominator
```

```
        return denominator, numerator

largest = 0, 0

for i in range(1, 1001):
    if i%sqrt(i) != 0:
        continued_fraction = cf(i)
        if len(continued_fraction) % 2 != 0:
            u, v = cf_inv(continued_fraction)
            u, v = 2*u**2+1, 2*u*v
        else:
            u, v = cf_inv(continued_fraction)
        if u > largest[1]:
            largest = i, u

print(largest[0])

661
```

Capítulo 67

Problema 67 - Soma do caminho máximo II

No original conforme o site projecteuler.net

By starting at the top of the triangle below and moving to adjacent numbers on the row below, the maximum total from top to bottom is 23.

```
3
7 4
2 4 6
8 5 9 3
```

That is, $3 + 7 + 4 + 9 = 23$.

Find the maximum total from top to bottom in triangle.txt (right click and 'Save Link/Target As...'), a 15K text file containing a triangle with one-hundred rows.

NOTE: This is a much more difficult version of Problem 18. It is not possible to try every route to solve this problem, as there are 299 altogether! If you could check one trillion (10^{12}) routes every second it would take over twenty billion years to check them all. There is an efficient algorithm to solve it. ;o)

Answer: 7273

Soma de caminho máximo II Cujo enunciado é:

Começando na parte superior do triângulo e movendo-se aos números contíguos da fila de baixo, a rota máxima de acima abaixo é 23.

```
3
7 4
2 4 6
8 5 9 3
```

Ou seja, $3 + 7 + 4 + 9 = 23$.

Deve-se achar a rota máxima de acima abaixo no arquivo triangle.txt (disponível no site projecteuler.net sob o problema número 67) um arquivo de texto de 15KB que contém um triângulo com cem linhas.

Nota: Esta é uma versão muito mais difícil do problema 18. É impossível experimentar cada rota, dado que existem 2^{99} caminhos no total. Se você conseguisse comprovar um trilhão (10^{12}) de rotas por segundo, você precisaria de 20 bilhões de anos para comprovar todas. Obviamente é necessário um algoritmo mais eficiente.

Vale ressaltar que no problema euler 18, eu resolvi lá por força bruta através de uma elegante solução recursiva com 4 ou 5 linhas de código. É esta solução que está sendo desprezada aqui.

Primeiro, a decisão de como armazenar os dados. Escolhi montar uma matriz de 100 x 100 números, ainda que haja um desperdício de 50% (que são zeros). Mas este desperdício é compensado pela simplicidade do código.

A função APL2 que lê o arquivo original e monta esta matriz está no workspace euler711, com o nome de `acerta67`, lendo os dados originalmente guardados em `f:\p\varejao\euler\triangle67.txt`.

Sua execução cria uma variável de nome TRI67 dentro deste workspace. Esta variável pode ser descartada pois é facilmente recriada.

Dáí, fui no www.mathblog.dk na busca de ajuda. Não faltou, está lá.

Ajuda

O sujeito sugere usar uma estratégia de programação dinâmica aqui. Ele explica o algoritmo usando o exemplo

```

  3
 7 4
2 4 6
8 5 9 3

```

Começando no topo do triângulo (3), pode-se escolher à esquerda (7) ou à direita (4). Para decidir qual escolher, deve-se basicamente resolver dois problemas. O primeiro é

```

  7
 2 4
8 5 9

```

e o segundo é

```

  4
 4 6
5 9 3

```

Fazendo o mesmo raciocínio para baixo, vai-se avançando até o problema constar de um único número, cuja resposta é obviamente trivial (ou seja a resposta é o próprio único número). Tendo respondido este problema, pode-se avançar uma linha para cima, e responder à questão posta por

```

  x
 x x
a x x
b c x x

```

Que é exatamente $a + (b \lceil c)$. Agora pode-se ir para cima resolvendo cada um dos subproblemas propostos até chegar ao topo.

Sub-estrutura ótima

Olhando para os 2 subproblemas acima, vê-se que há uma parte que é comum

```

  4
5 9

```

Este subproblema tem resposta única não importando como ele foi calculado. Quando um problema tem esta propriedade diz-se que ele tem uma sub-estrutura ótima. Assim, ele (o subproblema) pode ser calculado uma única vez, economizando-se o que se perderia na busca bruta. Usando a programação dinâmica, na primeira iteração (última linha) são 3 comparações e 3 adições. Na próxima linha são 2 e 2 e a na primeira linha (última interação) é uma de cada. Total 6 comparações e 6 adições.

Para problemas pequenos pode-se dizer *grande coisa*. Mas para um triângulo com 15 linhas, vai-se passar de 14×2^{13} operações para 15 comparações e 15 adições.

Solução

Vai-se aproveitar a matriz, para evitar (e simplificar) a construção de subproblemas. Acompanhe os passos. Supondo que o problema começa com

```

  3
 7 4
 2 4 6
8 5 9 3

```

Usando a regra $a + (b \lceil c)$ de baixo para cima, na primeira iteração tem-se

```

  3
 7 4
10 13 15

```

Fazendo a segunda iteração fica

```

  3
20 19

```

E finalizando o algoritmo, a resposta está no topo da pirâmide que é

```

23

```

67.1 Solução em APL

```
      ∇euler67;x;i;j
[1]   x←TRI67
[2]   A dados estao em TRI67
[3]   i←99
[4]   t1:→(i<1)/fim
[5]   j←1
[6]   t2:→(j>i)/vorta
[7]   x[i;j]←x[i;j]+x[i+1;j][x[i+1;j+1]
[8]   j←j+1
[9]   →t2
[10]  vorta:i←i-1
[11]  →t1
[12]  fim:x[1;1]
```

```
      euler67
7273
```

67.2 Solução em J

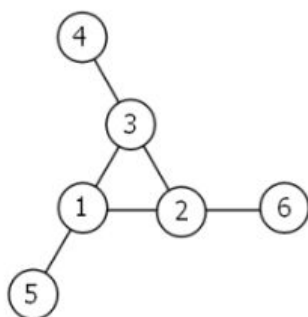
```
euler67=: 3 : 0
  data=: 'm' fread 'c:\p\quarta\0067_triangle.txt'
  dn=: 100 100$0
  for_i. i.100 do.
    dn=: (100{."i{data) i}dn
  end.
  i=.98
  while. i>:0 do.
    j=:0
    while. j<:i do.
      dn=: (((<i,j){dn)+((<(i+1),j){dn) >. (<(i+1),j+1){dn)(<i,j){dn
      j=:j+1
    end.
    i=.i-1
  end.
  (<0,0){dn
)

load 'c:/p/quarta/euler67.ijs'
euler67''
7273
```

Capítulo 68

Problema 68 - Anel 5-gon mágico

Considere o seguinte anel 3-gon mágico, preenchido com os números de 1 a 6 e com cada linha somando 9

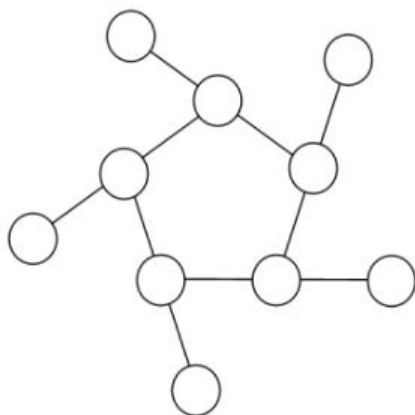


Começando com o grupo de menor valor externo (4,3,2) no exemplo, e indo na direção horária, cada solução pode ser descrita de maneira única. Por exemplo, a solução acima pode ser descrita pelo conjunto 4,3,2;6,2,1;5,1,3.

É possível completar o anel com 4 totais diferentes: 9, 10, 11 e 12. Existem 8 soluções no total.

Total	Conjunto de soluções
9	4,2,3; 5,3,1; 6,1,2
9	4,3,2; 6,2,1; 5,1,3
10	2,3,5; 4,5,1; 6,1,3
10	2,5,3; 6,3,1; 4,1,5
11	1,4,6; 3,6,2; 5,2,4
11	1,6,4; 5,4,2; 3,2,6
12	1,5,6; 2,6,4; 3,4,5
12	1,6,5; 3,5,4; 2,4,6

Usando os números de 1 a 10, e dependendo dos arranjos, é possível formar soluções de 16 e 17 dígitos. Qual é o maior conjunto de 16 dígitos para um anel mágico 5-gon ?



68.1 Solução em Python

Solução capturada do site

<https://www.ivl-projecteuler.com/overview-of-problems/25-difficulty/problem-68>. com peque-

nas adaptações.

```
def join_str(alist):
    temp_list = alist
    return "".join(str(temp_list[y]) for y in range(len(temp_list)))

def compute():
    numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
    final_list = []
    for a in numbers:
        numbers_b = numbers[:]
        numbers_b.remove(a)
        for b in numbers_b:
            numbers_c = numbers_b[:]
            numbers_c.remove(b)
            for c in numbers_c:
                numbers_d = numbers_c[:]
                numbers_d.remove(c)
                for d in numbers_d:
                    numbers_e = numbers_d[:]
                    numbers_e.remove(d)
                    for e in numbers_e:
                        numbers_f = numbers_e[:]
                        numbers_f.remove(e)
                        for f in numbers_f:
                            numbers_g = numbers_f[:]
                            numbers_g.remove(f)
                            for g in numbers_g:
                                numbers_h = numbers_g[:]
                                numbers_h.remove(g)
                                for h in numbers_h:
                                    numbers_i = numbers_h[:]
                                    numbers_i.remove(h)
                                    for i in numbers_i:
                                        numbers_j = numbers_i[:]
                                        numbers_j.remove(i)
                                        for j in numbers_j:
                                            if a + b + c == d + c + e == f + e + g == h + g + i == j + i + b:
                                                if a < d and a < f and a < h and a < j:
                                                    temp_list = [a, b, c, d, c, e, f, e, g, h, g, i, j, i, b]
                                                    temp_var = join_str(temp_list)
                                                    final_list.append(int(temp_var))

    return (final_list)
print(compute())
```

Capítulo 69

Problema 69 - Totiente máximo

A Função totiente de Euler, $\phi(n)$, às vezes chamada de função phi, é definida como o número de inteiros positivos que não excedem n e que são relativamente primos para n . Por exemplo, 1,2,4,5,7 e 8 são todos menores do que 9 e primos relativo a 9, ou seja $\phi(9) = 6$.

n	Relativamente primo	$\phi(n)$	$n/\phi(n)$
2	1	1	2
3	1,2	2	1.5
4	1,3	2	2
5	1,2,3,4	4	1.25
6	1,5	2	3
7	1,2,3,4,5,6	6	1.1666...
8	1,3,5,7	4	2
9	1,2,4,5,7,8	6	1.5
10	1,3,7,9	4	2.5

Pode-se observar que $n = 6$ produz um máximo de $n/\phi(n)$, para $n \leq 10$. Encontre o valor de d , ($d \leq 1000000$) para o qual $n/\phi(n)$ é um máximo.

69.1 Solução em J

Bingo, de primeira ! Traduzi de APL para J as-is. O APL demora muito, pois as funções são construídas do zero. Em J usei primitivas poderosas.

```
euler69=: 3 : 0
  n=:1000000
  max=: -9999
  while. n>1 do.
    f=:5 p: n
    h=:n%f
    if. h>max do.
      max=:h
      nmax=:n
    end.
    n=:n-1
  end.
  nmax
)

load 'c:/p/quarta/euler69.ijs'
euler69''
510510
```

Capítulo 70

Problema 70 - Permutação de totientes

A Função totiente de Euler, $\phi(n)$, às vezes chamada de função phi, é definida como o número de inteiros positivos que não excedem n e que são relativamente primos para n . Por exemplo, 1, 2, 4, 5, 7 e 8 são todos menores do que 9 e primos relativo a 9, ou seja $\phi(9) = 6$.

O número 1 é considerado primo relativamente a todo número positivo, então $\phi(1) = 1$. Interessantemente, $\phi(87109) = 79180$ e pode-se observar que 87109 é uma permutação de 79180. Encontre o valor de n , $1 < n < 10^7$ para o qual $\phi(n)$ é uma permutação de n e a proporção $n/\phi(n)$ produz um mínimo.

70.1 Solução em Python

obtida em

<https://betaprojects.com/solutions/project-euler/project-euler-problem-070-solution/>.

```
def is_permutation(a, b):
    return sorted(str(a)) == sorted(str(b))

def solve(limit):
    phi = list(range(limit))
    min_ratio = float('inf')
    result = 0

    # Calculate all totients using sieve method
    for i in range(2, limit):
        if phi[i] == i: # i is prime
            for j in range(i, limit, i):
                phi[j] = phi[j] * (i - 1) // i

    # Check if current number is a solution
    for i in range(2, limit):
        if i > 1:
            ratio = i / phi[i]
            if ratio < min_ratio and is_permutation(i, phi[i]):
                min_ratio = ratio
                result = i

    return result

print(solve(int(input())))
8319823
```

Tentei reproduzir esta implementação em J e não consegui. Há um comportamento esquisito aqui, veja-se

```
9999999%9999950
1
9999999%9999949
1.00001
```

Por conta desta “ineficiência” do J, que ainda não descobri como contornar/manusear o meu programa J do problema dava resposta errada. A propósito, anteriormente, fiz uma implementação da minha cabeça (seguindo estritamente a definição do problema Euler) == pena que não consigo mais a recuperar == e o mesmo erro acima reportado aconteceu. Para não perder o registro segue o programa J (que NÃO funciona)

```
permuta=: 4 : 0
  xx=: ":x
  yy=: ":y
  if. (($xx)=$yy) do.
    if. ($xx)=+/(xx/:xx)=yy/:yy do.
      1
    return.
  end.
end.
0
)
euler70=: 3 : 0
  phi=:i.10000000
  mini=:999999x
  resu=:0
  for_i. 2+i.9999998 do.
    if. i=i{phi do.
      j=:i
      while. j<9999998 do.
        phi=:((j{phi) * <.(i-1)%i) j}phi
        j=:j+i
      end.
    end.
    if. i>1 do.
      ratio=:i%i{phi
      if. (ratio < mini) *. i permuta i{phi do.
        mini=:ratio
        resu=:i
      end.
    end.
  end.
  resu
)
```

Bola pra frente !

Capítulo 71

Problema 71 - Frações ordenadas

Considere a fração $\frac{n}{d}$, onde n e d são inteiros positivos. Se $n < d$ e $HCF(n, d) = 1$ ($HCF = \text{highest common factor}$) ela é chamada uma fração própria reduzida. Se listarmos as frações própria reduzidas com $d \leq 8$ em ordem ascendente de tamanho, obtem-se:

$$\frac{1}{8}, \frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{2}{7}, \frac{1}{3}, \frac{3}{8}, \frac{2}{5}, \frac{3}{7}, \frac{1}{2}, \frac{4}{7}, \frac{3}{5}, \frac{5}{8}, \frac{2}{3}, \frac{5}{7}, \frac{3}{4}, \frac{5}{6}, \frac{6}{7}, \frac{7}{8}$$

Pode-se ver que $\frac{2}{5}$ é a fração imediatamente à esquerda de $\frac{3}{7}$. Se listarmos o conjunto de frações próprias reduzidas com $d \leq 1000000$, em ordem ascendente, ache o numerador da fração imediatamente à esquerda de $\frac{3}{7}$.

A resposta (que encontrei em 16/mar/2017) é 428570. Não consegui achar a minha solução, então vão 3 achadas à esmo.

71.1 Soluções em J

Só para mostrar a potência desta linguagem

```
2 {.@x: {:(#~ <&(3%7)) /:~(1+9e5+i.1e5)(<.@* % []) 3%7
428570
(((({"1)2 x:n)<:1000000)i:1){n=:3r7-%7*1+i.999999x
428570r999997
```

71.2 Solução em Python

```
N=3.0/7
M=1000000
mina=N
min_=0
for b in range(1,M+1):
    a=int(b*N)
    if mina > N-1.0*a/b != 0:
        mina=N-1.0*a/b
        min_=b
print('Done,', int(min_*N), '/', min_)
Done, 428570 / 999997
```

Capítulo 72

Problema 72 - Contando frações

Considere a fração $\frac{n}{d}$, onde n e d são inteiros positivos. Se $n < d$ e $HCF(n, d) = 1$ ($HCF=$ *highest common factor*) ela é chamada uma fração própria reduzida. Se listarmos as frações própria reduzidas com $d \leq 8$ em ordem ascendente de tamanho, obtem-se:

$$\frac{1}{8}, \frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{2}{7}, \frac{3}{8}, \frac{2}{5}, \frac{3}{7}, \frac{1}{2}, \frac{4}{7}, \frac{3}{5}, \frac{5}{8}, \frac{2}{3}, \frac{4}{5}, \frac{3}{4}, \frac{5}{6}, \frac{6}{7}, \frac{7}{8}$$

Pode-se ver que há 21 elementos nesse conjunto. Quantos elementos há no conjunto de frações próprias reduzidas com $d \leq 1000000$?

Resposta: 303963552391

72.1 Solução em Python

Construída a partir de uma consulta a <https://raw.org/puzzle/project-euler/problem-72/>.

```
lim=1000000
phi=list(range(lim+1))
suma=0
for i in range(2,lim+1):
    if phi[i]==i:
        for j in range(i,lim+1,i):
            phi[j]=phi[j]/i*(i-1)
        suma=suma+phi[i]
print(suma)
```

303963552391.0

72.2 Solução em J

Em `phi[j]=phi[j]/i*(i-1)` (Python), há que converter para `phi=:(((j{phi}%i)*i-1)(j))phi` (J). Note um parênteses adicional necessário. Sem ele o resultado é 3052227 e não 303963552391 como deve ser.

```
euler72=: 3 : 0
    lim=:1000000
    phi=:i.1+lim
    suma=:0
    for_i. 2+i.lim-1 do.
        if. i=i{phi do.
            j=:i
            while. j<lim+1 do.
                phi=:(((j{phi}%i)*i-1)(j))phi
                j=:j+i
            end.
        end.
    end.
```

```
        suma=:suma+i{phi
end.
30j0":suma
)
```

```
load'c:/p/quarta/euler72.ijs'
euler72''
303963552391
```

Capítulo 73

Problema 73 - Contando frações em intervalo

Considere a fração $\frac{n}{d}$, onde n e d são inteiros positivos. Se $n < d$ e $HCF(n, d) = 1$ ($HCF = \text{highest common factor}$) ela é chamada uma fração própria reduzida. Se listarmos as frações própria reduzidas com $d \leq 8$ em ordem ascendente de tamanho, obtem-se:

$$\frac{1}{8}, \frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{2}{7}, \frac{1}{3}, \frac{3}{8}, \frac{2}{5}, \frac{3}{7}, \frac{1}{2}, \frac{4}{7}, \frac{3}{5}, \frac{5}{8}, \frac{2}{3}, \frac{5}{7}, \frac{3}{4}, \frac{5}{6}, \frac{6}{7}, \frac{7}{8}$$

Pode-se ver que existem 3 frações entre $\frac{1}{3}$ e $\frac{1}{2}$. Quantas frações existem entre $\frac{1}{3}$ e $\frac{1}{2}$ no conjunto ordenado de frações próprias reduzidas para $d \leq 12000$?

A resposta é 7295372.

Solução para o problema 73 do Projeto Euler, Copyright (c) Projeto Nayuki. Todos os direitos reservados.

<https://www.nayuki.io/page/project-euler-solutions> <https://github.com/nayuki/Project-Euler-solutions>

A árvore de Stern-Brocot é uma árvore de busca binária infinita de todos os números racionais positivos, onde cada número aparece apenas uma vez e está em termos mais baixos. Ela é formada começando com os dois sentinelas $0/1$ e $1/1$. Iterando infinitamente em qualquer ordem, entre quaisquer duas frações atualmente adjacentes L_n/L_d e R_n/R_d , insira uma nova fração $(L_n+R_n)/(L_d+R_d)$. Veja MathWorld para uma visualização: <http://mathworld.wolfram.com/Stern-BrocotTree.html>

O algoritmo natural é o seguinte: Conta o número de frações reduzidas n/d de modo que $leftN/leftD < n/d < rightN/rightD$ e $d \leq 12000$.

$leftN/leftD$ e $rightN/rightD$ devem ser adjacentes na árvore de Stern-Brocot em algum ponto do processo de geração.

```
def stern_brocot_count(leftn, leftd, rightn, rightd):
    d = leftd + rightd
    if d > 12000:
        return 0
    else:
        n = leftn + rightn
        return 1 + stern_brocot_count(leftn, leftd, n, d) +
            stern_brocot_count(n, d, rightn, rightd)
```

Mas, em vez disso, usamos a busca em profundidade em uma pilha explícita, porque ter um grande número de quadros de pilha parece ser suportado no Linux, mas não no Windows.

73.1 Solução em Python

```
def compute():
    ans = 0
    stack = [(1, 3, 1, 2)]
    while len(stack) > 0:
        leftn, leftd, rightn, rightd = stack.pop()
        d = leftd + rightd
        if d <= 12000:
            n = leftn + rightn
            ans = ans + 1
            stack.append((n, d, rightn, rightd))
            stack.append((leftn, leftd, n, d))
```

```
    return str(ans)
print(compute())
```

```
7295372
```

73.2 Solução em J

```
euler73=: 3 : 0
  ans=:0
  stack=:1 4$1 3 1 2
  while. (1{.$stack) > 0 do.
    leftn =: (<0,0){stack
    leftd =: (<0,1){stack
    rightn =: (<0,2){stack
    rightd =: (<0,3){stack
    stack=:1}.stack
    d =: leftd+rightd
    if. d <: 12000 do.
      n=:leftn+rightn
      ans=:ans+1
      stack=: (1 4$n,d,rightn,rightd),stack
      stack=: (1 4$leftn,leftd,n,d),stack
    end.
  end.
  ans
)
```

```
load 'c:/p/quarta/euler73.ijs'
euler73''
7295372
```

Capítulo 74

Problema 74 - Cadeia de dígitos no fatorial

O número 145 é bem conhecido pela propriedade de que a soma dos fatoriais dos seus dígitos é igual a 145.

$$1! + 4! + 5! = 1 + 24 + 120 = 145$$

Talvez menos conhecido seja o 169, para o qual é produzida uma cadeia mais longa de números associados a 169. Verifica-se que existem apenas 3 desses laços:

$$169 \rightarrow 363601 \rightarrow 1454 \rightarrow 169$$

$$871 \rightarrow 45361 \rightarrow 871$$

$$872 \rightarrow 45362 \rightarrow 872$$

Não é difícil provar que todo número inicial acabará ficando preso em um laço. Exemplos

$$69 \rightarrow 363600 \rightarrow 1454 \rightarrow 169 \rightarrow 363601 \text{ (to1454)}$$

$$78 \rightarrow 45360 \rightarrow 871 \rightarrow 45361 \text{ (} \rightarrow 871 \text{)}$$

$$540 \rightarrow 145 \text{ (} \rightarrow 145 \text{)}$$

Começando com 69 se produz uma cadeia com 5 termos não repetidos, mas a maior cadeia não repetida com um número inicial abaixo de 1 milhão é de 60 termos. Quantas cadeias, com número inicial abaixo de 1000000, contém exatamente 60 termos não repetidos ?

A resposta é 402.

74.1 Solução em APL

```

      ∇r←eul74;i;ctd;seq;z;w
[1]   i←2
[2]   ctd←0
[3]   t1:→(i>1000000)/fim
[4]   z←i
[5]   seq←,z
[6]   t2:w←fatu(z)
[7]   →(wεseq)/dobrou
[8]   seq←seq,w
[9]   z←w
[10]  →t2
[11]  dobrou:
[12]  →(60≠pseq)/soma
[13]  ctd←ctd+1
[14]  soma:
[15]  i←i+1
[16]  →(0≠1000000|i)/t1
[17]  □←i
[18]  →t1
[19]  fim:r←ctd

      eul74
402
```

74.2 Solução em J

```
fatu=: 3 : 0
  fa=.1 1 2 6 24 120 720 5040 40320 362880
  z=.:y
  i=. 0
  r=. 0
  while. i<$z do.
    r=. r+ (".i{z){fa
    i=. i+1
  end.
  r
)
euler74=: 3 : 0
  ctd=.0
  for_j. 2+i.999999 do.
    zz=. j
    seq=.,zz
    label_t2.
    w=. fatu zz
    if. w e. seq do.
      if. 60=$seq do.
        ctd=.ctd+1
      end.
      continue.
    end.
    seq=.seq,w
    zz=. w
    goto_t2.
  end.
  ctd
)
```

```
load 'c:/p/quarta/euler74.ijs'
euler74''
402
```

Capítulo 75

Problema 75 - Triângulos retângulos inteiros singulares

Acontece que 12cm é o menor pedaço de fio que pode ser dobrado para formar um triângulo retângulo de lados inteiros exatamente de uma maneira, mas há muitos outros exemplos. $12\text{cm} : (3, 4, 5)$

$24\text{cm} : (6, 8, 10)$

$30\text{cm} : (5, 12, 13)$

$36\text{cm} : (9, 12, 15)$

$40\text{cm} : (8, 15, 17)$

$48\text{cm} : (12, 16, 20)$

Em contraste, alguns comprimentos de fio, como 20cm , não pode ser dobrado para formar um triângulo retângulo de lados inteiros, e outros comprimentos permitem que mais de uma solução seja encontrada; por exemplo, usando 120cm é possível formar exatamente três triângulos retângulos de lados inteiros diferentes.

$120\text{cm} : (30, 40, 50), (20, 48, 52), (24, 45, 51)$

Dado que L é o comprimento de um fio, para quantos valores de $L \leq 1500000$ é possível formar exatamente um triângulo retângulo com lados inteiros ?

75.1 Solução em Python

```
# https://www.nayuki.io/page/project-euler-solutions
import math
def compute():
    LIMIT = 1500000
    triples = set()
    for s in range(3, math.isqrt(LIMIT) + 1, 2):
        for t in range(s - 2, 0, -2):
            if math.gcd(s, t) == 1:
                a = s * t
                b = (s * s - t * t) // 2
                c = (s * s + t * t) // 2
                if a + b + c <= LIMIT:
                    triples.add((a, b, c))
    ways = [0] * (LIMIT + 1)
    for triple in triples:
        sigma = sum(triple)
        for i in range(sigma, len(ways), sigma):
            ways[i] = ways[i] + 1
    ans = ways.count(1) #retorna quantos ways valem 1
    return str(ans)

print(compute())

161667
```

75.2 Solução em J

```
euler75=: 3 : 0
  li=:1500000
  tri=:0 3$0
  s=.3
  while. s< <.1+li^0.5 do.
    t=. s-2
    while. t>0 do.
      if. 1=s +. t do.
        a=.s*t
        b=.<.0.5*(s*s)-(t*t)
        c=.<.0.5*(s*s)+(t*t)
        if. li>a+b+c do.
          tri=.tri,1 3$a,b,c
        end.
      end.
      t=.t-2
    end.
    s=.s+2
  end.
  wa=.(li+1)$0
  for_j. i.1{.$tri do.
    sig=.+/j{tri
    i=.sig
    while. i<li+1 do.
      wa=.(1+i{wa) i}wa
      i=.i+sig
    end.
  end.
  +/wa=1
)

load'c:/p/quarta/euler75.ijs'
euler75''
161667
```

Capítulo 76

Problema 76 - Contando somas

É possível escrever 5 como uma soma de 6 maneiras diferentes:

$4 + 1$

$3 + 2$

$3 + 1 + 1$

$2 + 2 + 1$

$2 + 1 + 1 + 1$

$1 + 1 + 1 + 1 + 1$

De quantas maneiras 100 pode ser escrita como uma soma de ao menos dois inteiros positivos ?

76.1 Solução em Python

```
# https://www.nayuki.io/page/project-euler-solutions
def compute():
    LIMIT = 100
    partitions = []
    for i in range(LIMIT + 1):
        partitions.append([None] * (LIMIT + 1))
    for j in reversed(range(LIMIT + 1)):
        if j == i:
            val = 1
        elif j > i:
            val = 0
        elif j == 0:
            val = partitions[i][j + 1]
        else:
            val = partitions[i][j + 1] + partitions[i - j][j]
        partitions[i][j] = val
    ans = partitions[LIMIT][1] - 1
    return str(ans)
print(compute())

190569291
```

76.2 Solução em J

```
euler76=: 3 : 0
    par=. 101 101$_1
    for_i. i.101 do.
        for_j. |.i.101 do.
            if. i=j do.
                val=.1
```

```
elseif. j>i do.
  val=.0
elseif. j=0 do.
  val=.(<i,j+1){par
else.
  val=.((<i,j+1){par)+(<(i-j),j){par
end.
par=.val (<i,j)}par
end.
end.
_1+(<100 1){par
)

load'c:/p/quarta/euler76.ijs' euler76'' 190569291
```

Capítulo 77

Problema 77 - Sumarizações de primos

É possível escrever 10 como uma soma de primos de exatas 5 maneiras diferentes:

$7 + 3$

$5 + 5$

$5 + 3 + 2$

$3 + 3 + 2 + 2$

$2 + 2 + 2 + 2 + 2$

Qual é o primeiro valor que pode ser escrito como uma soma de primos de 5000 maneiras diferentes ?

A resposta é 71.

Primeiro uma solução mandraque do usuário olegyk em J

```
dp=: +/@:~.@q:
<: # ( , ( +/@: ( * dp@>: @i. @-@# ) % # ) ) ^: ( <5000@{ : ) ^: _ ] 1 0
```

71

77.1 Solução em Python

```
#usuario dwrpayne no forum do problema, com adaptações
import eulerlib
primes = eulerlib.primes(1000)

def countprimesums(used, remaining, min):
    if used > 1 and remaining == 0: return 1
    count = 0
    for p in primes:
        if p > remaining: break
        if p < min: continue
        count = count + countprimesums(used+1, remaining-p, p)
    return count

for num in range(1000):
    if countprimesums(0, num, 1) > 5000:
        print (num)
        break
```

71

77.2 Solução em J

```
pri=:p:i.1000
countprim=: 3 : 0
```

```

    if.((0{y}>1) *. (0=1{y) do.
        1
        return.
    end.
    ct=. 0
    for_j. i.$pri do.
        ca=.j{pri
        if. ca>1{y do. break. end.
        if. ca<2{y do. continue. end.
        ct=.ct+countprim (1+0{y),((1{y)-ca),ca
    end.
    ct
)
euler77=: 3 : 0
    for_num. i.1000 do.
        if. (countprim 0,num,1)>5000 do.
            num
            break.
        end.
    end.
end.
)

load'c:/p/quarta/euler77.ijs'
euler77''
71

```

Capítulo 78

Problema 78 - Partições de moedas

Seja $p(n)$ a representação do número de maneiras diferentes pelas quais n moedas podem ser separadas em pilhas. Por exemplo, 5 moedas podem ser separadas em pilhas de exatamente 7 maneiras diferentes, então $p(5) = 7$.

```
ooooo
oooo o
ooo oo
ooo o o
oo oo o
oo o o o
o o o o o
```

Ache o menor valor de n para o qual $p(n)$ seja divisível por 1 milhão.
A resposta é 55374.

78.1 Solução em Python

```
#usuário dpnash, do fórum de solução do problema 78 no euler
import math
def generalized_pentagonal(n):
    n_1 = (math.floor(n+1) // 2)
    if (n % 2 == 0):
        n_1 = n_1 * -1
    return (3*n_1*n_1 - n_1) // 2

partitions = {0: 0, 1: 1, 2: 1}
n = 100000
for i in range(1, n):
    if (i < 2):
        p = 1
    else:
        p = 0
        k = 0
        while (k < i):
            k = k + 1
            gp = generalized_pentagonal(k)
            sgn = 1
            if ((math.floor(k+1) // 2) % 2 == 0):
                sgn = -1
            if (i-gp < 0):
                break
            p = p + (sgn * partitions[i-gp])
        partitions[i] = p
    if (p % int(1e6) == 0):
        print(f'result for n={i-1} is {p}')
        break
```

result for n=55374

Uma dica preciosa de Python, aproveitando o ensejo. Seja o dicionário

```
d = {0:5, 1:11, 4:34} #duplas key,value
Para acessar o value, escreve-se d[0] que é igual a 5
Para alterar, d[0]=22, 22 é o novo valor da chave 0
Se a chave não existe da erro
d.keys() - volta todas as chaves # dict_keys([0, 1, 4])
d.values() - volta todos os valores # dict_values([5, 11, 34])
for cha, val in d.items(): # dict_items([(0, 5), (1, 11), (4, 34)])
    print(f"{cha}:{val}")
```

Capítulo 79

Problema 79 - Derivação de senha

Um método de segurança comum usado para serviços bancários on-line é pedir ao usuário três caracteres aleatórios de uma senha. Por exemplo, se a senha fosse 531278, eles poderiam pedir o 2º, 3º e 5º caracteres; a resposta esperada seria: 317.

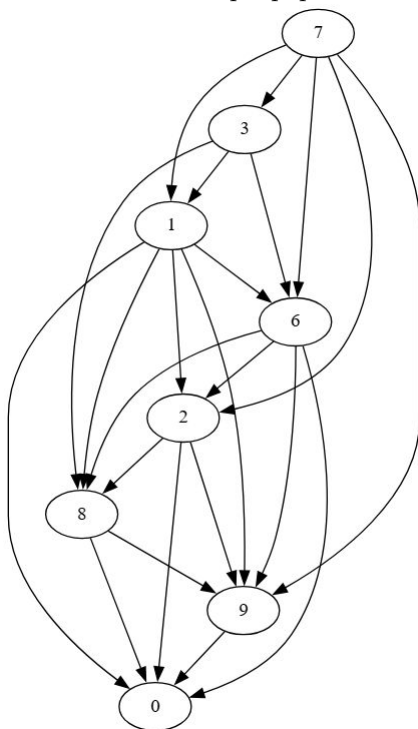
O arquivo de texto, keylog.txt, contém cinquenta tentativas de login bem-sucedidas.

Considerando que os três caracteres são sempre solicitados em ordem, analise o arquivo para determinar a senha secreta mais curta possível, de comprimento desconhecido.

O conteúdo do arquivo citado é

319, 680, 180, 690, 129, 620, 762, 689, 762, 318, 368, 710, 720, 710, 629, 168, 160,
689, 716, 731, 736, 729, 316, 729, 729, 710, 769, 290, 719, 680, 318, 389, 162, 289,
162, 718, 729, 319, 790, 680, 890, 362, 319, 760, 316, 729, 380, 319, 728, 716

Uma maneira lapis-papel de resolver este exercício é desenhando o grafo a seguir



A solução é 73162890.

Capítulo 80

problema 80 - Expansão digital da raiz quadrada

É bem sabido que se a raiz quadrada de um número natural não for um inteiro, então ele é irracional. A expansão decimal de tais raízes quadradas é infinita sem qualquer padrão de repetição. A raiz quadrada de dois é 1.41421356237309504880..., e a soma digital dos primeiros cem dígitos decimais é 475. Para os primeiros cem números naturais, encontre o total das somas digitais dos primeiros cem dígitos decimais para todas as raízes quadradas irracionais. A resposta é 40886

80.1 Solução em APL

```

      ∇su←euler80;i;su
[1]  su←0
[2]  i←1
[3]  t1:→(i>100)/fim
[4]  →(i∈(ι10)*2)/soma
[5]  su←su+somad80 100 sqr80 i
[6]  soma:i←i+1
[7]  →t1
[8]  fim:

      ∇r←somad80 x;i;s
[1]  i←1
[2]  s←0
[3]  t1:→(i>ρx)/fim
[4]  s←s+⊥x[i]
[5]  i←i+1
[6]  →t1
[7]  fim:r←s

      ∇r←d sqr80 n;a;b;li
[1]  li←d+1
[2]  a←⌈5×n
[3]  b←⌈5
[4]  t1:
[5]  ⌘ ('a=',a), '    b=',b
[6]  →((ρb)>li)/ufa
[7]  →(a maiig80 b)/foi
[8]  naofoi:a←a,'00'
[9]  b←b[ι-1+ρb], '05'
[10] →junta
[11] foi:
[12] a←a menos80 b
[13] b←b mais80 '10'
[14] junta:→t1
[15] ufa:r←-2↓b

      ∇r←a mais80 b;t;aa;bb;i;vu;aux
[1]  t←(ρ,a)[ρ,b

```

```

[2] aa←' 0' edit(-t)†a
[3] bb←' 0' edit(-t)†b
[4] r←tp' '
[5] i←t
[6] vu←0
[7] t1:→(i<1)/fim
[8] aux←(‡aa[i])+(‡bb[i])+vu
[9] →(aux>9)/vaium
[10] r[i]←1 0‡aux
[11] vu←0
[12] →pula
[13] vaium:
[14] r[i]←1 0‡aux-10
[15] vu←1
[16] pula:i←i-1
[17] →t1
[18] fim:

```

```

    ∇r←a menos80 b;t;aa;bb;i;vu;aux
[1] t←(ρ,a)†ρ,b
[2] aa←' 0' edit(-t)†a
[3] bb←' 0' edit(-t)†b
[4] r←tp' '
[5] i←t
[6] vu←0
[7] t1:→(i<1)/fim
[8] aux←((‡aa[i])-vu)-(‡bb[i])
[9] →(aux<0)/vaium
[10] r[i]←1 0‡aux
[11] vu←0
[12] →pula
[13] vaium:
[14] r[i]←1 0‡10+aux
[15] vu←1
[16] pula:i←i-1
[17] →t1
[18] fim:→(r[1]='0')/tira
[19] →0
[20] tira:r←1↓r

```

```

    ∇r←a maiig80 b;t;aa;bb;i
[1] t←(ρ,a)†ρ,b
[2] aa←' 0' edit(-t)†a
[3] bb←' 0' edit(-t)†b
[4] r←1
[5] i←1
[6] t1:→(i>t)/acabou
[7] →((‡aa[i])>‡bb[i])/sim
[8] →((‡aa[i])<‡bb[i])/nao
[9] i←i+1
[10] →t1
[11] sim:r←1
[12] →0
[13] nao:r←0
[14] →0
[15] acabou:r←1
[16] →0
[17]

```

euler80

40886

80.2 Solução em Python

Solução obtida do usuário ilan no fórum de solução do problema

```
from decimal import *
getcontext().prec = 105
print(sum( sum(map(int,str(Decimal(n).sqrt())
    .replace('.', ''))[:100] )) for n in range(100)
    if n not in [j*j for j in range(10)] ))
40886
```

Capítulo 81

Problema 81 - Soma dos caminhos: duas maneiras

Na matriz 5×5 abaixo a soma mínima do caminho do canto superior esquerdo para o canto inferior direito, movendo-se apenas para a direita e para baixo, é indicada em vermelho negrito e é igual a 2427

$$\begin{pmatrix} \mathbf{131} & 673 & 234 & 103 & 18 \\ \mathbf{201} & \mathbf{96} & \mathbf{342} & 965 & 150 \\ 630 & 803 & \mathbf{746} & \mathbf{422} & 111 \\ 537 & 699 & 497 & \mathbf{121} & 956 \\ 805 & 732 & 524 & \mathbf{37} & \mathbf{331} \end{pmatrix}$$

Encontre a soma mínima do caminho do canto superior esquerdo para o canto inferior direito movendo apenas para a direita e para baixo em matrix.txt (clique com o botão direito e "Salvar link/destino como..."), um arquivo de texto de 31K contendo uma matriz 80×80 .

A solução é 427337.

81.1 Solução em APL

```
      ∇r←euler81;x;i;j
[1]  A p81 eh a matriz do problema 81
[2]  x←p81
[3]  i←79
[4]  t1:→(i<1)/f1
[5]  x[80;i]←x[80;i]+x[80;i+1]
[6]  x[i;80]←x[i;80]+x[(i+1);80]
[7]  i←i-1
[8]  →t1
[9]  f1:Ax
[10] i←79
[11] t2:→(i<1)/f5
[12] j←79
[13] t3:→(j<1)/f6
[14] x[i;j]←x[i;j]+x[i+1;j][x[i;j+1]
[15] j←j-1
[16] →t3
[17] f6:i←i-1
[18] →t2
[19] f5:x[1;1]
```

81.2 Solução em J

```
euler81=: 3 : 0
  data=: 'm' fread 'c:\p\quarta\0081_matrix.txt'
  dn=:80 80$0
```

```

for_i. i.80 do.
  dn:=("i{data}i)dn
end.

i=.78
while.i>:0 do.
  dn:=(((<79,i){dn)+( <79,i+1){dn)( <79,i){dn
  dn:=(((<i,79){dn)+( <(i+1),79){dn)( <i,79){dn
  i=.i-1
end.

i=.78
while. i>:0 do.
  j=.78
  while. j>:0 do.
    dn:=(((<i,j){dn)+( <(i+1),j){dn)<.( <i,j+1){dn))( <i,j){dn
    j=.j-1
  end.
  i=.i-1
end.
(<0,0){dn
)

load 'c:/p/quarta/euler81.ijs'
euler81''
427337

```

Capítulo 82

Problema 82 - Soma do caminho: três caminhos

A soma mínima do caminho na matriz abaixo, 5×5 , começando em qualquer célula da coluna da esquerda e terminando em qualquer célula da coluna da direita, e movendo apenas para cima, para baixo e para a direita, é indicada em vermelho e negrito; a soma é igual a 994.

$$\begin{pmatrix} 131 & 673 & \mathbf{234} & \mathbf{103} & \mathbf{18} \\ \mathbf{201} & \mathbf{96} & \mathbf{342} & 965 & 150 \\ 630 & 803 & 746 & 422 & 111 \\ 537 & 699 & 497 & 121 & 956 \\ 805 & 732 & 524 & 37 & 331 \end{pmatrix}$$

Encontre a soma mínima do caminho da coluna da esquerda para a coluna da direita em matrix.txt, um arquivo de 31Kb, contendo uma matriz de 80×80 .

A resposta é 260324.

82.1 Solução em APL

```

    ∇euler82;x;tam;sol;i;j
[1]  x←p82
[2]  tam←80
[3]  sol←80ρ0
[4]  i←1
[5]  t1:→(i>tam)/f1
[6]  sol[i]←x[i;tam]
[7]  i←i+1
[8]  →t1
[9]  f1:i←tam-1
[10] t2:→(i<1)/f2
[11] sol[1]←sol[1]+x[1;i]
[12] j←2
[13] t3:→(j>tam)/f3
[14] sol[j]←(sol[j-1]+x[j;i])(sol[j]+x[j;i])
[15] j←j+1
[16] →t3
[17] f3:j←tam-1
[18] t4:→(j<1)/f5
[19] sol[j]←sol[j][sol[j+1]+x[j;i]
[20] j←j-1
[21] →t4
[22] f5:i←i-1
[23] →t2
[24] f2: [/sol

    euler82
260324
```

82.2 Solução em J

```
euler82=: 3 : 0
  data=: 'm' fread 'c:\p\quarta\0082_matrix.txt'
  dn=:80 80$0
  for_i. i.80 do.
    dn=:("i{data)i}dn
  end.
  sol=.80$0
  i=.0
  tam=.80
  while. i<tam do.
    ((<i,tam-1){dn) i}sol
    i=.i+1
  end.
  i=.tam-1
  while. i>:0 do.
    sol=.((0{sol)+( <0,i){dn)0}sol
    j=.1
    while. j<tam do.
      sol=.((((<j-1){sol)+( <j,i){dn)<.(j{sol)+( <j,i){dn))j}sol
      j=.j+1
    end.
    j=.tam-2
    while. j>:0 do.
      sol=.((j{sol)<.(j+1){sol)+( <j,i){dn)j}sol
      j=.j-1
    end.
    i=.i-1
  end.
  <./sol
)

load 'c:/p/quarta/euler82.ijs'
euler82''
260324
```

Capítulo 83

Problema 83 - Soma do caminho: 4 caminhos

NOTA: Este problema é uma versão significativamente mais desafiadora do Problema 81 .

Na matriz abaixo, 5×5 a soma mínima do caminho do canto superior esquerdo para o canto inferior direito, movendo para a esquerda, direita, para cima e para baixo, é indicada em vermelho negrito e é igual a 2297.

$$\begin{pmatrix} 131 & 673 & 234 & 103 & 18 \\ 201 & 96 & 342 & 965 & 150 \\ 630 & 803 & 746 & 422 & 111 \\ 537 & 699 & 497 & 121 & 956 \\ 805 & 732 & 524 & 37 & 331 \end{pmatrix}$$

Encontre a soma mínima do caminho do canto superior esquerdo para o canto inferior direito movendo para a esquerda, direita, para cima e para baixo em matrix.txt (clique com o botão direito e "Salvar link/destino como..."), um arquivo de texto de 31K contendo uma matriz 80×80 .

83.1 Solução em APL

Função obtida do WS VIVX831, que implementa o algoritmo A* para este problema.

```

      ∇r←astar e;f1;f2;f3;f4;h;i;inf;j;la;menor;ocara;t;zica;ult;uba
[1]  inf←10000000000
[2]  h←(pe)p[/,e
[3]  t←1↑pe
[4]  i←t
[5]  t1:→(i<1)/fim
[6]  j←t
[7]  t2:→(j<1)/oba
[8]  h[i;j]←h[i;j]×-1+(t-i-1)+t-j-1
[9]  j←j-1
[10] →t2
[11] oba:i←i-1
[12] →t1
[13] fim:ULTH←h
[14] la←0 5p0
[15] la←la,[1]1 1,e[1;1],(e[1;1]+h[1;1]),-1
[16] A la = LIN, COL, SOMA, CUSTO, PAI
[17] comeco:
[18] menor←1↑(la[;4]εzica←[/la[;4])/[1]1↑pla
[19] ocara←,la[menor;]
[20] la[menor;4]←la[menor;4]+inf
[21] f1←ocara[12]+-1 0 A acima
[22] f2←ocara[12]+0 1 A direita
[23] f3←ocara[12]+1 0 A baixo
[24] f4←ocara[12]+0 -11 A esquerda
[25] →((0=[/f1)∨(t<[/f1))/pula1
[26] →(∨/2=+/la[;1 2]=(((1↑pla),2)p f1))/pula1
[27] la←la,[1]1 5p f1,(ocara[3]+e[f1[1];f1[2]]),(ocara[3]+e[f1[1];f1[2]]+h[f1[1
```

```

];f1[2])),menor
[28] →(2=+/(2pt)=f1)/orra
[29] pula1:
[30] →((0=[/f2)∨(t<[/f2))/pula2
[31] →(∨/2=+/la[;1 2]=(((1tp1a),2)pf2))/pula2
[32] la←la,[1]1 5pf2,(ocara[3]+e[f2[1];f2[2]]),(ocara[3]+e[f2[1];f2[2]]+h[f2[1];f2[2]]),menor
[33] →(2=+/(2pt)=f2)/orra
[34] pula2:
[35] →((0=[/f3)∨(t<[/f3))/pula3
[36] →(∨/2=+/la[;1 2]=(((1tp1a),2)pf3))/pula3
[37] la←la,[1]1 5pf3,(ocara[3]+e[f3[1];f3[2]]),(ocara[3]+e[f3[1];f3[2]]+h[f3[1];f3[2]]),menor
[38] →(2=+/(2pt)=f3)/orra
[39] pula3:
[40] →((0=[/f4)∨(t<[/f4))/pula4
[41] →(∨/2=+/la[;1 2]=(((1tp1a),2)pf4))/pula4
[42] la←la,[1]1 5pf4,(ocara[3]+e[f4[1];f4[2]]),(ocara[3]+e[f4[1];f4[2]]+h[f4[1];f4[2]]),menor
[43] →(2=+/(2pt)=f4)/orra
[44] pula4:
[45] →comeco
[46] orra:
[47] # '-----'
[48] # 1 numera la
[49] # '-----'
[50] LA←la
[51] ult←1tp1a
[52] CAM←0 2p0
[53] r←la[ult;3]
[54] tg:→(ult=-1)/fim33
[55] CAM←CAM,[1]1 2pla[ult;12]
[56] ult←la[ult;5]
[57] →tg
[58] fim33:
[59] r←r,1tpCAM
[60] →0

```

83.2 Solução em Python

retirado do usuário jccchurch do forum de soluções do problema. Devidamente adaptado.

```

with open('c:/p/quarta/0083_matrix.txt', 'r') as f:
    l=[[int(num) for num in line.split(',')]] for line in f]
print(len(l))

def solve83(matrix):
    grid = []
    height = len(matrix)
    width = len(matrix[0])
    for i in range(height):
        grid += [ [-1] * width ]
    grid[-1][-1] = matrix[-1][-1]
    queue = [(grid[-1][-1], height-2,width-1),(grid[-1][-1], height-1,width-2)]
    while len(queue) > 0:
        dist, i,j = queue.pop(0)
        if grid[ i][j] >= 0: continue
        grid[ i][j] = matrix[ i][j] + dist
        for m,n in [(i-1,j),(i+1,j),(i,j-1),(i,j+1)]:
            if m >= 0 and m < height and n >= 0 and n < width:

```

```
        if grid[m][n] == -1:
            queue += [(grid[i][j],m,n)]
        queue.sort()
    return grid[0][0]
print(solve83(l))
```

```
80
425185
```

Capítulo 84

Problema 84 - Probabilidades do banco imobiliário

No jogo Banco Imobiliário, o tabuleiro padrão é configurado da seguinte maneira:

GO	A1	CC1	A2	T1	R1	B1	CH1	B2	B3	JAIL
H2										C1
T2										U1
H1										C2
CH3										C3
R4										R2
G3										D1
CC3										CC2
G2										D2
G1										D3
G2J	F3	U2	F2	F1	R3	E3	E2	CH2	E1	FP

Um jogador começa no quadrado GO e soma as pontuações em dois dados de 6 lados para determinar o número de quadrados que ele avança em sentido horário. Sem nenhuma regra adicional, esperaríamos visitar cada quadrado com a mesma probabilidade: 2,5%. No entanto, pousar em G2J (Go To Jail), CC (community chest) e CH (chance) muda essa distribuição.

Além de G2J, e uma carta de cada CC e CH, que ordena que o jogador vá diretamente para a prisão, se um jogador rolar três duplas consecutivas, ele não avança o resultado de sua terceira rolagem. Em vez disso, ele prossegue diretamente para a prisão.

No início do jogo, as cartas CC e CH são embaralhadas. Quando um jogador pausa em CC ou CH, ele pega uma carta do topo da respectiva pilha e, após seguir as instruções, ela é devolvida ao fundo da pilha. Há dezesseis cartas em cada pilha, mas para o propósito deste problema estamos preocupados apenas com cartas que ordenam um movimento; qualquer instrução não relacionada ao movimento será ignorada e o jogador permanecerá no quadrado CC/CH.

- Baú comunitário (2/16 cartas):
 - Avançar para IR
 - Vá para a CADEIA
- Chance (10/16 cartas):
 - Avançar para IR
 - Vá para a CADEIA
 - Vá para C1
 - Ir para E3
 - Vá para H2
 - Vá para R1
 - Ir para a próxima R (empresa ferroviária)
 - Ir para o próximo R
 - Vá para a próxima U (empresa de serviços públicos)
 - Volte 3 quadrados.

O cerne deste problema diz respeito à probabilidade de visitar um quadrado em particular. Ou seja, a probabilidade de terminar naquele quadrado após uma jogada. Por esta razão, deve ficar claro que, com exceção de G2J para o qual a probabilidade de terminar nele é zero, os quadrados CH terão as menores probabilidades, pois 5/8 solicitam um movimento para outro quadrado, e é o quadrado final em que o jogador termina em cada jogada que estamos interessados. Não faremos distinção entre "Apenas Visitando" e ser enviado para a CADEIA, e também ignoraremos a regra sobre exigir um duplo para "sair da cadeia", assumindo que eles paguem para sair em seu próximo turno.

Começando em GO e numerando os quadrados sequencialmente de 00 a 39, podemos concatenar esses números de dois dígitos para produzir strings que correspondem a conjuntos de quadrados.

Estatisticamente, pode-se mostrar que os três quadrados mais populares, em ordem, são JAIL (6,24%) = Quadrado 10, E3 (3,18%) = Quadrado 24 e GO (3,09%) = Quadrado 00. Portanto, esses três quadrados mais populares podem ser listados com a sequência modal de seis dígitos: 102400.

Se, em vez de usar dois dados de 6 lados, forem usados dois dados de 4 lados, encontre a sequência modal de seis dígitos.

A resposta é 101524

84.1 Solução em Python

Solução obtida do site <https://www.nayuki.io/page/project-euler-solutions> devidamente adaptado.

```
import random
def compute():
    TRIALS = 10**7
    visitcounts = [0] * 40
    chance = CardDeck(16)
    communitychest = CardDeck(16)
    consecutivedoubles = 0
    location = 0
    for i in range(TRIALS):
        # Roll tetrahedral dice
        die0 = random.randint(1, 4)
        die1 = random.randint(1, 4)
        consecutivedoubles = (consecutivedoubles + 1) if (die0 == die1) else 0
        if consecutivedoubles < 3:
            location = (location + die0 + die1) % 40
        else:
            location = 30
            consecutivedoubles = 0
        # Process actions for some locations
        if location in (7, 22, 36): # Chance
            card = chance.next_card()
            if card == 0: location = 0
            elif card == 1: location = 10
            elif card == 2: location = 11
            elif card == 3: location = 24
            elif card == 4: location = 39
            elif card == 5: location = 5
            elif card in (6, 7): # Next railway
                location = (location + 5) // 10 % 4 * 10 + 5
            elif card == 8: # Next utility
                location = 28 if (12 < location < 28) else 12
            elif card == 9:
                location -= 3
            else:
                pass
        elif location == 30: # Go to jail
            location = 10
        else:
            pass
        if location in (2, 17, 33): # Community chest
            card = communitychest.next_card()
            if card == 0: location = 0
            elif card == 1: location = 10
        visitcounts[location] += 1
```

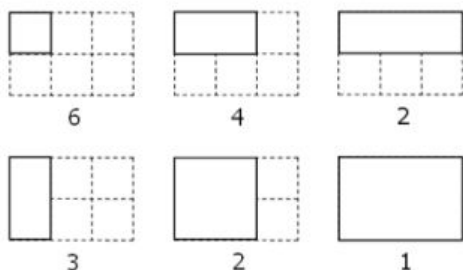
```
temp = sorted(enumerate(visitcounts), key=(lambda ic: -ic[1]))
ans = "".join(f"{i:02}" for (i, c) in temp[ : 3])
return str(ans)
class CardDeck:
    def __init__(self, size):
        self.cards = list(range(size))
        self.index = size
    def next_card(self):
        if self.index == len(self.cards):
            random.shuffle(self.cards)
            self.index = 0
        result = self.cards[self.index]
        self.index += 1
        return result
print(compute())
```

101524

Capítulo 85

Problema 85 - Contando retângulos

Contando cuidadosamente, pode-se ver que uma grade retangular medindo 3×2 contém dezoito retângulos:



Embora não exista uma grade retangular que contenha exatamente dois milhões de retângulos, encontre a área da grade com a solução mais próxima.

A resposta é 2772, que corresponde às dimensões 36 por 77 e que contém 1.999.998 retângulos. (Dois a menos que o objetivo).

85.1 Solução em Python

Solução obtida do usuário tzaman no fórum de soluções do problema, adaptada.

```
def rect(n,m):
    return n*(n+1)*m*(m+1)/4
target = 2000000
n = 1
m = 2000
minD = target
minN = 0
minM = 0
while(m>n):
    d = abs(rect(n,m)-target)
    if minD>d:
        minD = d
        minN = n
        minM = m
    if rect(n,m)>target:
        m = m - 1
    else:
        n = n + 1
print(minD, ":", minN, "*", minM, "=", minN*minM)
```

2.0 : 36 * 77 = 2772

85.2 Solução em J

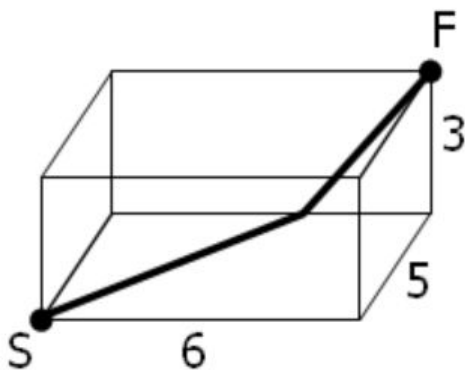
```
rect=: 4 : 0
  0.25*x*(x+1)*y*y+1
)
euler85=: 3 : 0
  target=.2000000
  n=. 1
  m=. 2000
  mind=.target
  minn=.0
  minm=.0
  while.m>n do.
    d=.(n rect m)-target
    if. mind>d do.
      mind=.d
      minn=.n
      minm=.m
    end.
    if.(n rect m)>target do.
      m=.m-1
    else.
      n=.n+1
    end.
  end.
  mind,minn,minm,minn*minm

load 'c:/p/quarta/euler85.ijs'
euler85''
2 36 77 2772
```

Capítulo 86

Problema 86 - Rota cubóide

Uma aranha, S , está sentada em um canto de uma sala cubóide, medindo 6, e uma mosca, F , fica no canto oposto. Ao viajar nas superfícies da sala, a distância mais curta em "linha reta" de S a F é 10 e o caminho é mostrado no diagrama.



No entanto, há até três candidatos ao caminho "mais curto" para qualquer cuboide dado e a rota mais curta nem sempre tem comprimento inteiro.

Pode-se demonstrar que existem exatamente 2060 cubóides distintos, ignorando rotações, com dimensões inteiras, até um tamanho máximo de $M \times M \times M$, para o qual a rota mais curta tem comprimento inteiro quando $M = 100$. Este é o menor valor de M para o qual o número de soluções excede primeiro dois mil; o número de soluções quando $M = 99$ é 1975.

Encontre o menor valor de M de modo que o número de soluções primeiro exceda um milhão. A resposta é 1818.

86.1 Solução em Python

```
import math, itertools
def compute():
    solutions = []
    def generate_solutions():
        for s in itertools.count(3, 2):
            for t in range(s - 2, 0, -2):
                if s * s // 2 >= limit * 3:
                    return
                if math.gcd(s, t) == 1:
                    for k in itertools.count(1):
                        a = s * t * k
                        b = (s * s - t * t) // 2 * k
                        c = (s * s + t * t) // 2 * k
                        if a >= limit and b >= limit:
                            break
                        find_splits(a, b, c)
                        find_splits(b, a, c)
    def find_splits(a, b, c):
        z = b
```

```

for x in range(1, a):
    y = a - x
    if y < x:
        break
    if c * c == min(
        (x + y) * (x + y) + z * z,
        (y + z) * (y + z) + x * x,
        (z + x) * (z + x) + y * y):
        temp = max(x, y, z)
        if temp < limit:
            item = tuple(sorted((x, y, z)))
            solutions[temp].add(item)
cumulativesolutions = [0]
limit = 1
while True:
    while len(solutions) < limit:
        solutions.append(set())
    generate_solutions()
    for i in range(len(cumulativesolutions), limit):
        sum = cumulativesolutions[i - 1] + len(solutions[i])
        cumulativesolutions.append(sum)
        if sum > 1000000:
            return str(i)
    limit = limit * 2
print(compute())

```

1818

Capítulo 87

Problema 87 - Triplas potências de primos

No original conforme o site projecteuler.net

The smallest number expressible as the sum of a prime square, prime cube, and prime fourth power is 28. In fact, there are exactly four numbers below fifty that can be expressed in such a way:

$$28 = 2^2 + 2^3 + 2^4 33 = 3^2 + 2^3 + 2^4 49 = 5^2 + 2^3 + 2^4 47 = 2^2 + 3^3 + 2^4$$

How many numbers below fifty million can be expressed as the sum of a prime square, prime cube, and prime fourth power?

Answer: 1097343

Potências triplas de primos O menor número que pode ser expresso como uma soma de um primo ao quadrado, um primo ao cubo e um primo à quarta potência é 28. Há exatos 4 números abaixo de 50 que podem ser expressos dessa forma:

$$28 = 2^2 + 2^3 + 2^4$$

$$33 = 3^2 + 2^3 + 2^4$$

$$49 = 5^2 + 2^3 + 2^4$$

$$47 = 2^2 + 3^3 + 2^4$$

Quantos números abaixo de 50.000.000 podem ser expressos como a soma de um primo ao quadrado, ao cubo e à quarta potência ?

87.1 Solução em APL

```
∇euler87 lim;p12;p13;p14;i;j;k;aa;e1;e2;a1;qual;so
[1]  A este cara precisa 40MB de workspace. Chamar:
[2]  A apl2win -ws 40M numa janela DOS
[3]  qual←(lim÷8)ρ0
[4]  p14←p13←p12←PRIMO[ι908]
[5]  p12←p12*2
[6]  p13←p13*3
[7]  p14←p14*4
[8]  p12←(p12<lim)/p12
[9]  p13←(p13<lim)/p13
[10] p14←(p14<lim)/p14
[11] i←1
[12] t1:→(i>ρp14)/fim
[13] j←1
[14] t2:→(j>ρp13)/fi8
[15] k←1
[16] t3:→(k>ρp12)/fi9
[17] →(lim<aa←p12[k]+p13[j]+p14[i])/fi9
[18] A p12[k],p13[j],p14[i],aa
[19] e1←1+[aa÷8]
[20] e2←1+8|aa
[21] a1←2 2 2 2 2 2 2 2qual[e1]
[22] a1[e2]←1
```

```

[23] qual[e1]←2 2 2 2 2 2 2 2 1a1
[24] oba:k←k+1
[25] →t3
[26] fi9:
[27] j←j+1
[28] →t2
[29] fi8:
[30] i←i+1
[31] →t1
[32] fim:
[33] i←1
[34] so←0
[35] t6:→(i>pqual)/fim9
[36] so←so++/2 2 2 2 2 2 2 2 2τqual[i]
[37] i←i+1
[38] →t6
[39] fim9:so

```

```

euler87 50000000
1097343

```

Problema 88 - Números soma e produto

No original conforme o site projecteuler.net

A natural number, N , that can be written as the sum and product of a given set of at least two natural numbers, $a_1, a_2, \dots, a_k$ is called a product-sum number: $N = a_1 + a_2 + \dots + a_k = a_1 \times a_2 \times \dots \times a_k$.

For example, $6 = 1 + 2 + 3 = 1 \times 2 \times 3$.

For a given set of size, k , we shall call the smallest N with this property a minimal product-sum number. The minimal product-sum numbers for sets of size, $k = 2, 3, 4, 5$, and 6 are as follows.

$k=2$: $4 = 2 \times 2 = 2 + 2$ $k=3$: $6 = 1 \times 2 \times 3 = 1 + 2 + 3$ $k=4$: $8 = 1 \times 1 \times 2 \times 4 = 1 + 1 + 2 + 4$ $k=5$: $8 = 1 \times 1 \times 2 \times 2 \times 2 = 1 + 1 + 2 + 2 + 2$ $k=6$: $12 = 1 \times 1 \times 1 \times 1 \times 2 \times 6 = 1 + 1 + 1 + 1 + 2 + 6$

Hence for $2 \leq k \leq 6$, the sum of all the minimal product-sum numbers is $4 + 6 + 8 + 12 = 30$; note that 8 is only counted once in the sum.

In fact, as the complete set of minimal product-sum numbers for $2 \leq k \leq 12$ is $4, 6, 8, 12, 15, 16$, the sum is 61 .

What is the sum of all the minimal product-sum numbers for $2 \leq k \leq 12000$?

Answer: 7587457 Completed on Fri, 30 May 2014, 16:02

Cujo enunciado é:

Um número natural, N , que pode ser escrito como a soma e o produto de um conjunto dado de ao menos dois números naturais, $\{a_1, a_2, \dots, a_k\}$ se denomina número produto-soma: $N = a_1 + a_2 + \dots + a_k = a_1 \times a_2 \times \dots \times a_k$.

Por exemplo, $6 = 1 + 2 + 3 = 1 \times 2 \times 3$.

Para um conjunto dado seu tamanho, k , vamos chamar ao menor N com esta propriedade um número produto-soma mínimo. Os mínimos números produto-soma para os conjuntos de tamanho, $k = 2, 3, 4, 5$ e 6 são os seguintes.

$k = 2$: $4 = 2 \times 2 = 2 + 2$

$k = 3$: $6 = 1 \times 2 \times 3 = 1 + 2 + 3$

$k = 4$: $8 = 1 \times 1 \times 2 \times 4 = 1 + 1 + 2 + 4$

$k = 5$: $8 = 1 \times 1 \times 2 \times 2 \times 2 = 1 + 1 + 2 + 2 + 2$

$k = 6$: $12 = 1 \times 1 \times 1 \times 1 \times 2 \times 6 = 1 + 1 + 1 + 1 + 2 + 6$

Disso para $2 \leq k \leq 6$, a soma de todos os números produto-soma mínimos seja $4 + 6 + 8 + 12 = 30$; veja que 8 só se conta uma vez na soma.

De fato, como o conjunto completo de números produto-soma mínimos para $2 \leq k \leq 12$ é $\{4, 6, 8, 12, 15, 16\}$, a soma é 61 .

Qual é a soma de todos os números produto-soma mínimos para $2 \leq k \leq 12000$?

Para bem resumir, para um determinado conjunto de cardinalidade k o produto-soma mínimo deve ter k parcelas e k fatores.

Tive um insight de que um produto deste tipo para o conjunto de cardinalidade k sempre pode ser encontrado fazendo a multiplicação $k \times 2 \times 1 \times 1 \times \dots = 2k$ cujo resultado será $2k$. Já a soma do mesmo conjunto de cardinalidade k pode ser obtida fazendo $1 + 1 + 1 \dots + 1 = k$, isto é somando k vezes a parcela 1 . Então, se o produto soma mínimo existir para um determinado conjunto ele estará entre k e $2k$.

A coisa não foi muito além, porque tive dificuldade de como testar todas as combinações possíveis de parcelas e de fatores na busca do mesmo resultado.

Daí fui buscar ajuda no <http://www.mathblog.dk/> e vejamos o que veio de lá: Se você tem um conjunto de fatores, digamos $\{2, 3, 4\}$, pode-se fazer $2 \times 3 \times 4 = 24$ e também pode-se fazer $2 + 3 + 4 = 9$. Este resultado não é o que esperamos encontrar, mas adicionando uns, altera-se a soma, mas não o produto. Se somarmos $24 - 9 = 15$ uns vamos obter o produto-soma com 18 parcelas/fatores. Este procedimento pode ser generalizado para qualquer caso. Agora, se ele é ou não é mínimo, aí são outros quinhentos.

O segundo insight do site é o mesmo que eu já havia enunciado acima garantindo que o produto-soma mínimo para $psm(k)$ está entre $k \leq psm(k) \leq 2k$.

A solução

Precisamos fatorar todos os números entre 2 e 24000 checando se essa fatoração é um produto-soma mínimo de algum k . Mas, não é fácil fazer esta fatoração.

Daí ele usou outra abordagem: gerar todas as fatorações com 2 fatores, a saber:

$2 \times 2, 2 \times 3, \dots 2 \times 12000$

$3 \times 3, 3 \times 4, \dots 3 \times 8000$

e assim por diante até achar $\sqrt{12000} \times \sqrt{12000} \approx 110 \times 110$.

Depois disso, o mesmo para 3 fatores:

$2 \times 2 \times 2, 2 \times 2 \times 3, \dots 2 \times 2 \times 6000$

Assim por diante até atingir 14 fatores já que $\lfloor \log_2(24000) \rfloor = 14$.

A maneira fácil de fazer é ter 14 loops (for) um dentro do outro, mas ele buscou uma solução mais genérica e criou um loop while com um contador para saltar para cima e para baixo no array para incrementar os fatores na medida do necessário. Daí eu não consegui entender o C++ dele. Fui atrás de outra solução na Internet e achei esta que está aí embaixo. Originalmente em C++, funcionou no CodeBlocks e daí fiquei naquela pindaíba para traduzir para C. Finalmente deu certo, em C escoreito. Ficou assim:

88.1 Solução em C++

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int minK[12001];
int factors[16];
void findMinK(int product, int upperBound, int level) {
    int i,j;
    if (level > 15) return;
    int ones;
    for (i = 2; i <= upperBound; ++i)
    {
        if (product * i > 24000) break;
        else ones = product * i;
        factors[level] = i;
        for (j = 1; j <= level; ++j) ones -= factors[j];
        if (ones + level <= 12000 && product * i < minK[ones + level])
            minK[ones + level] = product * i;
        findMinK(product * i, 24000 / (product * i), level + 1);
    }
}
int main() {
    int i,achou,j;
    for (i=0;i<=12000;i++) {
        minK[i] = 16000000;
    }
    for (i=0;i<16;i++) {
        factors[i]=0;
    }
    findMinK(1, 24000, 1);
    long int answer = 0;
    for (i=2;i<=12000;i++) {
        achou=0;
        for (j=2;j<i;j++){ // eliminando os duplos
            if (minK[i]==minK[j]) {
                achou=1;
                break;
            }
        }
        if (achou==0) {
            answer = answer + minK[i];
        }
    }
}
```

```

    printf("resposta = %d\n", answer);
    return 0;
}

```

Daí fiz a função em APL2 (funções euler88, findMinK88 e variável minK) e cheguei ao mesmo resultado (depois de umas 4 horas buscando entender o C++ e sendo enganado pela inicialização do minK. Olhando o código parece que é 1, mas na verdade é infinito.

Tanto o C quanto o APL dão a resposta correta: 7.587.457.

88.2 Solução em APL

```

    ∇r←euler88;x
[1] minK88←12001ρ16843009
[2] factors←16ρ0
[3] findMinK88 1 24000 1
[4] x←minK88[2+⌊11999]
[5] +/((x⌊x)=⌊px)/x

    ∇findMinK88 ji;product;upperBound;level;i;ones;j;⌈IO
[1] ⌈IO←0
[2] product←ji[0]
[3] upperBound←ji[1]
[4] level←ji[2]
[5] →(level>15)/0
[6] i←2
[7] t1:→(i>upperBound)/0
[8] →((product×i)>24000)/0
[9] ones←product×i
[10] factors[level]←i
[11] j←1
[12] t2:→(j>level)/prox
[13] ones←ones-factors[j]
[14] j←j+1
[15] →t2
[16] prox:→((ones+level)>12000)/viva
[17] →((product×i)≥minK88[ones+level])/viva
[18] minK88[ones+level]←product×i
[19] viva:findMinK88(product×i),(⌊24000÷product×i),level+1
[20] i←i+1
[21] →t1
[22]

    euler88
7587457

```

Capítulo 89

Problema 89 - Numerais romanos

No original conforme o site projecteuler.net

For a number written in Roman numerals to be considered valid there are basic rules which must be followed. Even though the rules allow some numbers to be expressed in more than one way there is always a "best" way of writing a particular number.

For example, it would appear that there are at least six ways of writing the number sixteen:

```
IIIIIIIIIIIIIIII
VIIIIIIIIIII
VVIIIIII
XIIIIIII
VVVI
XVI
```

However, according to the rules only XIIIIII and XVI are valid, and the last example is considered to be the most efficient, as it uses the least number of numerals.

The 11K text file, `roman.txt` (right click and 'Save Link/Target As...'), contains one thousand numbers written in valid, but not necessarily minimal, Roman numerals; see About... Roman Numerals for the definitive rules for this problem.

Find the number of characters saved by writing each of these in their minimal form.

Note: You can assume that all the Roman numerals in the file contain no more than four consecutive identical units.

Numerais romanos Para que um número escrito em algarismos romanos seja considerado válido, há regras básicas que devem ser seguidas. Embora as regras permitam que alguns números sejam expressos de mais de uma maneira, sempre há uma "melhor" maneira de escrever um número em particular.

Por exemplo, parece que há pelo menos seis maneiras de escrever o número dezesseis:

```
IIIIIIIIIIIIIIII VIIIIIIIII VVIIIIII XIIIIIII VVVI XVI
```

Entretanto, de acordo com as regras, apenas XIIIIII e XVI são válidos, e o último exemplo é considerado o mais eficiente, pois utiliza o menor número de numerais.

O arquivo de texto 11K, `roman.txt` (clique com o botão direito e selecione "Salvar link/destino como..."), contém mil números escritos em numerais romanos válidos, mas não necessariamente mínimos; consulte Sobre... numerais romanos para obter as regras definitivas para este problema.

Descubra o número de caracteres economizados escrevendo cada um deles em sua forma mínima.

Observação: você pode presumir que todos os algarismos romanos no arquivo não contêm mais do que quatro unidades idênticas consecutivas Answer: 743

89.1 Solução em APL

A variável `rom89` contém os dados lidos do arquivo citado na definição.

```
⍵euler89;x;som;i;z;ta1;ta2;z2
[1] x←rom89
[2] som←0
[3] i←1
[4] t1:→(i>1000)/fim
[5] z←r2a89 trim x[i;]
[6] ta1←ptrim x[i;]
[7] z2←a2r89 z
```

```
[8]   ta2←ptrim z2
[9]   som←som+(ta1-ta2)
[10]  i←i+1
[11]  →t1
[12] fim:som
```

Capítulo 90

Problema 90 - Pares de dígitos do cubo

Cada uma das seis faces de um cubo tem um dígito diferente (0 a 9) escrito nele; o mesmo é feito para um segundo cubo. Ao colocar os dois cubos lado a lado em posições diferentes, podemos formar uma variedade de números de 2-dígitos.

Por exemplo, o número quadrado 64 poderia ser formado:



De fato, escolhendo cuidadosamente os dígitos em ambos os cubos, é possível exibir todos os números quadrados abaixo de cem: 01,04,09,16,25,36,49,64 e 81. Por exemplo, uma maneira de conseguir isso é colocando $\{0, 5, 6, 7, 8, 9\}$ em um cubo e $\{1, 2, 3, 4, 8, 9\}$ no outro. No entanto, para este problema, permitiremos que 6 ou 9 sejam virados de cabeça para baixo para que um arranjo como $\{0, 5, 6, 7, 8, 9\}$ e $\{1, 2, 3, 4, 6, 7\}$ permite que todos os nove números quadrados sejam exibidos; caso contrário, seria impossível obter 09. Ao determinar um arranjo distinto, estamos interessados nos dígitos de cada cubo, não na ordem. $\{1, 2, 3, 4, 5, 6\}$ é equivalente a $\{3, 6, 4, 1, 2, 5\}$ $\{1, 2, 3, 4, 5, 6\}$ é distinto de $\{1, 2, 3, 4, 5, 9\}$

Mas porque estamos permitindo que 6 e 9 sejam revertidos, os dois conjuntos distintos no último exemplo representam ambos o conjunto estendido com a finalidade de formar números de 2-dígitos. Quantos arranjos distintos dos dois cubos permitem que todos os números quadrados sejam exibidos?

A resposta é 1217.

90.1 Solução em Python

solução obtida e adaptada de <https://www.ivl-projecteuler.com/overview-of-problems/40-difficulty/problems>

```
def dicecomb(): #Produces all dice combinations, there are 10 C 6 = 210
    dicecombs = set()
    for a in range(0,10):
        for b in range(0,9):
            for c in range(0,8):
                for d in range(0,7):
                    for e in range(0,6):
                        for f in range(0,5):
                            if len(set([a,b,c,d,e,f])) == 6:
                                dicecombs.add(tuple(sorted((a,b,c,d,e,f))))
    return list(dicecombs)
def valid_dice_pair(dice1, dice2):
    square_numbers = [(0,1), (0,4), (0,9), (1,6), (2,5), (3,6), (4,9), (6,4), (8,1)]
    squares_generated = set()
    for x in dice1:
        for y in dice2:
            if x == 6 or x == 9:
                squares_generated.add((6,y))
                squares_generated.add((9,y))
            squares_generated.add((y,9))
```

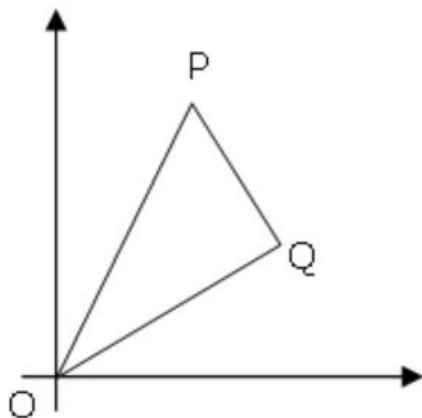
```
        squares_generated.add((y,6))
    elif y == 6 or y == 9:
        squares_generated.add((6,x))
        squares_generated.add((9,x))
        squares_generated.add((x,9))
        squares_generated.add((x,6))
    else:
        squares_generated.add((x,y))
        squares_generated.add((y,x))
for x in square_numbers:
    if x not in squares_generated:
        return False
return True
def compute():
    dice = dicecomb()
    count = 0
    for x in range(len(dice)):
        for y in range(x+1,len(dice)):
            if valid_dice_pair(dice[x], dice[y]):
                count = count + 1
    return count
print(compute())
```

1217

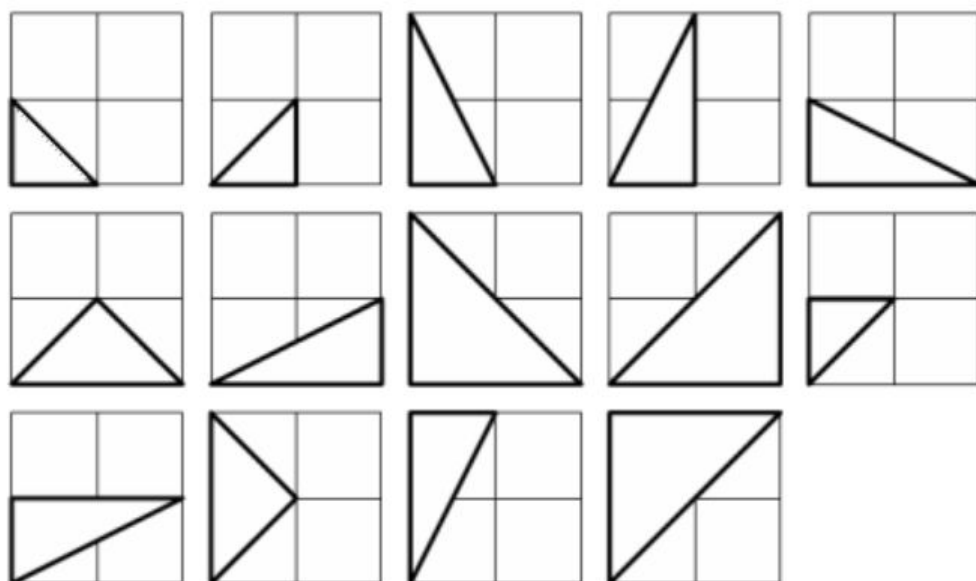
Capítulo 91

Problema 91 - Triângulos retângulos com coordenadas inteiras

Cujo enunciado é: Os pontos $P(x_1, y_1)$ e $Q(x_2, y_2)$ estão representados por coordenadas inteiras e se unem com a origem $O(0,0)$ para formar o triângulo OPQ .



Existem exatamente 14 triângulos com um ângulo reto que se formam quando as coordenadas se encontram entre 0 e 2 inclusive, ou seja $0 \leq x_1, y_1, x_2, y_2 \leq 2$.

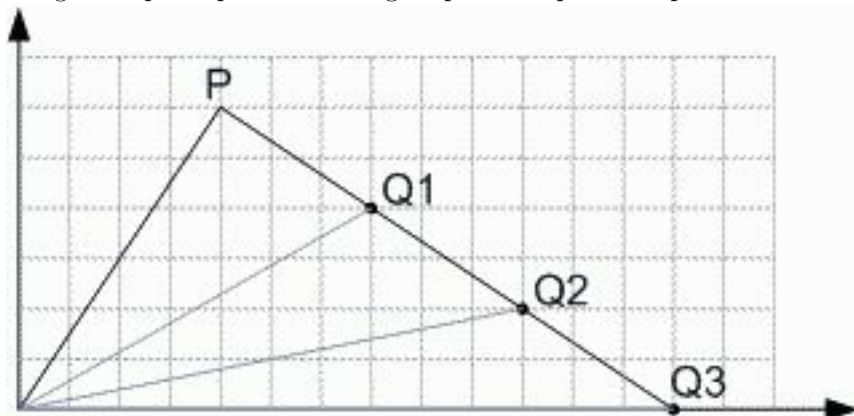


Dados $0 \leq x_1, y_1, x_2, y_2 \leq 50$ quantos triângulos retângulos podem ser formados ?

Estratégia de solução

caso geral

O caso geral supõe o ponto P em algum ponto do primeiro quadrante. No exemplo,



supõe-se P em $(4, 6)$. A inclinação da reta OP é $6/4$. Desenhando uma ortogonal (por causa do triângulo reto) por este ponto, sua inclinação é $-4/6$ ou simplificando $-2/3$. Isto significa que a cada 3 unidades andadas no eixo x , se cai 2 unidades no eixo y . Como a partida foi sobre um número inteiro, a cada etapa destas, vai-se estar em um inteiro também. Esta regra pode continuar até uma destas coisas ocorrer (a primeira delas)

- o valor máximo de x ser alcançado OU
- cruzar-se o eixo x (ou seja o valor de y ser zero).

No exemplo acima, o máximo é entre $\frac{x_{max}-4}{3}$ e $\frac{6}{2}$.

Definindo-se os incrementos como $d_x = \frac{y_1}{mdc(x_1, y_1)}$ e $d_y = \frac{x_1}{mdc(x_1, y_1)}$ a regra original deve ser: soma-se o mínimo entre

$$\frac{x_{max} - x_1}{d_x} \quad e \quad \frac{y_1}{d_y}$$

Substituindo e cozinhando as fórmulas, deve-se achar o mínimo entre

$$\frac{(x_{max} - x_1) \times mdc(x_1, y_1)}{y_1} \quad e \quad \frac{y_1 \times mdc(x_1, y_1)}{x_1}$$

casos regulares

1. Ângulo reto em O Para cada ponto inteiro em x que receba P podem ser criados x_{max} triângulos variando Q em todos os pontos do eixo y . No total, isto gera x_{max}^2 triângulos.
2. P no eixo x Para cada ponto P inteiro em x , levanta-se uma vertical de comprimento $1, 2, \dots, x_{max}$ e conecta-se a extremidade dessa vertical com a origem. Está criado um triângulo retângulo (ângulo reto em P). No total, isto gera x_{max}^2 triângulos.
3. P no eixo y Idênticas considerações ao item anterior. No total, isto gera x_{max}^2 triângulos.

Em resumo, os casos regulares são $3 \times x_{max}^2$.

Solução

Tirada do site <http://www.mathblog.dk/> diz:

```
int size = 50;
int result = size*size*3;
for (int x = 1; x <= size; x++) {
    for (int y = 1; y <= size; y++) {
        int fact = gcd(x, y);
        result += Math.Min(y*fact / x, (size - x)*fact / y) * 2;
    }
}
```

A programação do mdc, também veio de lá e é:

```

private int gcd(int a, int b) {
    int y, x;
    if (a > b) { x = a;    y = b;
    } else {     x = b;    y = a;    }
    while (x % y != 0) {
        int temp = x;    x = y;    y = temp % x;    }
    return y;
}

```

Programei isto em APL (ws: eule1315, funções euler91 e mdc91), e ela deu a resposta correta que é 14234. A programação está genérica (euler91 50) e portanto pode ser usada para gerar enunciados aleatórios e diferentes.

Uma atenção especial deve ser dada à tradução da fórmula C
result += Math.Min(y*fact /x, (size - x)*fact /y) * 2;
à sua equivalente em APL2.

91.1 Solução em APL

```

∇r←euler91 xmax;qt;i;j;m
[1] qt←xmax*2 A angulo reto na origem
[2] qt←qt+xmax*2 A triangulo com P no eixo x
[3] qt←qt+xmax*2 A triangulo com P no eixo y
[4] i←1
[5] t1:→(i>xmax)/fim
[6] j←1
[7] t2:→(j>xmax)/soma
[8] m←i mdc91 j
[9] qt←qt+2×(⌊(xmax-i)×(m÷j)⌋)⌊j×m÷i
[10] j←j+1
[11] →t2
[12] soma:i←i+1
[13] →t1
[14] fim:r←qt

∇r←a mdc91 b;x;y;t
[1] x←a⌈b
[2] y←a⌊b
[3] t11:→(0=y⌊x)/fim1
[4] t←x
[5] x←y
[6] y←x⌊t
[7] →t11
[8] fim1:r←y

euler91 50
14234

```

91.2 Solução em J

Note que aqui se usou a função MDC do J (+. diádico e não booleano).

```

euler91=: 3 : 0
    qt=.3*50^2
    for_i. 1+i.50 do.
        for_j. 1+i.50 do.
            m=. i +. j NB. mdc(i,j)
            qt =. qt+2*(<.(50-i)*(m%j))<. <. j*m%i
        end.
    end.
    qt
)

```

```
load'c:/p/quarta/euler91.ijs'  
euler91''  
14234
```

Capítulo 92

Problema 92 - Correntes de dígitos quadrados

No original conforme o site projecteuler.net

A number chain is created by continuously adding the square of the digits in a number to form a new number until it has been seen before.

For example,

$44 \rightarrow 32 \rightarrow 13 \rightarrow 10 \rightarrow 1 \rightarrow 1$

$85 \rightarrow 89 \rightarrow 145 \rightarrow 42 \rightarrow 20 \rightarrow 4 \rightarrow 16 \rightarrow 37 \rightarrow 58 \rightarrow 89$

Therefore any chain that arrives at 1 or 89 will become stuck in an endless loop. What is most amazing is that EVERY starting number will eventually arrive at 1 or 89.

How many starting numbers below ten million will arrive at 89?

Answer: 8581146 Completed on Tue, 14 May 2013, 22:27

Cadeias de dígitos quadrados Uma cadeia de números é criada se adicionarmos continuamente o quadrado dos dígitos de um número a ele formando um número ainda não aparecido. Por exemplo,

$44 \rightarrow 32 \rightarrow 13 \rightarrow 10 \rightarrow 1 \rightarrow 1$

$85 \rightarrow 89 \rightarrow 145 \rightarrow 42 \rightarrow 20 \rightarrow 4 \rightarrow 16 \rightarrow 37 \rightarrow 58 \rightarrow 89$

Qualquer cadeia que chegue a 1 ou a 89 entrará em loop. O mais incrível é que todos os números chegam a 1 ou a 89. Quantos números menores do que 10.000.000 chegam a 89 ?

92.1 Solução em APL

```
∇euler92;c;ctd
[1]  c←1
[2]  ctd←0
[3]  t1:→(c≥10000000)/aca
[7]  →(1=onde92 c)/soma
[8]  c←c+1
[9]  →t1
[10] soma:
[11] ctd←ctd+1
[12] c←c+1
[13] →t1
[14] aca:ctd
```

```
∇r←onde92 x;z;i;som
[1]  A devolve 1 se x→89 e 0 se x→1
[2]  t0:
[3]  z←,⌜x
[4]  som←0
[5]  i←1
[6]  t1:→(i>pz)/fim
[7]  som←som+(⌊z[i])2
[8]  i←i+1
[9]  →t1
[10] fim:→(som=89)/aca89
[11] →(som=1)/aca1
```

```
[12] x←som
[13] →t0
[14] aca1:r←0
[15] →0
[16] aca89:r←1
[17] →0
```

8581146

92.2 Solução em J

```
onde92=: 3 : 0
  som=.0
  while. (som~:1) *. som~: 89 do.
    yy=:y
    som=.0
    for_i. i.$yy do.
      som=.som+(".i{yy)^2
    end.
    y=.som
  end.
  som=89
)
```

```
euler92=: 3 : 0
  ctd=. 0
  for_i. 1+i.9999999 do.
    if. onde92 i do.
      ctd=.ctd+1
    end.
  end.
  ctd
)
```

```
load'c:p/quarta/euler92.ijs'
euler92''
8581146
```

Capítulo 93

Problema 93 - Expressões aritméticas

Cujo enunciado é: Usando cada um dos dígitos do conjunto {1, 2, 3, 4}, exatamente uma vez e fazendo uso das quatro operações aritméticas (+, -, *, /) e os parênteses/colchetes, é possível formar diferentes inteiros, positivos.

Por exemplo

```
8 = (4 * (1 + 3)) / 2
14 = 4 * (3 + 1 / 2)
19 = 4 * (2 + 3) - 1
36 = 3 * 4 * (2 + 1)
```

Considere que as concatenações dos dígitos como **12 + 34** não são permitidas. Usando o conjunto {1, 2, 3, 4} é possível obter trinta e um números diferentes dos 36 que seriam o máximo e cada um dos números de 1 a 28 podem ser formados antes de chegar ao primeiro que não pode ser expresso desta maneira.

Ache o conjunto de 4 dígitos inteiros, $a < b < c < d$, para os quais se obtém o conjunto mais comprido de inteiros positivos consecutivos de $1an$, dando a tua resposta como a cadeia *abcd*.

Funções

comb93 Copiei um IDIOM do APL2 que gera n combinações de k objetos (**n comb93 k**). Neste caso, ao chamar com 10 comb93 4, ele gera as combinações de 10 objetos 4 a 4. Em outras palavras ela gera todas as 210 combinações de 4 dígitos retirados de 10 (0..9). Deixo o APL aqui, pois é bonito de ver:

opes93 Gera as 24 combinações possíveis desses 4 dígitos acima selecionados. Eis o começo: 1 2 3 4, 1 2 4 3, 1 3 2 4, ..., 4 3 2 1.

lug93 Gera $4^3 = 64$ combinações das 4 operações (+-) aparecendo em 3 lugares na expressão. Eis o começo: 1 1 1, 1 1 2, 1 1 3, ... , 4 4 4.

euler93 A função integradora. Chama as 3 acima, para organizar as coisas, e daí intercala parênteses no meio da expressão para alterar a ordem em que as coisas são executadas. Também faz o controle da maior sequência para cada uma das alternativas, imprimindo ao final, a maior sequência e onde ela ocorre, ou seja em que linha da combinações de dígitos ela ocorreu. Esta é a resposta esperada pelo Euler, e deu certo.

```
[0] r←n comb93 k
[1] r←Q(⌊2<÷r)/r←1+((-k)↑1n)⌈1(!k)×k!n

[0] r←opes93;r1;r2;r3;r4
[1] r1←1 2 3 4 1 2 4 3 1 3 2 4 1 3 4 2 1 4 2 3 1 4 3 2
[2] r2←2 1 3 4 2 1 4 3 2 3 1 4 2 3 4 1 2 4 1 3 2 4 3 1
[3] r3←3 1 2 4 3 1 4 2 3 2 1 4 3 2 4 1 3 4 1 2 3 4 2 1
[4] r4←4 1 2 3 4 1 3 2 4 2 1 3 4 2 3 1 4 3 1 2 4 3 2 1
[5] r←24 4pr1,r2,r3,r4

[0] r←lug93
[1] r←64 3p0
[2] r[;1]←1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3
    3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
[3] r[;2]←1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 1 1
```

```

1 1 2 2 2 2 3 3 3 3 4 4 4 4 1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4
[4] r[;3]←1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4
3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4

```

```

[0] euler93;coma;cont;i;imax;in;inmax;j;k;lu;matu;max;op;x
[1] x←~1+10 comb93 4 A gera os 4 digitos entre 0 e 9
[2] A x←1 4p1 2 3 4 -- e o exemplo, dah=28
[3] op←opes93
[4] lu←lug93
[5] i←1
[6] max←-99999
[7] t1:→(i>1tpx)/fim
[8] cont←1000p0
[9] j←1
[10] t2:→(j>1tpop)/f5
[11] k←1
[12] t3:→(k>1tplu)/f6
[13] coma←15p' '
[14] inmax←1000
[15] coma[2 6 10 14]←(1 0px[i;])[op[j;]]
[16] coma[4 8 12]←'+-xd'[lu[k;]]
[17] A coma
[18] in←±coma
[19] →((in<1)v(in>inmax)v(in≠[in]))/j1
[20] cont[in]←1
[21] j1:
[22] coma[5 11]←'()'
[23] in←±coma
[24] →((in<1)v(in>inmax)v(in≠[in]))/j2
[25] cont[in]←1
[26] j2:
[27] coma[5 11]←' '
[28] coma[9 15]←'()'
[29] in←±coma
[30] →((in<1)v(in>inmax)v(in≠[in]))/j3
[31] cont[in]←1
[32] j3:
[33] coma[9 15]←' '
[34] coma[5 15]←'()'
[35] in←±coma
[36] →((in<1)v(in>inmax)v(in≠[in]))/j4
[37] cont[in]←1
[38] j4:
[39] coma[5 15]←' '
[40] coma[1 11]←'()'
[41] in←±coma
[42] →((in<1)v(in>inmax)v(in≠[in]))/j5
[43] cont[in]←1
[44] j5:
[45] coma[1 11]←' '
[46] coma[1 7 9 15]←'()()'
[47] in←±coma
[48] →((in<1)v(in>inmax)v(in≠[in]))/j6
[49] cont[in]←1
[50] j6:
[51] k←k+1
[52] →t3
[53] f6:j←j+1
[54] →t2
[55] f5:
[56] matu←~1+1t(~cont)/ι1000
[57] →(matu≤max)/continua
[58] matu,i,x[i;]

```

```

[59] max←matu
[60] imax←i
[61] continua:i←i+1
[62] →t1
[63] fim:('max = ',⌈max),('      abcd = ',⌈x[imax;])

```

Todas estas funções estão dentro do **WS=EULE1315** e devem ser lá consultada. A propósito não vejo como este exercício pode ser aproveitado no esquema **vivo**.

Uma possível resposta

Escolhida entre as expressões mais facilmente entendíveis. A variável inteira tem 3554 linhas.

(1 * 8)-(2 + 5)	1	(2 - 1)+(5 * 8)	41
(1 - 2)+(8 - 5)	2	(1 * 2)+(5 * 8)	42
1 +(2 * 5)- 8	3	1 + 2 +(5 * 8)	43
(2 - 1)+(8 - 5)	4	(5 + 1 d 2)* 8	44
(2 d 1)-(5 - 8)	5	(2 - 1 - 8)* 5	45
(1 * 2)*(8 - 5)	6	(8 * 1 + 5)- 2	46
(1 + 5 + 8)d 2	7	2 +(1 + 8)* 5	47
2 *(1 - 5)+ 8	8	(2 - 1 - 5)* 8	48
(1 + 2)*(8 - 5)	9	(2 + 5)*(8 - 1)	49
1 + 5 + 8 d 2	10	1 *(2 + 8)* 5	50
1 *(2 * 8)- 5	11	1 +(2 + 8)* 5	51
(1 - 2)+(5 + 8)	12	? 52, 53, 54	
(2 - 1)*(5 + 8)	13	(1 + 2 + 8)* 5	55
2 +(5 - 1)+ 8	14	1 *(2 + 5)* 8	56
(1 * 2)+(5 + 8)	15	1 +(2 + 5)* 8	57
1 + 2 + 5 + 8	16	(1 + 5)*(2 + 8)	60
(2 * 5)-(1 - 8)	17	(1 + 8)*(2 + 5)	63
(1 * 2 * 5)+ 8	18	(1 + 2 + 5)* 8	64
1 +(2 * 5)+ 8	19	2 * 5 *(8 - 1)	70
(1 d 2)*(5 * 8)	20	8 *(2 * 5)- 1	72
(1 * 2 * 8)+ 5	21	5 *(2 * 8)- 1	75
1 +(2 * 8)+ 5	22	2 *(5 * 8)- 1	78
5 +(1 + 8)* 2	23	(2 * 5 * 8)- 1	79
(1 + 5)d(2 d 8)	24	1 * 2 * 5 * 8	80
(1 + 8 d 2)* 5	25	1 + 2 * 5 * 8	81
1 * 2 *(5 + 8)	26	1 +(2 * 5 * 8)	81
1 +(8 + 5)* 2	27	2 *(1 + 8 * 5)	82
(1 + 8 + 5)* 2	28	5 *(1 + 2 * 8)	85
5 +(1 + 2)* 8	29	(1 + 2 * 5)* 8	88
1 * 5 *(8 - 2)	30	(1 + 8)*(2 * 5)	90
1 -(2 - 8)* 5	31	(1 + 5)*(2 * 8)	96
(1 + 5 - 2)* 8	32	(1 + 2)*(5 * 8)	120
(5 * 8 - 1)- 2	33		
2 - 8 *(1 - 5)	34		
(1 + 8 - 2)* 5	35		
(1 + 5)*(8 - 2)	36		
2 +(8 - 1)* 5	37		
(1 * 5 * 8)- 2	38		
(1 - 2)+(5 * 8)	39		
(2 - 1)*(5 * 8)	40		

Para entender a programação, bolei o seguinte esquema de comando a receber o “execute” no final da montagem

		op			op			op							
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
	a				b				c				d		

A resposta

é 51 inteiros contínuos (1..51), que estão ai acima descritos, e eles ocorrem com os números 1, 2, 5 e 8, que é a resposta esperada pelo Euler.

Capítulo 94

Problema 94 - Triângulos quase equiláteros

It is easily proved that no equilateral triangle exists with integral length sides and integral area. However, the almost equilateral triangle 5-5-6 has an area of 12 square units.

We shall define an almost equilateral triangle to be a triangle for which two sides are equal and the third differs by no more than one unit.

Find the sum of the perimeters of all almost equilateral triangles with integral side lengths and area and whose perimeters do not exceed one billion (1,000,000,000).

Answer: 518408346 Completed on Tue, 3 Jun 2014, 09:13

Cujo enunciado é: É fácil provar que não existe nenhum triângulo equilátero com lados de comprimento inteiro e cuja área é também inteira. No entanto, o triângulo quasi-equilátero 5-5-6 tem uma área de 12 unidades.

Define-se um triângulo quasi-equilátero como um triângulo em que dois lados são iguais e o terceiro não difere destes em mais do que uma unidade.

Ache a soma dos perímetros de todos os triângulos quasi equiláteros com lados de comprimento inteiro, área inteira e cujos perímetros não excedam a 1 bilhão (1.000.000.000).

Estratégia de solução

Programei uma força bruta em APL2 e em C, e ambas deram resultados diferentes (e consequentemente errados) motivado pela falta de precisão no manuseio de big numbers.

```
#include<stdio.h>
#include<math.h>
main() {
    double q,t,i,a,b,p,s,pp;
    q=0.0; t=0.0; i=2.0;
    while(i<=333334.0) {
        a=i;
        b=i-1;
        pp=a+b+b;
        p=pp/2;
        if (pp<=1000000000){
            s = sqrt(p*(p-a)*(p-b)*(p-b));
            if (s==floor(s)) { // aqui esta a desconfianca
                t=t+a+b+b;
            }
            printf("%f e %f\n",t,q);
            q++;
        }
        b=i+1;
        pp=a+b+b;
        p=pp/2;
        if (pp<=1000000000){
            s = sqrt(p*(p-a)*(p-b)*(p-b));
            if (s==floor(s)) {
                t=t+a+b+b;
            }
            printf("%f - %f\n",t,q);
            q++;
        }
        i++;
    }
```

```

    }
    i++;
}
printf("%f %f\n",t,q);
}

```

mathblog

Daí, fui buscar ajuda em www.mathblog.dk. Ele diz que estamos lidando com um triângulo de lado a e $b + 1$ ou $b - 1$ e começa tratando cada um destes casos de maneira diferente:

Caso $b=a+1$

Dividindo o triângulo original em 2 por uma vertical, temos dois triângulos retângulos cujos lados são h , $b/2$ e a hipotenusa é a . A área de cada um destes triângulos é $1/2.ba.h$. e a área de ambos é $b.h$.

Usando Pitágoras, para achar o valor de h , tem-se
 $(b/2)^2 + h^2 = a^2$, mas lembrando que $b = a + 1$ fica,
 $((a + 1)/2)^2 + h^2 = a^2$ e substituindo fica
 $3a^2 = 4h^2 + 2a + 1$ e multiplicando tudo por 3 fica
 $9a^2 = 12h^2 + 6a + 3$ e reorganizando e somando 1 em cada lado, fica
 $9a^2 - 6a + 1 = 12h^2 + 4$ e dividindo por 4 fica
 $\frac{9a^2 - 6a + 1}{4} = 3h^2 - 1$ e o primeiro lado é um quadrado notável e fica
 $(\frac{3a-1}{2})^2 - 3h^2 = 1$ e substituindo $\frac{3a-1}{2}$ por x e h por y fica
 $x^2 - 3y^2 = 1$ que é uma equação de Pell.

Caso $b=a-1$

Dividindo o triângulo original em 2 por uma vertical, temos dois triângulos retângulos cujos lados são h , $b/2$ e a hipotenusa é a . A área de cada um destes triângulos é $1/2.ba.h$. e a área de ambos é $b.h$.

Usando Pitágoras, para achar o valor de h , tem-se
 $(b/2)^2 + h^2 = a^2$, mas lembrando que $b = a - 1$ fica,
 $((a - 1)/2)^2 + h^2 = a^2$ e substituindo fica
 $3a^2 = 4h^2 + 2a - 1$ e multiplicando tudo por 3 fica
 $9a^2 = 12h^2 + 6a - 3$ e reorganizando e somando 1 em cada lado, fica
 $9a^2 + 6a + 1 = 12h^2 + 4$ e dividindo por 4 fica
 $\frac{9a^2 + 6a + 1}{4} = 3h^2 - 1$ e o primeiro lado é um quadrado notável e fica
 $(\frac{3a+1}{2})^2 - 3h^2 = 1$ e substituindo $\frac{3a+1}{2}$ por x e h por y fica
 $x^2 - 3y^2 = 1$ que é a mesma equação de Pell.

Pell

Da equação de Pell, pode-se obter a e h em função de x e y e fica

$$a = \frac{2x+1}{3} \text{ e } h = \frac{2x-1}{3} \text{ e } h = y.$$

Pode-se também calcular a área do triângulo que é $S = \frac{(a \pm 1)h}{2}$, ou
 $(\frac{2x \pm 1}{3} \pm 1)y/2$ ou ainda
 $S = \frac{(x \pm 2)y}{3}.$

Agora, uma vez achada uma solução para as equações de Pell, deve-se verificar se a área e o lado do triângulo são números inteiros. Caso eles sejam, esta solução é também solução do problema original.

Na teoria da equação (diofantina, é importante ressaltar) o primeiro par de x e y é $(2, 1)$. Olhando direitinho isso, fica

$x = \frac{3a \pm 1}{2} = 2$ e $y = 1$. Neste caso a pode ser $5/3$ (desprezado por não inteiro) e 1. Se $a = 1$ e $h = 1$, temos um triângulo degenerado, cuja area é 0.

Precisamos ir para o segundo elemento da solução de Pell. Usa-se aqui a relação de recorrência.

$$x_{k+1} = x_1 x_k + n y_1 y_k$$

e

$$y_{k+1} = x_1 y_k + y_1 x_k$$

arrumando isso, fica

$$x_2 = x_1x_1 + 3y_1y_1 \text{ e}$$

$y_2 = x_1y_1 + y_1x_1$ e substituindo o primeiro par fica

$$x_2 = 2.2 + 3.1.1 \text{ e}$$

$y_2 = 2.1 + 1.2$ e substituindo o segundo par fica (7,4).

Generalizando estas fórmulas, fica:

$$x_{prox} = 2 \times x_{atu} + 3 \times y_{atu} \text{ e}$$

$$y_{prox} = 2 \times y_{atu} + x_{atu}.$$

Deve-se ressaltar que na força bruta há que examinar cerca de 600 milhões de triângulos (sem contar a dificuldade de lidar com os tais big numbers). Fazendo deste jeito são apenas 30 triângulos a examinar. Uma boa redução.

Outra coisa: no código C ao usar apenas long não se obtiveram os resultados corretos. Precisou-se usar long long (2 vezes).

Com tudo isso em mente, vamos ao programa:

94.1 Solução em C++

```
#include<stdio.h>
main() {
    long long x = 2; // primeiro valor de x
    long long y = 1; // primeiro valor de y
    long long limite = 1000000000; // limite maximo do perimetro
    long long resultado = 0; // onde vamos somar a tralha
    while (1) {
        // caso b=a+1
        long long vezes3 = 2 * x - 1;
        long long areavezes3 = y * (x-2);
        if (avezes3 > limite) break;
        if ((avezes3 > 0) && (areavezes3 > 0) && ((avezes3 % 3) == 0)
            && ((areavezes3 % 3) == 0)) {
            long long a = vezes3 / 3;
            long long area = areavezes3 / 3;
            resultado = resultado + 3 * a + 1;
        }
        // caso b = a-1
        long long vezes3 = 2 * x + 1;
        long long areavezes3 = y * (x+2);

        if ((avezes3 > 0) && (areavezes3 > 0) && ((avezes3 % 3) == 0)
            && ((areavezes3 % 3) == 0)) {
            long long a = vezes3 / 3;
            long long area = areavezes3 / 3;
            resultado = resultado + (3 * a) - 1;
        }
    }

    long long proxx = (2*x)+(3*y);
    long long proxy = (2*y)+x;
    x=proxx;
    y=proxy;
    printf("x=%ld y=%ld\n",x,y);
}
printf("Soma=%ld\n",resultado);
}
```

Este programa deu a resposta certa 518.408.346. Programando este mesmo programa em APL2 (função euler94, workspace euler711) obteve-se a mesma resposta, passando por cima do pesadelo: int, long, long long.

94.2 Solução em APL

```
[0] r←euler94;x;y;px;py;a;area;av3;arv3;res;lim
[1] x←2
[2] y←1
```

```

[3] lim←1000000000
[4] res←0
[5] t1:
[6] av3←(2×x)-1
[7] arv3←y×(x-2)
[8] →(av3>lim)/fim
[9] →((av3≤0)∨(arv3≤0)∨(0≠3|av3)∨(0≠3|arv3))/pula
[10] a←av3÷3
[11] area←arv3÷3
[12] res←res+1+3×a
[13] pula:
[14] av3←(2×x)+1
[15] arv3←y×(x+2)
[16] →(av3>lim)/fim
[17] →((av3≤0)∨(arv3≤0)∨(0≠3|av3)∨(0≠3|arv3))/pula2
[18] a←av3÷3
[19] area←arv3÷3
[20] res←res+1+3×a
[21] pula2:
[22] px←(2×x)+(3×y)
[23] py←(2×y)+x
[24] y←py
[25] x←px
[26] →t1
[27] fim:r←res
[28]
[29]

```

518408346

94.3 Solução em J

```

euler94=: 3 : 0
  xx=.2x
  yy=.1x
  lim=.1000000000x
  res=.0x
  while. 1=1 do.
    vezes3=._1+2*xx
    areavezes3=.yy*(xx-2)
    if. vezes3>lim do. break. end.
    if. (avezes3>0) *. (areavezes3>0) *. (0=3|avezes3) *. 0=3|areavezes3 do.
      a=.(1x)*avezes3 % 3
      area=. areavezes3 % 3
      res=.res+1+3*a
    end.
    vezes3=. 1+2*xx
    areavezes3=. (xx+2)*yy
    if. (avezes3>0) *. (areavezes3>0) *. (0=3|avezes3) *. 0=3|areavezes3 do.
      a=.avezes3 % 3
      area=. areavezes3 % 3
      res=.res+(3*a)-1
    end.
    proxx=.(2*xx)+(3*yy)
    proxy=.(2*yy)+xx
    xx=.proxx
    yy=.proxy
  end.
  res
)

```

```
load'c:/p/quarta/euler94.ijs'  
euler94''  
518408346
```

Capítulo 95

Problema 95 - Cadeias amigáveis

The proper divisors of a number are all the divisors excluding the number itself. For example, the proper divisors of 28 are 1, 2, 4, 7, and 14. As the sum of these divisors is equal to 28, we call it a perfect number.

Interestingly the sum of the proper divisors of 220 is 284 and the sum of the proper divisors of 284 is 220, forming a chain of two numbers. For this reason, 220 and 284 are called an amicable pair.

Perhaps less well known are longer chains. For example, starting with 12496, we form a chain of five numbers:

12496 → 14288 → 15472 → 14536 → 14264 (→ 12496 → ...)

Since this chain returns to its starting point, it is called an amicable chain.

Find the smallest member of the longest amicable chain with no element exceeding one million.

Answer: 14316 Completed on Fri, 6 Jun 2014, 00:14

Este problema tem a definição:

Os divisores próprios de um número são todos os divisores dele exceto o próprio número. Por exemplo, os divisores próprios de 28 são 1, 2, 4, 7 e 14. Como a soma destes divisores é igual a 28, nós o chamamos de número perfeito.

Curiosamente a soma dos divisores próprios de 220 é 284 e a soma dos divisores próprios de 284 é 220, formando uma cadeia de 2 números. Por essa razão, 220 e 284 são um par de números amigos.

Talvez menos conhecidas são as cadeias mais compridas. Por exemplo, começando com 12496, pode-se formar uma cadeia de 5 números:

12496 → 14288 → 15472 → 14536 → 14264 (→ 12496 → ...)

Dado que esta cadeia retorna ao seu ponto de partida, ela se chama uma cadeia de números amigos.

Encontre o menor membro da maior cadeia de números amigos na qual nenhum elemento supere o milhão.

95.1 Solução em C++

```
#include<stdio.h>
long int nu[500000];
long int ta[500000];
long int cad[1000];
int inu;
long int somdiv(long int n){
    char lixo;
    int som=1;
    int q=2;
    while (q<=(n/2)) {
        if ((n%q)==0) {
            som=som+q;
        }
        q++;
    }
    return(som);
}
main(){
    long int j,k,ic,res,sem;
    inu=0;
    ic=0;
    j=0;
    while(j<=999999){
        ic=1;
```

```

sem=j;
cad[0]=j;
while(ic<1000){
    res=somdiv(sem);
    cad[ic]=res;
    if (res==1) {
        ic=1001;
    }
    if (res>1000000){
        ic=1001;
    }
    if ((res==cad[0])&&(ic!=0)&&(ic!=1002)) {
        nu[inu]=j;
        ta[inu]=ic;
        inu++;
    }
    ic=1001;
}
ic++;
sem=res;
if (ic>1002){printf("deu xabu\n"); exit(1);}
}
j++;
if (0==(j%10000)) {
    printf("ate aqui %d\n",j);
}
}

for (j=0;j<inu;j++){
    printf("%d %d %d\n",j,ta[j],nu[j]);
}

long int max=-99999;
long int maxj=1;
for(j=1;j<=inu;j++){
    if (max>ta[j]) {
        max=ta[j];
    }
    maxj=j;
}
printf("%d",nu[maxj]);
}

```

Este programa rodou e curiosamente listou muitas cadeias com 28 membros, que aparentemente é o maior tamanho de uma cadeia com elementos menores que o milhão. A propósito, como o número inicial faz parte da cadeia, a busca se estendeu até o número 999.999, podendo algum deles escapar da análise por ter formado um elemento da cadeia (lá para a frente) maior que o milhão.

Para resolver o problema, peguei o último número listado e joguei ele na função euler95 do workspace euler711 e achei visualmente o menor elemento da cadeia. Ao digitar este número no euler, tive um susto: acertei de cara.

95.2 Solução em APL

```

[0] euler95 n;c;x;y
[1] c←1pn
[2] t1:
[3] x←divs23 n
[4] y←+/x
[5] →(0=y)/nafo
[6] →(y=c[1])/jatem
[7] c←c,y
[8] →(1000000<y)/milhao
[9] n←y

```

```
[10] →t1
[11] jatem:'cadeia ',⌘pc
[12] c
[13] →0
[14] milhao:'milhao'
[15] →0
[16] nafo:'nao forma cadeia'
```

Capítulo 96

Problema 96 - Sudoku

No original conforme o site projecteuler.net

Su Doku (Japanese meaning number place) is the name given to a popular puzzle concept. Its origin is unclear, but credit must be attributed to Leonhard Euler who invented a similar, and much more difficult, puzzle idea called Latin Squares. The objective of Su Doku puzzles, however, is to replace the blanks (or zeros) in a 9 by 9 grid in such that each row, column, and 3 by 3 box contains each of the digits 1 to 9. Below is an example of a typical starting puzzle grid and its solution grid.

veja figura no site problema 96

A well constructed Su Doku puzzle has a unique solution and can be solved by logic, although it may be necessary to employ "guess and test" methods in order to eliminate options (there is much contested opinion over this). The complexity of the search determines the difficulty of the puzzle; the example above is considered easy because it can be solved by straight forward direct deduction.

The 6K text file, sudoku.txt (right click and 'Save Link/Target As...'), contains fifty different Su Doku puzzles ranging in difficulty, but all with unique solutions (the first puzzle in the file is the example above).

By solving all fifty puzzles find the sum of the 3-digit numbers found in the top left corner of each solution grid; for example, 483 is the 3-digit number found in the top left corner of the solution grid above.

Answer: 24702

Sudoku O quebra cabeça Sudoku (em japonês: lugar dos números) foi inventado há pouco tempo, mas sua origem é creditada a Leonard Euler, que inventou algo bem mais difícil chamado quadrado latino. O objetivo ao resolver um sudoku é trocar os espaços ou zeros em uma grade 9×9 , de modo que cada linha, cada coluna e cada grupo (de 3×3) tenha todos os dígitos de 1 a 9. Abaixo, um puzzle típico com a sua solução ao lado.

0	0	3	0	2	0	6	0	0	4	8	3	9	2	1	6	5	7
9	0	0	3	0	5	0	0	1	9	6	7	3	4	5	8	2	1
0	0	1	8	0	6	4	0	0	2	5	1	8	7	6	4	9	3
0	0	8	1	0	2	9	0	0	5	4	8	1	3	2	9	7	6
7	0	0	0	0	0	0	0	8	7	2	9	5	6	4	1	3	8
0	0	6	7	0	8	2	0	0	1	3	6	7	9	8	2	4	5
0	0	2	6	0	9	5	0	0	3	7	2	6	8	9	5	1	4
8	0	0	2	0	3	0	0	9	8	1	4	2	5	3	7	6	9
0	0	5	0	1	0	3	0	0	6	9	5	4	1	7	3	8	2

Um sudoku bem construído tem solução única e pode ser resolvido apenas usando a lógica. O exemplo acima é considerado fácil. Leia o arquivo citado, contendo 50 puzzles sudoku e resolva-os todos, somando os números obtidos na primeira linha, colunas 1, 2 e 3. Por exemplo, ao resolver o caso acima, você deverá somar 483. O primeiro puzzle do arquivo corresponde ao exemplo acima resolvido. Ache a soma obtida somando os números achados nos sudokus

96.1 Solução em APL

```

    ∇euler96;x;ma;j;ctd;lixo;k
[1]  x←HFM8 'c:/apl2/euler96_sud.txt'
[2]  ctd←0
[3]  x←50 10 9px
[4]  j←1
[5]  ⍝ M96←50 9 9p0
[6]  viv:→(j>50)/fim
[7]  M←M96[j;;]                ⍝←9 9p±(162p1 0)x[j;1+ι9;ι9]
[8]  'M antes ',⌈j
[9]  M
[10] k←1
[11] ufa:→(M[1;k]=0)/achou
[12] k←k+1
[13] →ufa
[14] achou:
[15] SUDOKUa 1,k
[16] 'M depois '
[17] M
[18] '-----'
[19] ⍝lixo←⌈
[20] ctd←ctd++/100 10 1×M[1;1 2 3]
[21] j←j+1
[22] →viv
[23] fim:
[24] ctd

    ∇r←SUDOKUa x;l;c;k;resp;l1;c1
[1]  resp←0
[2]  l←x[1] ⋄ c←x[2]
[3]  ⍝ →(l>3)/feio
[4]  ⍝ se a linha acima for comentada o programa vai ate o fim...
[5]  k←1
[6]  volta:
[7]  resp←0
[8]  →(~EPOSSIVEL k,l,c)/soma
[9]  M[l;c]←k
[10] ⍝ →((l=9)^(c=9))/fechou ----- alteracao -----
[11] →((l=9)^(~0∈c↓M[l;]))/fechou
[12] (l1 c1)←OPROXIMO l,c
[13] resp←SUDOKUa l1,c1
[14] →(resp=999)/feio
[15] →(resp=1)/conta
[16] M[l1;c1]←0
[17] →soma
[18] fechou:resp←1
[19] soma:k←k+1
[20] conta:→((k≤9)^(resp≠1))/volta
[21] acabou:r←resp
[22] →0
[23] feio:r←999

    ∇r←EPOSSIVEL x;k;l;c;sg
[1]  k←x[1] ⋄ l←x[2] ⋄ c←x[3]
[2]  →(~k∈M[l;])/bl1
[3]  r←0
[4]  →0
[5]  bl1:→(~k∈M[;c])/bl2

```

```

[6]   r←0
[7]   →0
[8]   bl2:sg←SUBGRUPO l,c
[9]   →(~k∈(,M)[sg])/bl3
[10]  r←0
[11]  →0
[12]  bl3:r←1
[13]

```

```

      ∇r←OPROXIMO x;l;c
[1]   l←x[1]  ⋄  c←x[2]
[2]   →((l>9)∨(c>9))/lixo
[3]   c←c+1
[4]   s1:→(c>9)/somal
[5]   →(M[l;c]≠0)/somac
[6]   r←l,c
[7]   →0
[8]   somac:c←c+1
[9]   →s1
[10]  somal:c←1
[11]  l←l+1
[12]  →s1
[13]  lixo:'deu xabu'

```

24702

Capítulo 97

Problema 97 - Maior primo não mersenne

The first known prime found to exceed one million digits was discovered in 1999, and is a Mersenne prime of the form $2^{6972593}-1$; it contains exactly 2,098,960 digits. Subsequently other Mersenne primes, of the form 2^p-1 , have been found which contain more digits.

However, in 2004 there was found a massive non-Mersenne prime which contains 2,357,207 digits: $28433 \times 2^{7830457} + 1$. Find the last ten digits of this prime number.

Answer: 8739992577 Completed on Fri, 6 Jun 2014, 11:05

Este problema tem a definição:

O primeiro primo conhecido que superava o milhão de dígitos foi descoberto em 1999, e é um primo de Mersenne que tem a forma $2^{6972593} - 1$; ele contém exatamente 2.098.960 dígitos. Posteriormente foram achados outros primos de Mersenne, com a forma $2^p - 1$ contendo ainda mais dígitos.

No entanto, em 2004 encontrou-se um enorme primo não Mersenne, que contém 2.357.207 dígitos: $28433 \times 2^{7830457} + 1$. Ache os últimos 10 dígitos deste primo.

97.1 Solução em C++

Inacreditável, eis uma solução achada em <http://code.jasonbhill.com/c/project-euler-97/>

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char **argv)
{
    unsigned long i;
    unsigned long long n = 2;
    for (i=0; i<7830456; i++)
        n = (n << 1) % 10000000000;

    n = n * 28433;
    n = n+1;
    n = n % 10000000000;
    printf("Result %lld", n);
    return 0;
}
```

97.2 Solução em APL

Mais impressionante ainda é ter passado isto para o APL2 e rodado e dada a resposta certa, que a propósito é 8739992577. Eis o programa APL2

```
[0] euler97;n;i
[1] n←2
[2] i←0
[3] t1:→(i≥7830456)/fim
[4] n←10000000000|n×2
[5] i←i+1
```

```
[6] →t1
[7] fim:n←n×28433
[8] n←n+1
[9] n←10000000000 | n
[10] n
```

Capítulo 98

Problema 98 - Quadrados anagrâmicos

By replacing each of the letters in the word CARE with 1, 2, 9, and 6 respectively, we form a square number: $1296 = 36^2$. What is remarkable is that, by using the same digital substitutions, the anagram, RACE, also forms a square number: $9216 = 96^2$. We shall call CARE (and RACE) a square anagram word pair and specify further that leading zeroes are not permitted, neither may a different letter have the same digital value as another letter.

Using words.txt (right click and 'Save Link/Target As...'), a 16K text file containing nearly two-thousand common English words, find all the square anagram word pairs (a palindromic word is NOT considered to be an anagram of itself).

What is the largest square number formed by any member of such a pair?

NOTE: All anagrams formed must be contained in the given text file.

Answer: 18769 Completed on Thu, 16 Oct 2014, 09:15

Cujo enunciado é:

Trocando cada uma das letras da palavra inglesa CARE por 1,2,9 e 6 respectivamente, formamos um quadrado perfeito: $1296 = 36^2$. O interessante é que mediante o uso das mesmas substituições de dígitos, o anagrama RACE também forma um quadrado perfeito $9216 = 96^2$. Vai-se chamar CARE e RACE um par de anagramas quadrados e especificaremos ainda que os zeros da esquerda não estão permitidos. Também não pode haver letras diferentes associadas ao mesmo dígito.

Usando o arquivo disponível em projecteuler.net no problema 98 um arquivo de texto com 16KB que tem quase 2000 palavras comuns em inglês, ache todas os pares de anagramas quadrados (as palavras palíndromas não contam como anagramas de si mesmo).

Qual é o maior número quadrado formado por qualquer membro de um par deste tipo ?

NOTA: todos os anagramas formados devem estar contidos no arquivo de texto dado.

Solução

A função **acerta98** lê o arquivo que contém as palavras e cria uma matriz de 1786 x 14 contendo as palavras sem nenhum sinal esquisito. (vírgula, aspas, etc). O nome da matriz é **pls98**.

A função **pares98** lê a matriz **pls98** e cria uma matriz numérica chamada **cands98**, de 59 linhas por duas colunas contendo o índice das palavras candidatas a formarem pares de anagramas.

Para ver estas palavras chame a função **mostra98** que mostra os pares de palavras.

Fico por aqui.

Continuo em outro dia bem mais tarde. Estudando a explicação do dinamarkes, descobri que minha função **pares98** estava errada, e a reescrevi como **pares98alt**. Mudou a lista de palavras.

Olhando a lista descobri que as palavras tem 9 caracteres no máximo. Então $31623^2 = 1000014129$ e $31622^2 = 999950884$ daí porque usar este número no programa.

Dai escrevi a função **euler98**, e foi uma dificuldade só. Finalmente ela deu certo A MENOS da palavra **PATTERN** e **PARTNER**. Na minha opinião esta é a resposta certa. Botei o seguinte post no forum

Can anyone help me ? Why PARTNER and PATTERN aren't the correct answer ?
PARTNER=1860496 and PATTERN=1800964, and $1364^2=1860496$, $1342^2=1800964$.

A resposta correta é $18769 = 137^2$. Um americano gentil me respondeu e explicou porque eu estava errado e o site certo. Este problema fica em aberto.

98.1 Solução em APL

```
vr←euler98;xq;txq;i;j;a;b;rt;a1;a2;alfa  
[1] apares98alt
```

```

[2]  xq←31623 10ρ' '
[3]  txq←31623ρ0
[4]  mai←-9999999
[5]  i←1
[6]  t1:→(i>31623)/oba
[7]  xq[i;]←10†φi*2
[8]  txq[i]←ρφi*2
[9]  i←i+1
[10] →t1
[11] oba:i←1
[12] t2:→(i>1†pcands98)/fim
[13] j←4
[14] t3:→(j>31623)/ufa
[15] a←trim,pls98[cands98[i;1];]
[16] b←trim,pls98[cands98[i;2];]
[17] →(txq[j]≠pa)/somaj
[18] rt←Qa,[0.5]trim xq[j;]
[19] a1←((a1ιa1)=ιpa1)/a1←,rt[;1]      A nao pega pattern e partner...
[20] a2←((a2ιa2)=ιpa2)/a2←,rt[;2]
[21] →((pa1)≠pa2)/somaj
[22] alfa←rt[(,b)ι,rt[;1];2]
[23] →((ρφαalfa)≠txq[j])/somaj
[24] alfa←αalfa
[25] ρ αalfa
[26] →(alfa≠[alfa←alfa*0.5])/somaj
[27] rt
[28] a,' ',b,' ',(φalfa),' ',φ(αxq[j;])*0.5
[29] cand←alfa[(αxq[j;])*0.5]
[30] →((cand=1364)∨cand=1342)/somaj
[31] mai←mai[cand]
[32] somaj:j←j+1
[33] →t3
[34] ufa:
[35] i←i+1
[36] →t2
[37] fim:mai*2

```

18769

Capítulo 99

Problema 99 - Maior exponencial

No original conforme o site projecteuler.net

Comparing two numbers written in index form like 2^{11} and 3^7 is not difficult, as any calculator would confirm that $2^{11} = 2048 < 3^7 = 2187$.

However, confirming that $632382^{518061} > 519432^{525806}$ would be much more difficult, as both numbers contain over three million digits.

Using `base_exp.txt` (right click and 'Save Link/Target As...'), a 22K text file containing one thousand lines with a base/exponent pair on each line, determine which line number has the greatest numerical value.

NOTE: The first two lines in the file represent the numbers in the example given above. Answer: 709

Maior exponencial Comparar dois números escritos na forma exponencial tal como 2^{11} e 3^7 não é difícil e qualquer calculadora poderá confirmar que $2^{11} = 2048 < 3^7 = 2187$. Entretanto, confirmar que $632382^{518061} > 519432^{525806}$ poderia ser muito mais difícil, já que ambos os números contêm mais de 13 milhões de dígitos. No arquivo `bennn.txt` há 600 linhas com pares de base e expoente (um número em cada linha). Determine qual a linha que contém o maior valor numérico. Note que as primeiras duas linhas do arquivo representam os números do exemplo dado acima.

99.1 Solução em APL

```
∇euler99;x;y;z;i;c;MAX;IMAX
[1] x←HFM8 'c:/p/quarta/0099_base_exp.txt'
[2] y←⌊1↓x,','
[3] z←z←1000 2py
[4] i←1
[5] MAX←-999999
[6] t1:→(i>1000)/fim
[7] c←z[i;2]×⊗z[i;1]
[8] →(c<MAX)/soma
[9] MAX←c
[10] IMAX←i
[11] soma:i←i+1
[12] →t1
[13] fim:IMAX
```

Capítulo 100

Problema 100 - Probabilidade arranjada

If a box contains twenty-one coloured discs, composed of fifteen blue discs and six red discs, and two discs were taken at random, it can be seen that the probability of taking two blue discs, $P(BB) = (15/21) \times (14/20) = 1/2$.

The next such arrangement, for which there is exactly 50% chance of taking two blue discs at random, is a box containing eighty-five blue discs and thirty-five red discs.

By finding the first arrangement to contain over $10^{12} = 1,000,000,000,000$ discs in total, determine the number of blue discs that the box would contain.

Answer: 756872327473 Completed on Thu, 16 Oct 2014, 10:38

Cujo enunciado é:

Se uma caixa contém 21 discos coloridos, composta de quinze azuis e seis vermelhos e dois discos são tomados ao acaso pode-se observar que a probabilidade de ter 2 azuis é $P(AA) = \frac{15}{21} \times \frac{14}{20} = \frac{1}{2}$. A próxima disposição para a qual esta probabilidade seja exatamente $\frac{1}{2}$ é uma caixa com 85 discos azuis e 35 discos vermelhos.

Deve-se achar a primeira desta disposições em ter mais do que $10^{12} = 1.000.000.000.000$ discos no total. Achar quantos discos azuis esta caixa deve ter.

solução

Deve-se resolver o sistema

$$\frac{a}{t} \times \frac{a-1}{t-1} = \frac{1}{2}$$

rearranjando

$$2a^2 - 2a - t^2 + t = 0$$

Trata-se de uma equação diofantina. Achei um site de um argentino na internet que resolve

<http://www.alpertron.com.ar/QUAD.HTM>. Precisa liberar a execução de java e coisa e tal, daí ele responde:

$$X_{n+1} = PX_n + QY_n + K$$

$$Y_{n+1} = RX_n + SY_n + L$$

$$P = 3$$

$$Q = 2$$

$$K = -2$$

$$R = 4$$

$$S = 3$$

$$L = -3$$

Ou seja

$$a_{n+1} = 3a_n + 2t_n - 2$$

$$t_{n+1} = 4a_n + 3t_n - 3$$

100.1 Solução em APL

```
⎕←euler100;a;t;aux;taux;obj
[1] ⎕PP←20
[2] a←15
[3] t←21
[4] obj←1000000000000
[5] t1:→(t≥obj)/fim
```

```
[6]    aaux←(3×a)+(2×t)-2
[7]    taux←(4×a)+(3×t)-3
[8]    a←aaux
[9]    t←taux
[10]   →t1
[11] fim:
[12]   r←a
```

```
      euler100
756872327473
```

Capítulo 101

Problema 101 - Polinômio ótimo

If we are presented with the first k terms of a sequence it is impossible to say with certainty the value of the next term, as there are infinitely many polynomial functions that can model the sequence.

As an example, let us consider the sequence of cube numbers. This is defined by the generating function, $u_n = n^3$: 1, 8, 27, 64, 125, 216, ...

Suppose we were only given the first two terms of this sequence. Working on the principle that "simple is best" we should assume a linear relationship and predict the next term to be 15 (common difference 7). Even if we were presented with the first three terms, by the same principle of simplicity, a quadratic relationship should be assumed.

We shall define $OP(k, n)$ to be the n th term of the optimum polynomial generating function for the first k terms of a sequence. It should be clear that $OP(k, n)$ will accurately generate the terms of the sequence for $n \leq k$, and potentially the first incorrect term (FIT) will be $OP(k, k+1)$; in which case we shall call it a bad OP (BOP).

As a basis, if we were only given the first term of sequence, it would be most sensible to assume constancy; that is, for $n \geq 1$, $OP(1, n) = u_1$.

Hence we obtain the following OPs for the cubic sequence:

$$OP(1, n) = 1, 1, 1, 1, \dots$$

$$OP(2, n) = 7n - 6, 1, 8, 15, \dots$$

$$OP(3, n) = 6n^2 - 11n + 6, 1, 8, 27, 58, \dots$$

$$OP(4, n) = n^3, 1, 8, 27, 64, 125, \dots$$

Clearly no BOPs exist for $k \geq 4$.

By considering the sum of FITs generated by the BOPs (indicated in red above), we obtain $1 + 15 + 58 = 74$.

Consider the following tenth degree polynomial generating function:

$$u_n = 1 - n + n^2 - n^3 + n^4 - n^5 + n^6 - n^7 + n^8 - n^9 + n^{10}$$

Find the sum of FITs for the BOPs.

Answer: 37076114526 Completed on Sat, 18 Oct 2014, 13:01

Cujo enunciado é:

Se te derem os k primeiros números em uma série é impossível determinar com certeza o valor do próximo número, já que existem infinitas funções polinômicas que podem formar a série pedida.

Por exemplo, se considerarmos a sequência de números cúbicos. Eles são definidos pela função

$$u_n = n^3 : 1, 8, 27, 64, 125, 216, \dots$$

Suponhamos que nos dão apenas os primeiros 2 números desta série. Usando o princípio de "quanto mais simples, melhor" deveríamos usar uma relação linear e prever que o próximo termo será 15 (pela sua diferença comum de 7). Inclusive se nos dão 3 termos, pelo mesmo princípio da simplicidade, se poderia estabelecer uma relação de segundo grau.

Vamos definir $OP(k, n)$ como o termo n da função polinomial ótima que gera os primeiros k termos de uma série. Deve ficar claro que $OP(k, n)$ vai gerar com precisão os termos da série para $n \leq k$ e que o primeiro termo incorreto ("first incorrect term: FIT") será $OP(k, k+1)$ e neste caso ele será chamado de OP ruim ("bad OP: BOP").

Se nos dão apenas o primeiro termo da série o mais razoável seria supor uma constância deste termo, ou seja para $n \geq 1$, $OP(1, n) = u_1$.

Obtem-se assim os OPs seguintes para a sucessão de números cúbicos

$$OP(1, n) = 1 \quad 1, (1), 1, 1, \dots$$

$$OP(2, n) = 7n - 6 \quad 1, 8, (15), \dots$$

$$OP(3, n) = 6n^2 - 11n + 6 \quad 1, 8, 27, (58), \dots$$

$$OP(4, n) = n^3 \quad 1, 8, 27, 64, 125, \dots$$

É evidente que não existem BOPs para $k \geq 4$. Ao considerar a soma dos FITs gerados pelos BOPs (indicados acima entre parênteses) se obtém $1 + 15 + 58 = 74$.

Considere-se a seguinte função polinomial de grau 10:

$$u_n = 1 - n + n^2 - n^3 + n^4 - n^5 + n^6 - n^7 + n^8 - n^9 + n^{10}$$

Ache a soma dos *FITs* de todos os *BOPs*.

solução

Temos um conjunto de n pontos $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, ao qual queremos ajustar um polinômio de grau k . Dito de outro modo, queremos achar o conjunto de $k + 1$ coeficientes $\{a_0, a_1, a_2, \dots, a_k\}$ tais que permitam obter o polinômio $a_0 + a_1x + \dots + a_{k-1}x^{k-1} + a_kx^k$. Isto pode ser expresso pelo seguinte conjunto de equações lineares

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} x_1^k & x_1^{k-1} & \dots & x_1 & 1 \\ x_2^k & x_2^{k-1} & \dots & x_2 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_n^k & x_n^{k-1} & \dots & x_n & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{bmatrix}$$

Se tivermos mais pontos ($n > k$) então temos um sistema abundante (?) e não há uma solução que ajuste o polinômio com exatidão. Entretanto, pode-se obter o melhor ajuste através do método de mínimos quadrados.

Mas, se tentarmos ajustar um polinômio de grau $k - 1 = n$ a matriz é quadrada, e se todos os valores de x são distintos a matriz será inversível e então o conjunto de equações lineares terá uma solução. Para resolver este problema pode-se usar lápis e papel para resolver os 10 conjuntos de equações lineares começando com

$$y_1 = a_0$$

dando-nos a equação polinomial para 1 ponto. Segue-se o conjunto de 2 equações lineares

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}$$

e assim por diante. Mas, não é fácil fazer, vamos a outras alternativas

interpolação de polinômios de Lagrange

Este método é chamado de interpolação de polinômios de Lagrange. Foi descoberto por Waring em 1779, redescoberto por Euler em 1783 e finalmente publicado por Lagrange em 1785.

Dado um polinômio $P(x)$ de grau k ele pode ser representado usando um conjunto de $n = k + 1$ pontos $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$. Dada esta representação do polinômio pode-se obter o valor do polinômio para qualquer outro ponto através da fórmula

$$P(x) = \sum_{k=0}^{n-1} y_k \frac{\prod_{j \neq k} (x - x_j)}{\prod_{j \neq k} (x_k - x_j)}$$

implementação

Para o nosso problema específico, sabe-se que $x_1 = 1$, $x_2 = 2$ e assim por diante e também sabemos que todos os coeficientes são inteiros.

Para uma avaliação específica no ponto $x_{n+1} = b + 1$ pode-se avaliar o polinômio e então ajustar com a seguinte implementação

```
long result = 0;
for (long i = 1; i <= n; i++) {
    long temp1 = 1;
    long temp2 = 1;

    for (long j = 1; j <= n; j++) {
        if (i == j) {
            continue;
        } else {
            temp1 = temp1 * (n+1 - j);
            temp2 = temp2 * (i - j);
        }
    }
    result = result + (temp1 * y[i-1]/temp2);
}
```

o loop for mais externo corresponde à soma, e os loops internos correspondem aos 2 produtos na fórmula.
Tudo o que precisamos agora é gerar o polinômio

```
using System;
using System.Diagnostics;
namespace euler {
    class Problem101 {
        public static void Main(string[] args) {
            new Problem101().Lagrange();
        }
        public void Lagrange() {
            Stopwatch clock = Stopwatch.StartNew();
            //Generate the generator polynomial
            long[] coefficients = { 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1 };
            Polynomial poly = new Polynomial(coefficients);
            //Generate the points
            long[] y = new long[coefficients.Length];
            for (long i = 0; i < y.Length; i++) {
                y[i] = poly.evaluate(i + 1);
            }
            long fits = 0;
            for (long n = 1; n <= coefficients.Length-1; n++) {
                long result = 0;
                for (long i = 1; i <= n; i++) {
                    long temp1 = 1;
                    long temp2 = 1;
                    for (long j = 1; j <= n; j++) {
                        if (i == j) {
                            continue;
                        } else {
                            temp1 *= n+1 - j;
                            temp2 *= i - j;
                        }
                    }
                    result += temp1 * y[i-1]/temp2;
                }
                fits += result;
                //Console.WriteLine(result);
            }
            clock.Stop();
            Console.WriteLine("The sum of FITs is {0}", fits);
            Console.WriteLine("Solution took {0} ms", clock.Elapsed.TotalMilliseconds);
        }
    }
}
class Polynomial {
    private long[] coefficients;
    public long Degree;
    public Polynomial(long deg) {
        Degree = deg;
        coefficients = new long[deg + 1];
    }
    public Polynomial(long[] coefficients) {
        Degree = coefficients.Length - 1;
        this.coefficients = coefficients;
    }
    public long get(long i) {
        return coefficients[i];
    }
    public void set(long i, long value) {
        coefficients[i] = value;
    }
    public long evaluate(long x) {
        long result = 0;
    }
}
```

```

        for (long i = this.Degree; i >= 0; i--) {
            result = result * x + get(i);
        }
        return result;
    }
}

```

101.1 Solução em APL

```

veuler101;⎕IO;coef;y;i;fits;n;r;j;temp1;temp2
[1]  ⎕IO←0
[2]  coef←1 -1 1 -1 1 -1 1 -1 1 -1 1
[3]  y←11ρ0
[4]  i←0
[5]  t1:→(i≥11)/fim1
[6]  y[i]←avalial01 i+1
[7]  i←i+1
[8]  →t1
[9]  fim1:fits←0
[10] n←1
[11] t2:→(n>10)/fim2
[12] r←0
[13] i←1
[14] t3:→(i>n)/fim3
[15] temp1←1
[16] temp2←1
[17] j←1
[18] t4:→(j>n)/fim4
[19] →(i=j)/naofaz
[20] temp1←temp1×n+1-j
[21] temp2←temp2×i-j
[22] naofaz:j←j+1
[23] →t4
[24] fim4:
[25] r←r+(temp1×y[i-1]÷temp2)
[26] i←i+1
[27] →t3
[28] fim3:
[29] fits←fits+r
[30] n←n+1
[31] →t2
[32] fim2:
[33] fits

```

E a resposta achada (de primeira !) foi a certa: 37076114526

Capítulo 102

Problema 102 - Contenção de triângulo

No original conforme o site projecteuler.net

Three distinct points are plotted at random on a Cartesian plane, for which $-1000 \leq x, y \leq 1000$, such that a triangle is formed.

Consider the following two triangles:

A(-340,495), B(-153,-910), C(835,-947)

X(-175,41), Y(-421,-714), Z(574,-645)

It can be verified that triangle ABC contains the origin, whereas triangle XYZ does not.

Using triangles.txt (right click and 'Save Link/Target As...'), a 27K text file containing the co-ordinates of one thousand "random" triangles, find the number of triangles for which the interior contains the origin.

NOTE: The first two examples in the file represent the triangles in the example given above.

Answer: 228

Triângulos contidos Três pontos distintos, randômicos, dentro de um plano cartesiano, sujeitos a $-1000 \leq x, y \leq 1000$, formam um triângulo. Considere os 2 seguintes triângulos: A(-340, 495), B(-153,-910), C(835,-947) e X(-175,41), Y(-421,-714), Z(574,-645). Pode-se verificar que o triângulo ABC contém a origem, enquanto o triângulo XYZ não a contém. Leia o arquivo citado no site que contém as coordenadas de 1000 triângulos. Ache a quantidade de triângulos que contém a origem. Note que as primeiras duas linhas do arquivo representam os triângulos citados aqui.

102.1 Solução em APL

```

    veuler102;x;i;ctd;y;tot;a1;a2;a3
[1] x←HFM8 'c:/p/quarta/0102_triangles_m.txt'
[2] i←1
[3] ctd←0
[4] t1:→(i>1000)/fim
[5] y←⌊'-' edit x[i;]
[6] tot←|areatr102 y
[7] a1←|areatr102 y[1 2],0 0,y[5 6]
[8] a2←|areatr102 y[1 2 3 4],0 0
[9] a3←|areatr102 0 0,y[3 4 5 6]
[10] i,tot,(a1+a2+a3),a1,a2,a3,y
[11] →(tot≠a1+a2+a3)/pula
[12] ctd←ctd+1
[13] pula:i←i+1
[14] →t1
[15] fim:ctd

    Vr←areatr102 x;z;r1;r2
[1] z←3 3px[1],x[2],1,x[3],x[4],1,x[5],x[6],1
[2] r1←(z[1;1]×z[2;2]×z[3;3])+(z[1;3]×z[2;1]×z[3;2])+(z[1;2]×z[2;3]×z[3;1])
[3] r2←(z[1;3]×z[2;2]×z[3;1])+(z[1;2]×z[2;1]×z[3;3])+(z[1;1]×z[2;3]×z[3;2])
[4] r←0.5×r1-r2
```

Note que os arquivos TXT do projeto euler estão vindo apenas com o caracter LF (`x'13'`) terminando cada linha. Isso enlouquece a minha função de leitura (HFM8). Estou meio sem saco de ampliar o escopo de HFM8 para lidar com isso. Achei mais simples:

- Baixar o arquivo normalmente do site do projeto euler
- Editar o arquivo baixado e mandar trocar: ESTENDIDA,
- Trocar barra-ene por barra-erre barra-ene
- Salvar o arquivo (outro nome ?)
- Processar normalmente este segundo arquivo

Voilà.

Capítulo 103

Problema 103 - Somas de conjuntos especiais ótimas

Seja $S(A)$ a soma dos elementos do conjunto A de tamanho n . Chamaremos isso de *soma especial* do conjunto se para quaisquer dois subconjuntos disjuntos não vazios B e C se as seguintes propriedades são sempre verdadeiras:

1. $S(B) \neq S(C)$; ou seja, as somas dos subconjuntos não podem ser iguais.
2. Se B contém mais elementos do que C , então $S(B) > S(C)$.

Se $S(A)$ é minimizado para um dado n , chamaremos isso de um conjunto de soma especial ótimo. Os primeiros cinco conjuntos de soma especial ótimos são dados abaixo.

$n = 1 : \{1\}$

$n = 2 : \{1, 2\}$

$n = 3 : \{2, 3, 4\}$

$n = 4 : \{3, 5, 6, 7\}$

$n = 5 : \{6, 9, 11, 12, 13\}$

Parece que para um dado conjunto ótimo $A = \{a_1, a_2, \dots, a_n\}$, o próximo conjunto ótimo é da forma $B = \{b, a_1 + b, a_2 + b, \dots, a_n + b\}$ onde b é o elemento do meio na linha anterior.

Ao aplicar esta *regra* esperaríamos o conjunto ótimo para $n=6$ seria $A = \{11, 17, 20, 22, 23, 24\}$ com $S(A) = 117$. No entanto, este não é o conjunto ótimo, pois apenas aplicamos um algoritmo para fornecer um conjunto quase ótimo. O conjunto ótimo para $n = 6$ é $A = \{11, 18, 19, 20, 22, 25\}$ com $S(A) = 115$ e a sequência de caracteres correspondentes é 111819202225. Dado que A é um conjunto de soma especial ideal para $n = 7$, encontre sua sequência definida. NOTA: Este problema está relacionado ao Problema 105 e ao Problema 106.

103.1 Solução em Python

```
def compute():
    TARGET_SIZE = 7
    maxsum = 1

    while SpecialSumSet.make_set(TARGET_SIZE, maxsum) is None:
        maxsum *= 2
        i = maxsum // 4
        while i > 0:
            maxsum -= i
            if SpecialSumSet.make_set(TARGET_SIZE, maxsum) is None:
                maxsum += i
            i //= 2

    set = SpecialSumSet.make_set(TARGET_SIZE, maxsum) # Must be not None
    return "".join(map(str, set.values))

class SpecialSumSet:
    @staticmethod
    def make_set(targetsize, maximumsum):
        return SpecialSumSet._make_set(SpecialSumSet([], [True], [0], [0]), targetsize, \
            maximumsum, 1)
    @staticmethod
```

```

def _make_set(set, sizeremain, sumremain, startval):
    if sizeremain == 0: # Base case - success
        return set
    if sizeremain >= 2 and startval * sizeremain >= sumremain:
        return None
    endval = sumremain
    if len(set.values) >= 2:
        endval = min(set.values[0] + set.values[1] - 1, endval)
    for val in range(startval, endval + 1):
        temp = set.add(val)
        if temp is None:
            continue
        temp = SpecialSumSet._make_set(temp, sizeremain - 1, sumremain - val, val + 1)
        if temp is not None:
            return temp
    return None # No solution for the given current state
def __init__(self, vals, sumposb, minsum, maxsum):
    self.values = vals
    self.sumpossible = sumposb
    self.minimumsum = minsum
    self.maximumsum = maxsum
def add(self, val):
    if val <= 0:
        raise ValueError("Value must be positive")
    size = len(self.values)
    if size >= 1 and val <= self.values[-1]:
        raise ValueError("Must add values in ascending order")
    posb = self.sumpossible
    if any((posb[i] and posb[i - val]) for i in range(val, len(posb))):
        return None
    newsize = size + 1
    minsum = self.minimumsum
    maxsum = self.maximumsum
    newmin = [0] + [min(minsum[i], minsum[i - 1] + val) for i in range(1, newsize)] + \
        [minsum[size] + val]
    newmax = [0] + [max(maxsum[i], maxsum[i - 1] + val) for i in range(1, newsize)] + \
        [maxsum[size] + val]
    if any((newmax[i] >= newmin[i + 1]) for i in range(newsize)):
        return None
    newposb = posb + [False] * val
    for i in reversed(range(val, len(newposb))):
        newposb[i] |= newposb[i - val]
    return SpecialSumSet(self.values + [val], newposb, newmin, newmax)
if __name__ == "__main__":
    print(compute())

```

Programa buscado em <https://www.nayuki.io/page/project-euler-solutions>.

20313839404245

Capítulo 104

Problema 104 - Número de Fibonacci pandigital no fim

The Fibonacci sequence is defined by the recurrence relation:

$F_n = F_{n-1} + F_{n-2}$, where $F_1 = 1$ and $F_2 = 1$. It turns out that F_{541} , which contains 113 digits, is the first Fibonacci number for which the last nine digits are 1-9 pandigital (contain all the digits 1 to 9, but not necessarily in order). And F_{2749} , which contains 575 digits, is the first Fibonacci number for which the first nine digits are 1-9 pandigital.

Given that F_k is the first Fibonacci number for which the first nine digits AND the last nine digits are 1-9 pandigital, find k .

Answer: 329468

Cujo enunciado é: A sucessão de Fibonacci se define pela seguinte relação de recorrência

$$F_n = F_{n-1} + F_{n-2}$$

onde $F_1 = 1$ e $F_2 = 1$. Sabe-se que F_{541} que contém 113 dígitos, é o primeiro número de Fibonacci para o qual os últimos 9 dígitos são pandigitais de 1 a 9 (contém todos os dígitos do 1 ao 9, mas não necessariamente nesta ordem). E o F_{2749} o qual contém 575 dígitos é o primeiro número de Fibonacci para o qual os primeiros 9 dígitos são pandigitais de 1 a 9.

Sabendo que F_k é o primeiro número de Fibonacci para o qual os primeiros 9 dígitos e os últimos 9 dígitos são pandigitais de 1 a 9 ache o valor de k .

solução

Primeiro uma implementação 100% aderente à definição. Usei este programa APL

```

    ∇euler104;a;n;b;he;ta;r;che;cta
[1]  a←'1'
[2]  b←'1'
[3]  n←2
[4]  t1:r←a diomais104 b
[5]  n←n+1
[6]  he←9↑r
[7]  ta←~9↑r
[8]  che←(('1'€he)^(~2'€he)^(~3'€he)^(~4'€he)^(~5'€he)^(~6'€he)^(~7'€he)^(~8'€
      he)^(~9'€he))
[9]  cta←(('1'€ta)^(~2'€ta)^(~3'€ta)^(~4'€ta)^(~5'€ta)^(~6'€ta)^(~7'€ta)^(~8'€
      ta)^(~9'€ta))
[10] →(0=∇/che,cta)/proximo
[11] (⌘n),' - ',(⌘che),' - ',he,'          ',(⌘cta),' - ',ta
[12] →(1=^/che,cta)/0
[13] proximo:
[14] a←b
[15] b←r
[16] →t1

```

```

    ∇r←a diomais104 b;base;tam;i;vaium;parc
[1]  a←,a ⋄ b←,b ⋄ base←10
[2]  tam←1+(pa)⌈pb

```

```

[3]  a←(-tam)↑(tamp'0'),a
[4]  b←(-tam)↑(tamp'0'),b
[5]  r←tamp' '
[6]  i←tam
[7]  vaium←0
[8]  t1:→(i<1)/fim
[9]  parc←vaium+(⊕a[i])+⊕b[i]
[10] vaium←0
[11] →(parc<base)/ok
[12] vaium←1
[13] parc←parc-base
[14] ok:r[i]←1 0⊥parc
[15] i←i-1
[16] →t1
[17] fim:→(r[1]≠'0')/0
[18] r←1↓r

```

Estes dois programas rodando por 40 horas chegaram ao resultado correto, que era 329468. Daí entrei no forum e usei estratégias mais eficientes:

Usei para o pandigital do fim, o resto da divisão por 1000000000. Para o pandigital do começo, usei ...veja la...

O resumo da ópera é que a primeira solução demorou 40 horas de processamento e a segunda solução demorou 4 segundos.

104.1 Solução em APL

```

      ∇euler104v2;a;b;i;re;r;x
[1]  ⍳PP←16
[2]  a←1
[3]  b←1
[4]  i←2
[5]  t1:r←1000000000|a+b
[6]  i←i+1
[7]  re←⊥r
[8]  →((('1'∈re)^( '2'∈re)^( '3'∈re)^( '4'∈re)^( '5'∈re)^( '6'∈re)^( '7'∈re)^( '8'∈re)
      ^('9'∈re))/esim
[9]  vorta:
[10] a←b
[11] b←r
[12] →t1
[13] esim:
[14] i,r
[15] x←fibo104f i
[16] →((('1'∈x)^( '2'∈x)^( '3'∈x)^( '4'∈x)^( '5'∈x)^( '6'∈x)^( '7'∈x)^( '8'∈x)^( '9'∈x)
      )/ufa
[17] →vorta
[18] ufa:i
[19]

```

```

      ∇r←fibo104f n;g;t
[1]  g←0.5×1+5×0.5
[2]  t←(n×10⊗g)+10⊗1÷5×0.5
[3]  r←9↑⊥(10×12)×10×t-⌊t

```

104.2 Solução em J

```
fibo104f =: 3 : 0
```

```

g=:0.5*1+5^0.5
t=(y*10^.g)+10^.1%5^0.5
9{.":(10^12)*10^t-<.t
)

euler104=: 3 : 0
(9!:11)16
a=:1x
b=:1x
i=:2x
label_t1.
r=:1000000000|a+b
i=:i+1
re=:":r
if.('1'e.re)*.('2'e.re)*.('3'e.re)*.('4'e.re)*.('5'e.re)*.('6'e.re)*.('7'e.re)*.('8'e.re)*.
goto_esim.
end.
label_vorta.
a=:b
b=:r
goto_t1.
label_esim.
z=:fibo104f i
if.('1'e.z)*.('2'e.z)*.('3'e.z)*.('4'e.z)*.('5'e.z)*.('6'e.z)*.('7'e.z)*.('8'e.z)*.('9'e.z)
goto_ufa.
end.
goto_vorta.
label_ufa.
i
)

load 'c:/p/quarta/euler104.ijs'
euler104''
329468

```

Capítulo 105

Problema 105 - a resolver

Não resolvido

Capítulo 106

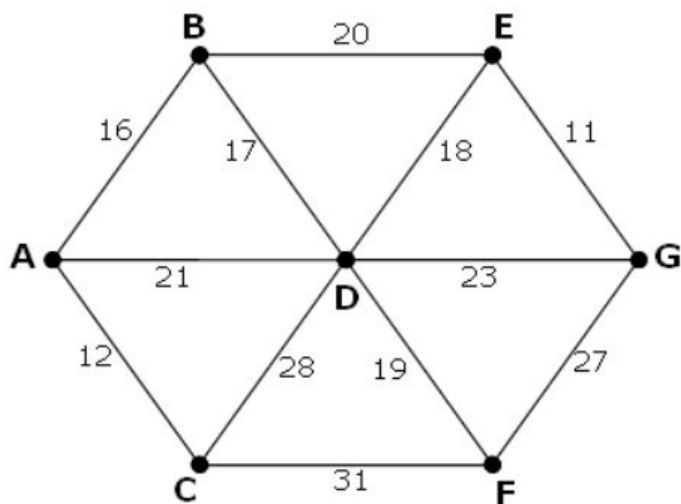
Problema 106 - a resolver

Não resolvido

Capítulo 107

Problema 107 - Rede mínima

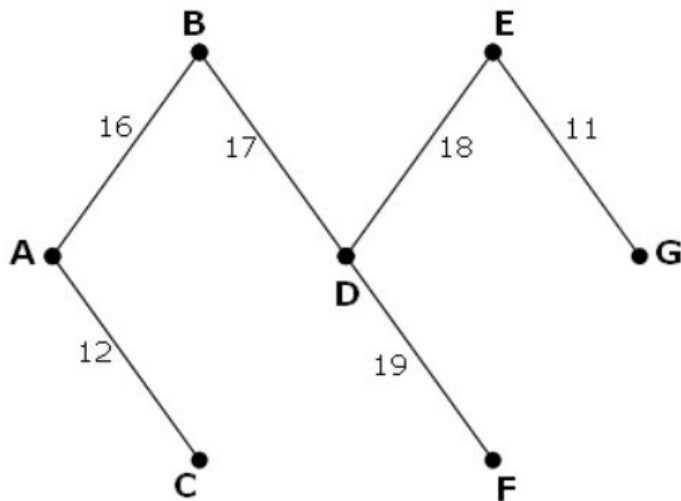
A seguinte rede não direcionada consiste em sete vértices e doze arestas com um peso total de 243.



A mesma rede pode ser representada pela matriz abaixo.

	A	B	C	D	E	F	G
A	-	16	12	21	-	-	-
B	16	-	-	17	20	-	-
C	12	-	-	28	-	31	-
D	21	17	28	-	18	19	23
E	-	20	-	18	-	-	11
F	-	-	31	19	-	-	27
G	-	-	-	23	11	27	-

No entanto, é possível otimizar a rede removendo algumas arestas e ainda garantir que todos os pontos na rede permaneçam conectados. A rede que atinge a economia máxima é mostrada abaixo. Ela tem um peso de 93, representando uma economia de $243 - 93 = 150$ da rede original.



Usando network.txt (clique com o botão direito e 'Salvar link/destino como...'), um arquivo de texto de 6K contendo uma rede com quarenta vértices e fornecido em forma de matriz, encontre a economia máxima que pode ser alcançada removendo arestas redundantes, garantindo ao mesmo tempo que a rede permaneça conectada.

A resposta é 259679.

A solução foi retirada do ws vivx634, já que a tentativa de reimplantar um algoritmo já conhecido (Prim), enfrentou alguns problemas. Mas, ladrão que rouba de si próprio não é ladrão.

107.1 Solução em APL

```

∇euler107;ww;zz;i;j
[1] ww←HFM8 'c:/p/quarta/0107_network.txt'
[2] zz←(2ρ1↑pw)ρ0
[3] :For i :In 1↑pw
[4]   zz[i;]←⊖'-0' edit ww[i;]
[5] :EndFor
[6] cus←0 3ρ0
[7] :For i :In 1↑pw
[8]   zz[i;i]←0
[9] :EndFor
[10] cumax←+/,zz
[11] (1↑pw)resolve constrlista zz

```

```

∇r←constrlista aa;i;j;ordem
[1] r←0 3ρ0
[2] i←1 ⋄ ordem←1↑paa
[3] tt:→(i>ordem)/aca
[4] j←1
[5] tr:→(j>ordem)/ufa
[6] →(aa[i;j]=0)/zero
[7] r←r,[1]1 3paa[i;j],i,j
[8] zero:→(j=j+1)/tr
[9] ufa:→(i=i+1)/tt
[10] aca:r←r[⊥r[;1];]
[11]

```

```

∇r←xu ondetta are;i
[1] i←1 ⋄ r←0
[2] tt:→(i>1↑pxu)/term
[3] →(are∈xu[i;])/achou
[4] naoachou:→(i=i+1)/tt
[5] achou:r←i
[6] term:

```

```

       $\forall r \leftarrow \text{ordem resolve arestas; arvore; xu; alin; acol; i; cus}$ 
[1]   arvore  $\leftarrow 0$  2p0  $\diamond$  cus  $\leftarrow 0$ 
[2]   xu  $\leftarrow (\text{ordem}, \text{ordem})p0$ 
[3]   xu[;1]  $\leftarrow (\text{ordem}, 1)p1\text{ordem}$ 
[4]   i  $\leftarrow 1$ 
[5]   tr:  $\rightarrow (i > 1 \uparrow \text{parestas})/\text{fim}$ 
[6]    $\rightarrow ((\text{alin} \leftarrow \text{xu ondet a arestas}[i;2]) = (\text{acol} \leftarrow \text{xu ondet a arestas}[i;3]))/\text{despreze}$ 
[7]   xu[alin;]  $\leftarrow (1 \downarrow \text{pxu}) \uparrow ((\text{xu}[alin;] \neq 0)/\text{xu}[alin;]), (\text{xu}[acol;] \neq 0)/\text{xu}[acol;]$ 
[8]   xu[acol;]  $\leftarrow 0$ 
[9]   arvore  $\leftarrow \text{arvore}, [1]1 \ 2\text{parestas}[i;2], \text{arestas}[i;3]$ 
[10]  cus  $\leftarrow \text{cus} + \text{arestas}[i;1]$ 
[11]   $\mathcal{R} \text{ xu}$ 
[12]   $\mathcal{R}' \text{ ---- '}$ 
[13]  despreze:  $\rightarrow (i = i \leftarrow i + 1)/\text{tr}$ 
[14]  fim: r  $\leftarrow \text{arvore}$ 
[15]  CUS  $\leftarrow \text{cus}$ 

```

```

      CUS
2153
      cumax
261832
      261832-2153
259679

```

Capítulo 108

Problema 108 - Recíprocas diofantinas - I

Na equação a seguir x , y e n são inteiros positivos

$$\frac{1}{x} + \frac{1}{y} = \frac{1}{n}$$

Para $n = 4$ existem exatamente 3 soluções distintas:

$$\frac{1}{5} + \frac{1}{20} = \frac{1}{4}$$

$$\frac{1}{6} + \frac{1}{12} = \frac{1}{4}$$

$$\frac{1}{8} + \frac{1}{8} = \frac{1}{4}$$

Qual o menor valor de n para o qual o número de soluções distintas excede a 1000 ?

A resposta correta é 180180.

Resolvido pelo usuário brian em python no repositório de soluções do problema 108 do site euler. Ele escreveu

I used pretty much the same solution as rayfil, except that I first tried numbers with lots of distinct prime factors. `nsols(2*3*5*7*11*13*17)` gave 1094 solutions, but wasn't the answer, so I then tried multiples of 30030 below this and got the result.

Usei praticamente a mesma solução do rayfil, exceto que primeiro tentei números com muitos fatores primos distintos. `nsols(2*3*5*7*11*13*17)` deu 1094 soluções, mas não era a resposta, então tentei múltiplos de 30030 abaixo disso e obtive o resultado.

108.1 Solução em Python

```
def nsols(n):  
    return sum(1 for x in range(n+1, n*2+1)  
               if ((n*x) % (x-n)==0))  
  
f = 2*3*5*7*11*13 # 2*3*5*7*11*13 = 30030  
for i in range(f, f*17, f):  
    if nsols(i) >= 1000:  
        print(i)  
        break  
  
180180
```

108.2 Solução em APL

```
∇r←euler108;f;i  
[1] f←2×3×5×7×11×13  
[2] i←f  
[3] :While i<f×17
```

```

[4]      :If 1000≤nsols108 i
[5]      i
[6]      :Return
[7]      :End
[8]      i←i+f
[9]      :End

```

```

      ∇r←nsols108 n
[1]      r←0
[2]      x←n+1
[3]      :While x<1+2×n
[4]      :If 0=(x-n)|n×x
[5]      r←r+1
[6]      :End
[7]      x←x+1
[8]      :End

```

180180

108.3 Solução em J

```

nsol108=: 3 : 0
  r=.0
  xx=.y+1
  while. xx<1+2*y do.
    if. 0=(xx-y)|y*xx do.
      r=.r+1
    end.
    xx=.xx+1
  end.
  r
)
euler108=: 3 : 0
  f=.2*3*5*7*11*13
  i=.f
  while. i<f*17 do.
    if. 1000<:nsol108 i do.
      i
      return.
    end.
    i=.i+f
  end.
)

```

```

      load 'c:/p/quarta/euler108.ijs'
      euler108''
180180

```

Capítulo 109

Problema 109 - Dardos

No jogo de dardos, um jogador lança três dardos em um alvo que é dividido em vinte seções de tamanhos iguais numeradas de um a vinte.



A pontuação de um dardo é determinada pelo número da região em que o dardo cai. Um dardo que cai fora do anel externo vermelho/verde pontua zero. As regiões preta e creme dentro deste anel representam pontuações simples. No entanto, o anel externo vermelho/verde e o anel do meio pontuam dupla e triplamente, respectivamente.

No centro do tabuleiro há dois círculos concêntricos chamados de bull region, ou bulls-eye. O bull externo vale 25 pontos e o bull interno é um double, vale 50 pontos.

Existem muitas variações de regras, mas no jogo mais popular os jogadores começarão com uma pontuação de 301 ou 501 e o primeiro jogador a reduzir seu total de corrida para zero é o vencedor. No entanto, é normal jogar um sistema de "doubles out", o que significa que o jogador deve acertar um double (incluindo o double bulls-eye no centro do tabuleiro) em seu dardo final para vencer; qualquer outro dardo que reduza seu total de corrida para um ou menos significa que a pontuação para aquele conjunto de três dardos é "estourada".

Quando um jogador consegue terminar com sua pontuação atual, isso é chamado de "checkout" e o checkout mais alto é 170: T20 T20 D25 (dois 20s triplos e um bull duplo).

Existem exatamente onze maneiras distintas de fazer check-out com uma pontuação de 6:

D3
 D1 D2
 S2 D2
 D2 D1
 S4 D1
 S1 S1 D2
 S1 T1 D1
 S1 S3 D1
 D1 D1 D1
 D1 S2 D1
 S2 S2 D1

Note que D1 D2 é considerado diferente de D2 D1, pois eles terminam em duplas diferentes. No entanto, a combinação S1 T1 D1 é considerada a mesma que T1 S1 D1.

Além disso, não incluiremos erros ao considerar combinações; por exemplo, D3 é o mesmo que 0 D3 e 0 0 D3.

Incrivelmente, há 42.336 maneiras diferentes de fazer check-out no total.

De quantas maneiras diferentes um jogador pode fazer check-out com uma pontuação menor que 100?

109.1 Solução em Python

A solução é a do site <https://github.com/nayuki/Project-Euler-solutions/blob/master/python/p109.py?ts=0>

```

def compute():
    # Both lists are orderless but duplicates are important; they are sort of like multisets
    points = [i * j for i in range(1, 21) for j in range(1, 4)] + [25, 50]
    doublepoints = [i * 2 for i in range(1, 21)] + [25 * 2]

    # Memoization array, with dimensions (3, 101, len(points))
    ways = [[[None] * len(points) for j in range(101)] for i in range(3)]

    # Number of ways to get exactly 'total' points in exactly 'throws' throws, using
    # items (unordered) from the 'points' list with index less than or equal to 'maxIndex'.
    def calc_ways(throws, total, maxindex):
        if ways[throws][total][maxindex] is None:
            if throws == 0:
                result = 1 if total == 0 else 0
            else:
                result = 0
                if maxindex > 0:
                    result += calc_ways(throws, total, maxindex - 1)
                    if points[maxindex] <= total:
                        result += calc_ways(throws - 1, total - points[maxindex], maxindex)
            ways[throws][total][maxindex] = result
        return ways[throws][total][maxindex]

    checkouts = 0
    for remainingpoints in range(1, 100):
        for throws in range(3):
            for p in doublepoints:
                if p <= remainingpoints:
                    checkouts += calc_ways(throws, remainingpoints - p, len(points) - 1)
    return str(checkouts)
print(compute())
  
```

Capítulo 110

Problema 110 - Diofantinas Recíprocas II

Na equação a seguir x , y e n são inteiros positivos.

$$\frac{1}{x} + \frac{1}{y} = \frac{1}{n}$$

Pode-se verificar que quando $n = 1260$ há 113 soluções distintas e este é o menor valor de n para o qual o número total de soluções distintas excede cem.

Qual é o menor valor de n para o qual o número de soluções distintas excede quatro milhões?

OBSERVAÇÃO: Este problema é uma versão muito mais difícil do Problema 108 e, como está muito além das limitações de uma abordagem de força bruta, requer uma implementação inteligente.

110.1 Solução em Python

A solução foi buscada em <http://wiki.san-ss.com.ar/project-euler-problem-110> que aparentemente é a solução de Santiago Alessandri.

```
from functools import reduce
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
mult = lambda x: reduce(lambda y, z: y * z, x, 1)
translate = lambda x: mult(primes[i] ** ((x[i] - 1) // 2) for i in range(len(x)))

def generator(limit):
    l = [1] * 14
    while l[13] < limit:
        i = 13
        while i > 0 and (l[i] == l[i-1]):
            l[i] = 1
            i -= 1
        l[i] += 2
        yield l

result = min(translate(l) for l in generator(13) if mult(l) > 8000000)
print("The result is:", result)
```

The result is: 9350130049860600

Capítulo 111

Problema 111 - Primos com corridas

Considerando números primos de 4-dígitos que contêm dígitos repetidos, é claro que eles não podem ser todos iguais: 1111 é divisível por 11 e 2222 é divisível por 22, e assim por diante. Mas há nove primos de 4-dígitos contendo três uns:

1117, 1151, 1171, 1181, 1511, 1811, 2111, 4111, 8111.

Diremos que $M(n, d)$ representa o número máximo de dígitos repetidos para um primo de n -dígitos onde d é o dígito repetido, $N(n, d)$ representa o número de tais primos, e $S(n, d)$ representa a soma desses números primos.

Então $M(4, 1) = 3$ é o número máximo de dígitos repetidos para um primo de 4-dígitos onde 1 é o dígito repetido, existem $N(4, 1) = 9$ de tais primos, e a soma desses primos é $S(4, 1) = 22275$. Acontece que para $d = 0$, só é possível ter $M(4, 0) = 2$ dígitos repetidos, mas há $N(4, 0) = 13$ de tais casos.

Da mesma forma obtemos os seguintes resultados para primos de 4-dígitos.

Digit, d	M(4, d)	N(4, d)	S(4, d)
0	2	13	67061
1	3	9	22275
2	3	1	2221
3	3	12	46214
4	3	2	8888
5	3	1	5557
6	3	1	6661
7	3	9	57863
8	3	1	8887
9	3	7	48073

Para $d = 0$ a 9 a soma de todos os $S(4, d)$ é 273700. Ache a soma de todos os $S(10, d)$.

111.1 Solução em Python

Solução obtida de <https://walkerdoescode.home.blog/2019/12/21/project-euler-problem-111/>

```
import time
from math import sqrt

def isPrime(n):
    if(n%2==0):
        return False
    c = 3
    upper = sqrt(n)
    while(c<=upper):
        if(n%c==0):
            return False
        c+=2
    return True

def convertNDigits(n,x):
    a = str(x)
    while len(a)<n:
```

```

        a = "0" + a
    return a

def allPermutations(myList):
    l = len(myList)
    if l==0:
        return [[]]
    if(l==1):
        return [myList]
    a = myList[0]
    b = allPermutations(myList[1:])
    total = []
    for x in b:
        for z in range(l):
            new = x[0:z]
            new.append(a)
            new.extend(x[z:])
            if new not in total:
                total.append(new)
    return total

def projectEulerProblemOneHundredEleven(n):
    final = 0
    for d in range(0,10):
        for m in range(n,-1,-1):
            total = 0
            order = []
            for x in range(m):
                order.append(1)
            for x in range(m,n):
                order.append(0)
            v = allPermutations(order)
            for x in range(0,10**(n-m)):
                a = convertNDigits(n-m, x)
                possible = True
                for q in a:
                    if q == str(d):
                        possible = False
                        break
                if possible:
                    for z in v:
                        s = ""
                        b = 0
                        for y in z:
                            if y==1:
                                s+=str(d)
                            else:
                                s+=a[b]
                                b+=1
                        if int(s[0])>0 and isPrime(int(s)):
                            total+=int(s)
            if total>0:
                final+=total
                break
    return final

start = time.time()
print ( projectEulerProblemOneHundredEleven(10))
print ("--- %s seconds ---" % (time.time()-start))

```

612407567715

--- 0.68894362449646 seconds ---

Capítulo 112

Problema 112 - Números rebotados

No original conforme o site projecteuler.net

Working from left-to-right if no digit is exceeded by the digit to its left it is called an increasing number; for example, 134468.

Similarly if no digit is exceeded by the digit to its right it is called a decreasing number; for example, 66420.

We shall call a positive integer that is neither increasing nor decreasing a "bouncy" number; for example, 155349.

Clearly there cannot be any bouncy numbers below one-hundred, but just over half of the numbers below one-thousand (525) are bouncy. In fact, the least number for which the proportion of bouncy numbers first reaches 50

Surprisingly, bouncy numbers become more and more common and by the time we reach 21780 the proportion of bouncy numbers is equal to 90

Find the least number for which the proportion of bouncy numbers is exactly 99

Answer: 1587000 Completed on Tue, 21 May 2013, 18:26

Números rebotados Trabalhando da esquerda para a direita, se nenhum dígito é excedido pelo seu vizinho da esquerda, diz-se que o número é crescente, como por exemplo, 134468. Similarmente, se nenhum dígito é excedido pelo seu vizinho da direita, diz-se que ele é decrescente, por exemplo, 66420. Se um inteiro positivo não é crescente nem decrescente, ele é um rebotado: por exemplo 155349. Claramente, não há nenhum rebotado abaixo de 100, mas mais da metade dos números abaixo de 1000 é rebotado (são 525). De fato, o menor número para o qual a proporção de rebotados primeiro alcança 50% é 538. Surpreendentemente os números rebotados tornam-se mais comuns à medida em que o número estudado cresce. Ache o menor número para o qual a proporção de rebotados é exatos 99%.

112.1 Solução em APL

```

    ∇euler112;x;ctb;perc
[1]  x←1
[2]  ctb←0
[3]
[4]  t1:
[5]  ctb←ctb+bounce x
[6]  perc←ctb÷x
[7]  →(perc≥0.99)/ufa
[8]
[9]  x←x+1
[10] →t1
[11] ufa:x,perc

    ∇r←bounce x;z;i;asc;des;tam
[1]  A devolve 1 se o x for bounce, 0 senao
[2]  r←0
[3]  tam←1+[10×x
[4]  z←(tamp10)⌊x
[5]  asc←1
[6]  i←1
[7]  t1:→(i≥pz)/f6
[8]  asc←asc^z[i+1]≥z[i]
[9]  i←i+1
```

```

[10] →t1
[11] f6:
[12] des←1
[13] i←1
[14] t2:→(i≥pz)/f7
[15] des←des^z[i+1]≤z[i]
[16] i←i+1
[17] →t2
[18] f7:r←(asc=0)^(des=0)
[19]

```

```

      euler112
1587000 0.99

```

112.2 Solução em J

```

bounce=: 3 : 0
  r=.0
  tam=.1+<.10^y
  z=.(tam$10)#:y NB. #: equivale ao T apleano
  asc=.1
  i=.0
  while.i<_1+$z do.
    asc=.asc *. (i+1)}z>i}z
    i=.i+1
  end.
  des=.1
  i=.0
  while.i<_1+$z do.
    des=.des *. ((i+1)}z)<i}z
    i=.i+1
  end.
  (asc=0)*.des=0
)
euler112=: 3 : 0
  xx=.1
  ctb=.0
  perc=.ctb%xx
  while. perc<0.99 do.
    ctb=.ctb+ bounce xx
    perc=.ctb%xx
    xx=.xx+1
  end.
  20j0 20j5":(xx-1),perc
)

```

```

load 'c:/p/quarta/euler112.ijs'
euler112 ''
      1587000      0.99000

```

Capítulo 113

Problema 113 - Números não rebotados

Trabalhando da esquerda para a direita, se nenhum dígito for excedido pelo dígito à sua esquerda, um número é chamado de número crescente; por exemplo, 134468.

Da mesma forma, se nenhum dígito for excedido pelo dígito à sua direita, é chamado de número decrescente; por exemplo, 66420.

Chamaremos um número inteiro positivo que não é crescente nem decrescente de um número "rebotado"; por exemplo, 155349.

À medida em que n aumenta, a proporção de números rebotados abaixo de n aumenta também de tal forma que só há 12951 números abaixo de um milhão que não são rebotados e apenas 277032 números não rebotados abaixo de 10^{10} .

Quantos números abaixo de um googol (10^{100}) não são rebotados?

A resposta é 51161058134250.

113.1 Solução em Python

Solução obtida de <https://www.nayuki.io/page/project-euler-solutions> O autor deu uma explicação que parece bem razoável. Veja lá no site.

```
import math
def compute():
    DIGITS = 100
    increasing = math.comb(DIGITS + 9, 9) - 1
    decreasing = math.comb(DIGITS + 10, 10) - (DIGITS + 1)
    flat = DIGITS * 9
    ans = increasing + decreasing - flat
    return str(ans)
print(compute())
```

51161058134250

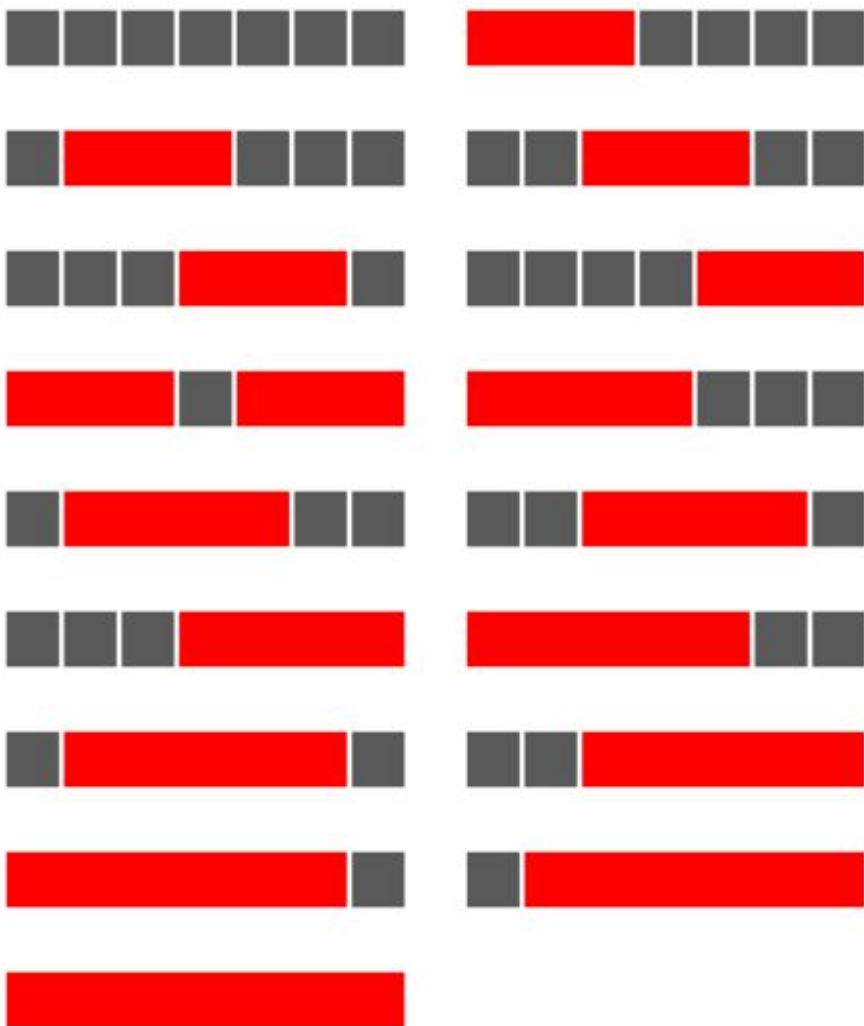
Capítulo

114

Problema 114 - Combinações de blocos de contagem

I

Uma fileira medindo sete unidades de comprimento tem blocos vermelhos com um comprimento mínimo de três unidades colocados nela, de modo que quaisquer dois blocos vermelhos (que podem ter comprimentos diferentes) sejam separados por pelo menos um quadrado cinza. Há exatamente dezessete maneiras de fazer isso.



De quantas maneiras uma linha medindo cinquenta unidades de comprimento pode ser preenchida?

NOTA: Embora o exemplo acima não se preste a essa possibilidade, em geral é permitido misturar tamanhos de bloco. Por exemplo, em uma fileira medindo oito unidades de comprimento, você pode usar vermelho (3), cinza (1) e vermelho (4).

A resposta é 16475640049.

114.1 Solução em Python

A resposta é apanhada em <https://www.nayuki.io/page/project-euler-solutions>.

```
def compute():
    # Dynamic programming
    LENGTH = 50
    ways = [0] * (LENGTH + 1)
    for n in range(len(ways)):
        if n < 3:
            ways[n] = 1
        else:
            ways[n] = ways[n - 1] + sum(ways[ : n - 3]) + 1
    return str(ways[-1])
print(compute())
```

114.2 Solução em J

```
euler114=: 3 : 0
    tam=.50
    ways=.(tam+1)$0
    for_n. i.$ways do.
        if. n<3 do.
            ways=.1 n}ways
        else.
            ways=.(1+ (+/(i.(n-3)){ways)  +(n-1){ways }n}ways
        end.
    end.
    _1{.ways
)

load'c:/p/quarta/euler114.ijs'
euler114''
16475640049
```

Capítulo 115

Problema 115 - Combinações de blocos de contagem II

NOTA: Esta é uma versão mais difícil do Problema 114 .

Uma linha medindo n unidades de comprimento tem blocos vermelhos com comprimento mínimo de m unidades colocadas sobre ele, de modo que quaisquer dois blocos vermelhos (que podem ter comprimentos diferentes) sejam separados por pelo menos um quadrado preto.

A função de contagem de preenchimento, $F(m, n)$ representa o número de maneiras pelas quais uma linha pode ser preenchida.

Por exemplo, $F(3, 29) = 673135$ e $F(3, 30) = 1089155$

Isto é, para $m = 3$, pode-se observar que $n = 30$ é o menor valor para o qual a função de contagem de preenchimento excede um milhão pela primeira vez.

Da mesma forma, para $M = 10$, pode-se verificar que $F(10, 56) = 880711$ e $F(10, 57) = 1148904$, então $n = 57$ é o menor valor para o qual a função fill-count excede um milhão pela primeira vez.

Para $m = 50$, encontre o menor valor de n para o qual a função de contagem de preenchimento excede primeiro um milhão.

115.1 Solução em Python

A resposta é apanhada em <https://www.nayuki.io/page/project-euler-solutions>.

```
import itertools
def compute():
    # Dynamic programming
    M = 50
    ways = [1]
    for n in itertools.count(1):
        s = ways[n - 1] + sum(ways[ : max(n - M, 0)])
        if n >= M:
            s += 1
        ways.append(s)
        if s > 1000000:
            return str(n)
    print(compute())
```

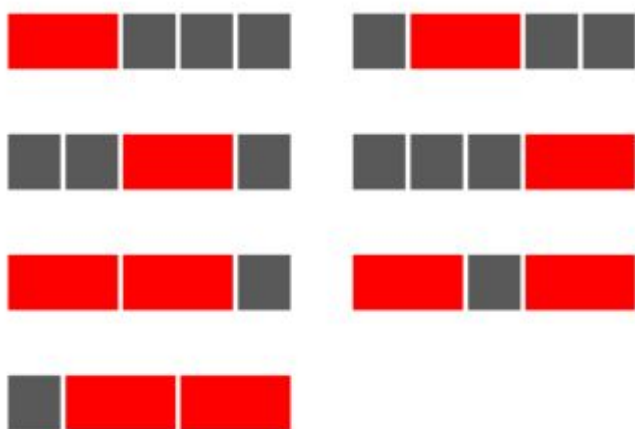
168

Capítulo 116

Problema 116 - Azulejos vermelhos, verdes ou azuis

Uma fileira de cinco peças quadradas cinzas deve ter algumas delas substituídas por peças retangulares coloridas escolhidas entre vermelho (comprimento dois), verde (comprimento três) ou azul (comprimento quatro).

Se forem escolhidas peças vermelhas, há exatamente sete maneiras de fazer isso.



Se forem escolhidas peças verdes, há três maneiras.



E se as peças azuis forem escolhidas, há duas maneiras.



Supondo que as cores não podem ser misturadas, existem $7 + 3 + 2 = 12$ maneiras de substituir as peças cinzas em uma fileira medindo cinco unidades de comprimento.

De quantas maneiras diferentes os ladrilhos cinzas em uma fileira medindo cinquenta unidades de comprimento podem ser substituídos se as cores não podem ser misturadas e pelo menos um ladrilho colorido deve ser usado?

NOTA: Isto está relacionado ao Problema 117 .

116.1 Solução em Python

Solução obtida de <https://www.nayuki.io/page/project-euler-solutions>

```
def compute():  
    LENGTH = 50
```

```

    return str(sum(count_ways(LENGTH, i) for i in range(2, 5)))
def count_ways(length, m):
    ways = [1] + [0] * length
    for n in range(1, len(ways)):
        ways[n] = ways[n]+ways[n - 1]
        if n >= m:
            ways[n] = ways[n]+ways[n - m]
    return ways[-1] - 1
print(compute())

```

116.2 Solução em J

```

count_ways=: 4 : 0
    ways=.(1$1),x$0
    for_n. 1+i._1+$ways do.
        ways=.(n{ways)+((n-1){ways)) n}ways
        if. n >: y do.
            ways=.(n{ways)+((n-y){ways)) n}ways
        end.
    end.
    _1+_1{.ways
)
euler116=: 3 : 0
    length=.y
    aa=. length count_ways 2
    bb=. length count_ways 3
    cc=. length count_ways 4
    aa+bb+cc
)

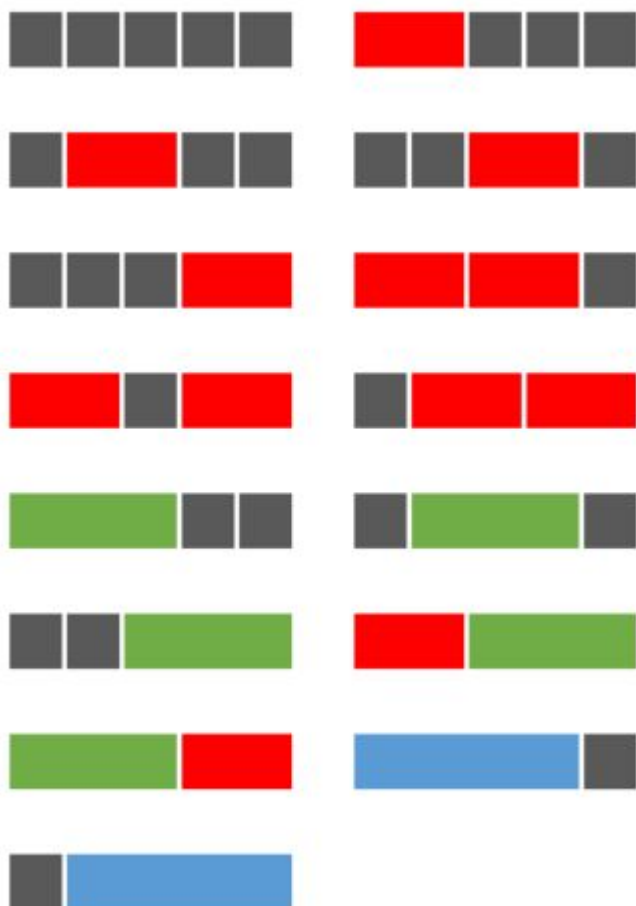
load 'c:/p/quarta/euler116.ijs'
euler116 5
12
euler116 50
20492570929

```

Capítulo 117

Problema 117 - Azulejos vermelhos, verdes e azuis

Usando uma combinação de peças quadradas cinzas e peças retangulares escolhidas entre: peças vermelhas (medindo duas unidades), peças verdes (medindo três unidades) e peças azuis (medindo quatro unidades), é possível ladrilhar uma fileira medindo cinco unidades de comprimento de exatamente quinze maneiras diferentes.



De quantas maneiras uma fileira medindo cinquenta unidades de comprimento pode ser ladrilhada?
NOTA: Isto está relacionado ao Problema 116 .

117.1 Solução em Python

Solução apanhada em <https://www.nayuki.io/page/project-euler-solutions>.

```
def compute():  
    LENGTH = 50  
    ways = [1] + [0] * LENGTH  
    for n in range(1, len(ways)):
```

```
    ways[n] = ways[n]+ sum(ways[max(n - 4, 0) : n])
    return str(ways[-1])
print(compute())
100808458960497
```

117.2 Solução em J

```
euler117=: 3 : 0
    length=.y
    ways=.(1$1),length$0
    n=.1
    while. n<$ways do.
        a=.(n-4)>.0
        ind=.a+i.4
        ways=.(+/ind{ways)+(n{ways)) n}ways
        n=.n+1
    end.
    _1{.ways
)
```

```
    load 'c:/p/quarta/euler117.ijs'
    euler117 5
15
    euler117 50
100808458960497
```

Capítulo 118

Problema 118 - Conjuntos primos pandigitais

Usando todos os dígitos de 1 a 9 e concatenando-os livremente para formar inteiros decimais, diferentes conjuntos podem ser formados. Curiosamente com o conjunto {2, 5, 47, 89, 631}, todos os elementos pertencentes a ele são primos.

Quantos conjuntos distintos contendo cada um dos dígitos de um a nove exatamente uma vez contêm apenas elementos primos?

Uma observação importante é que este problema eu não cheguei a entender. Dito isto, vamos a duas soluções, ambas obtidas do fórum do problema.

118.1 Solução em Python

Do usuário tolstopuz:

```
import math, itertools
primes = [2]
def isprime(p):
    for i in primes:
        if i * i > p:
            return True
        if p % i == 0:
            return False
    return True
x = 3
while x < 100000:
    while not isprime(x):
        x += 2
    primes += [x]
    x += 2
def t(c = 0, s = [], d = set('123456789')):
    if not d:
        c += 1
    else:
        for n in range(1, len(d) + 1):
            for x in itertools.permutations(d, n):
                y = int(''.join(x))
                if y > 1 and (not s or y > s[-1]) and isprime(y):
                    c = t(c, s + [y], d - set(x))
    return c
print(t())
44680
```

118.2 Solução em J

NB. permutations of 1..9 ending in an odd number

```

perms=((2 | {:)"1 # ]) >:(i.@!A.i.)9
inc=-:/:~          NB. is list increasing?
allp=*.:/@:(1&p:)  NB. does list consist only of primes?
NB. Generate boolean masks for partitions
m=(([: inc (9$1) +/;.1~ ])"1#]) 1,.#: i.256
NB. Test a mask and a permutation
f=(allp*inc)@:(10&#.&>)@:(<|.1)"1 f.
+/, m f/ perms

```

```

    load 'c:/p/quarta/euler118.ijs'
    +/, m f/ perms
44680

```

Capítulo 119

Problema 119 - Soma dos dígitos potenciados

O número 512 é interessante porque é igual à soma dos seus algarismos elevada a alguma potência: $5 + 1 + 2 = 8$ e $8^3 = 512$. Outro exemplo de um número com esta propriedade é $614656 = 28^4$.

Nós definiremos a_n como o n termo desta sequência e insistir que um número deve conter pelo menos dois dígitos para ter uma soma.

Você recebeu isso: $a_2 = 512$ e $a_{10} = 614656$. Encontre a_{30} .

119.1 Solução em J

```
quebra=: 3 : 0
  z=: ": y
  ct=:0
  for_i. i.$z do.
    ct=:ct+"i{z
  end.
  ct
)
display=: (1!:2)&2
ui=: 3 : 0
  nn=.(1!:1) 1
)
euler119=: 3 : 0
  n=:0
  c=:9
  while. n<y do.
    d=: quebra c
    dfix=:d
    i=:2
    while. (dfix<:c)*.dfix~:1 do.
      dfix=:d^i
      if. dfix=c do.
        n=:n+1
        display n,c,i
        ui ''
        break.
      end.
      i=:i+1
    end.
    c=:c+1
  end.
  c-1
)
```

Esta solução saiu direto da definição. No começo ela vai bem, mas por volta do 12º elemento da lista ela começa a demorar cada vez mais. Depois de perto de 1h de processamento este programa só havia chegado ao $n = 18$. Não ia dar.

```

1 81 2
2 512 3
3 2401 4
4 4913 3
5 5832 3
6 17576 3
7 19683 3
8 234256 4
9 390625 4
10 614656 4
11 1679616 4
12 17210368 5
13 34012224 6
14 52521875 5
15 60466176 5

```

Mas, com a listagem acima, fui na enciclopedia de sequências inteiras <https://oeis.org/> e pesquisei a sequência. Nesta página ela vai até $n = 21$ (é pouco), mas existe a referência de David W. Wilson que vai até $n = 1137$, mais do que suficiente. A resposta é 248155780267521. Tendo ganhado acesso ao fórum, olhe que solução maravilhosa do usuário dansanders, feito em Python, devidamente completada e modificada por mim.

Note que a solução inverte o problema, a saber

- A busca direta é considerar a sequência de naturais e para cada um, pesquisar se ele é algum elevado da soma dos seus dígitos. Demora inaceitável, já que apenas uma parcela ínfima de naturais obedece a esta lei de formação. Perde-se um tempo enorme fazendo verificações desnecessárias (ou melhor dizendo altamente improváveis).
- Uma estratégia muito melhor é pesquisar os números que são resultados das combinações razoáveis (em tamanho) de seus componentes elevados à alguma potência. A questão aqui, é que os números não são gerados na ordem correta (a que eles ocupam na sequência natural). Mas, como são poucos é fácil achar a ordem: basta ordená-los em ordem crescente). É isso que o programa em Python faz, com resultados, como já se disse, espetacular.

119.2 Solução em Python

```

def sumdigits(n):
    aa=str(n)
    c=0
    for i in range(len(aa)):
        c=c+int(aa[i])
    return c
r = []
for b in range(1, 100):
    for p in range(2, 11):
        a = b ** p
        s = sumdigits(a)
        if a > 9 and s == b:
            print (a, "=", s, "^", p)
            r.append(a)
r.sort()
for i in range(31):
    print(i+1,r[i])

```

Capítulo 120

Problema 120 - Restos quadrados

Seja r o resto quando $(a-1)^n + (a+1)^n$ é dividido por a^2 . Por exemplo, se $a = 7$ e $n = 3$ então $6^3 + 8^3 = 728$ que dividido por $7^2 = 49$ deixa como resto 42. n varia, assim como r , mas para $a = 7$ o valor máximo de r é 42. Para $3 \leq a \leq 1000$ ache $\sum r_{max}$.

Como não poderia deixar de ser, a solução deste problema passa pela “cozinhação” dos dados e das fórmulas.

Busquei inspiração em <https://www.nayuki.io/page/project-euler-solutions> e lá, anexo aos códigos existe um comentário meio cifrado, mas vamos lá.

A expressão é

$$\text{mod} \frac{(a+1)^n + (a-1)^n}{a^2}$$

Vale ressaltar que o que nos interessa é o módulo desta divisão e não o seu resultado numérico. Depois de alguma “cozinha” (veja lá no original) chega-se à conclusão de quando a é par, o resto máximo é $a(a-2)$ e quando a é ímpar, o resto máximo é $a(a-1)$, e daí a solução em Python é

120.1 Solução em Python

```
def euler120():
    soma=0
    for i in range(3, 1001):
        if 0==i%2:
            parimpar=2
        else:
            parimpar=1
        soma=soma + (i*(i-parimpar))
    return soma
print(euler120())

333082500
```

120.2 Solução em J

```
euler120=: 3 : 0
  i=.3
  su=.0
  while. i<1001 do.
    if. 0=2|i do.
      su=.su+i*i-2
    else.
      su=.su+i*i-1
    end.
    i=.i+1
  end.
```

```
su  
)
```

```
    load 'c:/quarta/euler120.ijs'  
    euler120''  
333082500
```

Capítulo 121

Problema 121 - Fundo de prêmios do jogo de disco

Uma bolsa contém um disco vermelho e um disco azul. Em um jogo de azar, um jogador pega um disco aleatoriamente e sua cor é anotada. Após cada turno, o disco é devolvido à bolsa, um disco vermelho extra é adicionado e outro disco é pego aleatoriamente.

O jogador paga £ 1 para jogar e ganha se tiver pegado mais discos azuis do que vermelhos no final do jogo.

Se o jogo for jogado por quatro turnos, a probabilidade de um jogador vencer é exatamente $11/120$, e então o prêmio máximo que o banqueiro deve alocar para vencer neste jogo seria £ 10 antes que eles esperassem incorrer em uma perda. Observe que qualquer pagamento será um número inteiro de libras e também inclui o £ 1 original pago para jogar o jogo, então no exemplo dado o jogador realmente ganha £ 9.

Encontre o prêmio máximo que deve ser alocado para um único jogo em que quinze turnos são jogados.

Uma possível solução A seguir uma explicação retirada de

<https://www.nayuki.io/page/project-euler-solutions>. dentro de p121.py: No início da rodada número k existem $k + 2$ discos (com origem de numeração em zero). Ou seja:

Na rodada 0 (a primeira), existem 2 discos ($1v+1a$)

Na rodada 1 (a segunda), existem 3 discos ($2v+1a$)

e assim por diante.

Um jogo com n rodadas tem $(n + 1) \times n \times (n - 1) \times \dots \times 1 = (n + 1)!$ resultados.

Seja $f(k, b)$ o número de quantas maneiras há de acumular exatamente b discos azuis depois de k rodadas.

$$f(0, 0) = 1$$

$$f(0, b) = 0, \text{ para } b > 0$$

$$f(k, 0) = k \times f(k - 1, 0), \text{ para } k > 0.$$

Adicionando um disco vermelho (onde existem k maneiras)

$$f(k, b) = f(k - 1, b - 1) + k \times f(k - 1, b), \text{ para } k > 0, b > 0.$$

Adicionando um disco azul (1 maneira) ou adicionando um disco vermelho (k maneiras)

A seguir, calcula-se a soma $f(n, k) + f(n, k + 1) + \dots + f(n, n)$ onde k é o menor número de discos azuis acumulados que excedem o número de discos vermelhos acumulados (que é $n - k$). Então $j = \lceil ((n + 1)/2) \rceil$.

Finalmente, a probabilidade de ganhar é aquela soma dividida por $(n + 1)!$.

Para qualquer jogo onde o custo de jogar é 1 e a probabilidade de ganhar é p o prêmio máximo sustentável é $1/p$, daqui o prêmio máximo sustentável inteiro é $\lfloor (1/p) \rfloor$.

121.1 Solução em Python

```
import math
def euler121():
    TURNS = 15
    ways = [[1]]
    for i in range(1, TURNS + 1):
        row = []
        for j in range(i + 1):
            temp = 0
            if j < i:
                temp = ways[i - 1][j] * i
            if j > 0:
                temp = temp + ways[i - 1][j - 1]
```

```
        row.append(temp)
    ways.append(row)
    numer=0
    for i in range(TURNS //2 +1, TURNS+1):
        numer=numer+ways[TURNS][i]
    denom = math.factorial(TURNS + 1)
    return str(denom // numer)
print(euler121())
```

2269

Capítulo 122

Problema 122 - Exponenciação eficiente

Suponha que você está trabalhando em um ambiente de programação onde não existe a função de exponenciação de expoentes inteiros ($y = x^n$), onde n é sempre um número inteiro maior que 0.

Uma possibilidade simples e prática é escrever a função de exponenciação como segue (em C)

```
1: float EXPO(float X, int N) {  
2:   int CT = 1;  
3:   float RES = 1;  
4:   WHILE (CT <= N) {  
5:     RES = RES * X;  
6:     CT = CT + 1;  
7:   }  
8:   RETURN RES;  
9: }
```

Matematicamente e imaginando um exemplo, (seja n^{15}), isto equivale a fazer:

$$n \times n \times n \times \dots \times n = n^{15}$$

ou seja equivale a fazer 14 multiplicações.

Entretanto, para calcular esse mesmo n^{15} , pode usar um método “binário” e achar o mesmo resultado com seis multiplicações:

$$\begin{aligned} n \times n &= n^2 \\ n^2 \times n^2 &= n^4 \\ n^4 \times n^4 &= n^8 \\ n^8 \times n^4 &= n^{12} \\ n^{12} \times n^2 &= n^{14} \\ n^{14} \times n &= n^{15} \end{aligned}$$

Mas, ainda é possível melhorar isso, usando apenas 5 multiplicações:

$$\begin{aligned} n \times n &= n^2 \\ n^2 \times n &= n^3 \\ n^3 \times n^3 &= n^6 \\ n^6 \times n^6 &= n^{12} \\ n^{12} \times n^3 &= n^{15} \end{aligned}$$

A seguir vai-se definir a função $m(k)$ como sendo o número mínimo de multiplicações necessárias para calcular n^k . Usando o exemplo acima, ter-se-ia $m(15) = 5$. Ache

$$\sum_{k=1}^{200} m(k)$$

A resposta é 1582.

122.1 Solução em APL

```
vr←euler122j;aa;cand;i;ind;j;onde;prof;prox;px;seq;velho;x  
[1] x←1000 17p0
```

```

[2] x[1;]←1 -1 0 0,13p0 A chave, &pai, qtdade de ciclos, 88=fechado, -1=aberto
[3] x[2;]←2 1 1 -1 1,12p0
[4] prox←3
[5] A ta dando erro no 77 (1-2-3-5-7-14-19-38-76-77) e o menor e
[6] A 1-2-4-8-9-17-34-43-77: da um a menos. Idem para o 154 (que pega
[7] A carona no 77). A solucao para este erro e guardar todos os minimos
[8] A e nao so o primeiro.
[9] t1:A→(prox>990)/acabou
[10] →(200=+/(1200)∈x[;1])/acabou
[11] velho←cand←1↑(x[;4]=-1)/11000
[12] A 'velho=',⌘velho
[13] prof←x[velho;3]
[14] A 'prof=',⌘prof
[15] seq←10
[16] aba:→(0=pseq)/fr
[17] →(seq[pseq]=1)/ter
[18] fr:seq←seq,x[cand;1]
[19] cand←x[cand;2]
[20] →aba
[21] ter:
[22] A 'seq=',⌘seq
[23] ind←pseq
[24] f4:→(0=ind)/lk
[25] →(1=v/aa←(seq[1]+seq[ind])=x[;1])/pula
[26] x[prox;]←((seq[1]+seq[ind]),velho,(prof+1),-1),(12↑seq),ind
[27] prox←prox+1
[28] →ikj
[29] pula:
[30] onde←aa/11↑px A so para ter certeza
[31] →(x[onde;3]≤prof+1)/ikj A que a logica esta certa
[32] 'olha aqui ',⌘onde,prox A ele nao entra aqui
[33] ikj:A nenhuma vez
[34] ind←ind-1
[35] →f4
[36] lk:x[velho;4]←88 A era 0
[37] →t1
[38]
[39] acabou:r←x[1prox-1;]
[40]
[41] A o jeito correto de achar a solucao e:
[42] A a←euler122j 800 (ou 900) por ai...
[43] A b←a[1] - pegar so a matriz gerada
[44] A pb - da 322 (para o caso de 800)
[45] A +/b[(b[;1]∈1200)/1322;3] - col3 eh a coluna onde esta a qtd de operacoes
[46] A a linha acima deu 1584. Lembrar que ha que retirar 2 por causa do 77
[47] A e do 154 e a resposta correta e 1582.
[48] A 1 numera 5 0 4 0 3 0 3 0 10 0 4 0 4 0 3 0 3 0 3 0
3 0 3 0 3 0 3 0 3 0 3 0b
[49]

```

Capítulo 123

Problema 123 - Restos de quadrados primos

Seja p_n o n -ésimo primo: 2,3,5,7,11,... e seja r o resto quando $(p_n - 1)^n + (p_n + 1)^n$ é dividido por p_n^2 . Por exemplo, quando $n = 3$, $p_3 = 5$ e $4^3 + 6^3 = 280 \equiv 5 \pmod{25}$. O menor valor de n para o qual o resto acima excede primeiro 10^9 é 7037. Encontre o menor valor de n para o qual o resto excede primeiro 10^{10} .

A resposta é 21035.

Usando-se a teoria subjacente ao problema 120, sabe-se que $(a - 1)^n + (a + 1)^n \pmod{a^2}$ é igual a 2 se n é par e é $2an$ se n é ímpar. Já que 2 pode ser desprezado por muito pequeno, só nos interessam os casos ímpares. a é o n -ésimo primo (com origem dos índices 1) então $a > n$. De facto, para $n \geq 5$ tem-se $a > 2n$ e então pode-se pegar $2an$ diretamente sem precisar pegar o módulo por a^2 . Daqui a solução

123.1 Solução em J

```
euler123 =: 3 : 0
  ind=:0
  primo:=(_1 p: 1000000)$0
  while. ind<$primo do.
    primo:= (p: ind) ind}primo
    ind=:ind+1
  end.
  n=:5
  while. n<$primo do.
    rem=:n*2*(n-1){primo
    if. rem>10^10 do.
      n
      return.
    end.
    n=:n+2
  end.
)
```

```
load 'c:/p/quarta/euler123.ijs'
euler123 ''
21035
```

Capítulo 124

Problema 124 - Radicais ordenados

O *radical* de n , $rad(n)$ é o produto dos fatores primos distintos de n . Por exemplo $504 = 2^3 \times 3^2 \times 7$, então $rad(504) = 2 \times 3 \times 7 = 42$. Se calcularmos $rad(n)$ para $1 \leq n \leq 10$, classificando-os por $rad(n)$ e desempatando em n se os $rad(n)$ forem iguais, obtem-se:

não classificado		classificado		
n	$rad(n)$	n	$rad(n)$	k
1	1	1	1	1
2	2	2	2	2
3	3	4	2	3
4	2	8	2	4
5	5	3	3	5
6	6	9	3	6
7	7	5	5	7
8	2	6	6	8
9	3	7	7	9
10	10	10	10	10

Seja $E(k)$ o k -ésimo elemento no resultado ordenado na linha k , por exemplo $E(4) = 8$ e $E(6) = 9$. Se $rad(n)$ está ordenado para $1 \leq n \leq 100000$, ache $E(10000)$.

A solução é 21417.

Buscado em <https://www.nayuki.io/page/project-euler-solutions>. A solução publicada lá em Python é

124.1 Solução em Python

```
def euler124():
    LIMIT = 100000
    rads = [0] + [1] * LIMIT
    for i in range(2, len(rads)):
        if rads[i] == 1:
            for j in range(i, len(rads), i):
                rads[j] = rads[j]*i
    data = sorted((rad, i) for (i, rad) in enumerate(rads))
    return str(data[10000][1])
print(euler124())
```

21417

Ai eu tentei transcrever para J, e fui até a ordenação, onde aparentemente empaquei

124.2 Solução em J

```
euler124=: 3 : 0
    limite=: 100000
```

```

rads=:0,limite$1
i=:2
while. i<$rads do.
  if. i{rads = 1 do.
    j=:i
    while. j<$rads do.
      rads=: (j{rads * i) j}rads
      j=:j+i
    end.
  end.
  i=:i+1
end.
data=:(/:rads){rads
10000{data
)

load 'c:/p/quarta/euler124.ijs'
euler124 ''
1947 // ESTE RESULTADO ESTA ERRADO //
```

Capítulo 125

Problema 125 - Somas palindrômicas

O número palindrômico 595 é interessante porque ele pode ser escrito como a soma de quadrados consecutivos $6^2 + 7^2 + 8^2 + 9^2 + 10^2 + 11^2 + 12^2$. Há exatos 11 palíndromos abaixo de 1000 que podem ser escritos como somas de quadrados consecutivos e a soma desses palíndromos é 4164. Note que $1 = 0^2 + 1^2$ não foi incluído neste problema. Ache a soma de todos os números menores do que 10^8 que são palíndromos e também podem ser escritos como soma de quadrados consecutivos.

A resposta é 2906969179.

Obtido de <https://www.nayuki.io/page/project-euler-solutions>.

125.1 Solução em Python

```
import itertools
def euler125():
    nums = set()
    for i in range(1, 10001):
        sigma = i * i
        for j in itertools.count(i + 1):
            sigma = sigma + j * j
            if sigma >= 100000000:
                break
            s = str(sigma)
            if s == s[::-1]: # Is palindrome
                nums.add(sigma)
    return str(sum(nums))

print(euler125())

2906969179
```

125.2 Solução em J

```
display=: (1!:)2

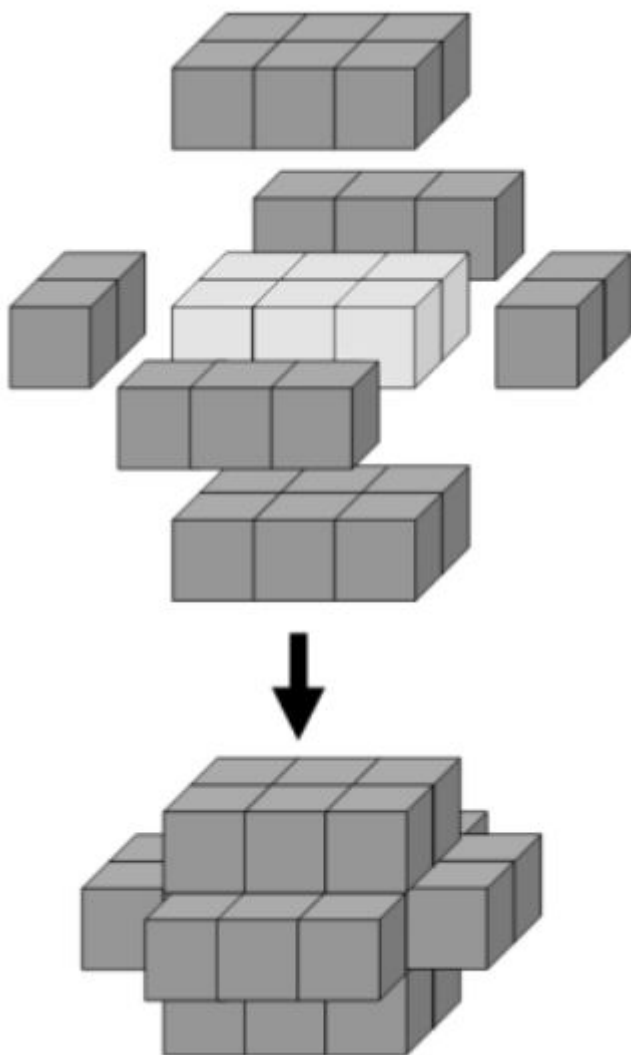
euler125=: 3 : 0
    nums=:0$0
    i=:1
    while. i<10001 do.
        sigma=:i*i
        j=:i+1
        while. j<_ do.
            sigma=:sigma+j*j
            if.sigma>: 10^8 do.
                break.
            
```

```
end.  
s=":sigma  
if. (($s)=+ /s=|.s) *. -.sigma e. nums do.  
  nums=:nums,sigma  
  display sigma  
end.  
j=:j+1  
end.  
i=:i+1  
end.  
+/nums  
)
```

```
load'c:/p/quarta/euler125.ijs'  
euler125''  
... muitos valores ...  
2906969179
```

Problema 126 - Camadas em cubóide

O número mínimo de cubos para cobrir qualquer face visível de um cubóide medindo $3 \times 2 \times 1$ é 22:



Se adicionarmos uma segunda camada a esse sólido, serão necessários quarenta e seis cubos para cobrir todas as faces visíveis, a terceira camada exigirá setenta e oito cubos e a quarta camada exigirá cento e dezoito cubos para cobrir todas as faces visíveis.

No entanto, a primeira camada em um cuboide medindo $5 \times 1 \times 1$ também requer vinte e dois cubos; da mesma forma, a primeira camada em cubóides medindo $5 \times 3 \times 1$, $7 \times 2 \times 1$ e $11 \times 1 \times 1$, todos contêm 46 cubos.

Define-se $C(n)$ para representar o número de cubóides que contêm n cubos em uma de suas camadas. Então $C(22) = 2$, $C(46) = 4$, $C(78) = 5$ e $C(118) = 8$.

Tem-se que 154 é o menor valor de n , para o qual $C(n) = 10$.

Encontre o menor valor de n para o qual $C(n) = 1000$.

Índice Remissivo

- 10001^o número primo, 36
- 11 ambientes, 11
- a milionésima permutação de 0..9, 84
- anel 5-gon mágico, 186
- APL, 11
- auto potências, 133
- azulejos vermelhos, verdes e azuis, 292
- azulejos vermelhos, verdes ou azuis, 290
- C++, 11
- cadeia de dígitos no fatorial, 196
- cadeias amigáveis, 247
- camadas em cubóide, 309
- caminhos na treliça, 61
- Clipper, 11
- combinações de blocos de contagem I, 287
- combinações de blocos de contagem II, 289
- conjuntos de pares de primos, 163
- conjuntos primos pandigitais, 294
- constante de Champernowne, 117
- contador do número de letras em nomes de números, 65
- contagens de dígitos poderosas, 173
- contando frações, 192
- contando frações em intervalo, 194
- contando retângulos, 220
- contando somas, 200
- contenção de triângulo, 264
- convergência da raiz quadrada, 156
- convergência de e , 178
- correntes de dígitos quadrados, 237
- CPC, 11
- dardos, 278
- decriptação XOR, 161
- deepseek, 11
- derivação de senha, 206
- dez dígitos iniciais da soma de 100 números de 50 dígitos cada, 56
- diferença entre somas, 33
- diofantinas recíprocas II, 280
- divisibilidade de substrings, 123
- equação diofantina, 181
- escores de nomes, 79
- euler, 11
- expansão digital da raiz quadrada, 207
- exponenciação eficiente, 302
- expressões aritméticas, 239
- fatores primos distintos, 131
- fatoriais dos dígitos, 106
- frações com cancelamento de dígitos, 103
- frações ordenadas, 191
- fundo de prêmios do jogo de disco, 300
- geminis, 11
- Harbour, 11
- J, 11
- Java, 12
- Javascript, 11
- Lisp, 11
- Lychrel, 152
- maior ciclo de primos quadráticos, 90
- maior ciclo recorrente de inversa, 87
- maior exponencial, 257
- maior fator primo, 23
- maior palindrômico de produto, 26
- maior primo não mersenne, 253
- maior produto de 4 dígitos adjacentes, 39
- maior sequência de Collatz, 59
- maior soma de 4 vizinhos em 9 direções, 49
- maneiras de obter 2 libras, 99
- Maple, 12
- Matlab, 11
- menor inteiro dividido sem resto, 29
- mãos de poker, 146
- múltiplos, 13
- múltiplos pandigitais, 114
- múltiplos permutados, 142
- numerais romanos, 229
- número de Fibonacci pandigital no fim, 268
- números amigos, 76
- números de Lychrel, 152
- números figurativos cíclicos, 166
- números iguais à soma da potência quinta de seus dígitos, 97
- números não rebotados, 286
- números pentagonais, 125
- números rebotados, 284
- números soma e produto, 226
- Octave, 11
- outra conjectura de Goldbach, 129
- palíndromos em base dupla, 110
- pares de dígitos do cubo, 231
- partições de moedas, 204

permutação de totientes, 189
 permutações cúbicas, 171
 permutações de primos, 136
 período ímpar de raízes quadradas, 175
 polinômio ótimo, 260
 potência dos dígitos somada, 154
 primeiro triangular com mais de 500 divisores, 52
 primo pandigital, 119
 primos circulares, 108
 primos com corridas, 281
 primos espirais, 158
 primos truncáveis, 112
 probabilidade arranjada, 258
 probabilidades do banco imobiliário, 217
 problema 1, 13
 problema 10, 47
 problema 100, 258
 problema 101, 260
 problema 102, 264
 problema 103, 266
 problema 104, 268
 problema 107, 273
 problema 108, 276
 problema 109, 278
 problema 11, 49
 problema 110, 280
 problema 111, 281
 problema 112, 284
 problema 113, 286
 problema 114, 287
 problema 115, 289
 problema 116, 290
 problema 117, 292
 problema 118, 294
 problema 119, 296
 problema 12, 52
 problema 120, 298
 problema 121, 300
 problema 122, 302
 problema 123, 304
 problema 124, 305
 problema 125, 307
 problema 126, 309
 problema 13, 56
 problema 14, 59
 problema 15, 61
 problema 16, 63
 problema 17, 65
 problema 18, 66
 problema 19, 69
 problema 2, 19
 problema 20, 75
 problema 21, 76
 problema 22, 79
 problema 23, 81
 problema 24, 84
 problema 25, 85
 problema 26, 87
 problema 28, 90, 92
 problema 29, 94
 problema 3, 23
 problema 30, 97
 problema 31, 99

problema 32, 101
 problema 33, 103
 problema 34, 106
 problema 35, 108
 problema 36, 110
 problema 37, 112
 problema 38, 114
 problema 39, 116
 problema 4, 26
 problema 40, 117
 problema 41, 119
 problema 42, 121
 problema 43, 123
 problema 44, 125
 problema 45, 127
 problema 46, 129
 problema 47, 131
 problema 48, 133
 problema 49, 136
 problema 5, 29
 problema 50, 138
 problema 51, 140
 problema 52, 142
 problema 53, 144
 problema 54, 146
 problema 55, 152
 problema 56, 154
 problema 57, 156
 problema 58, 158
 problema 59, 161
 problema 6, 33
 problema 60, 163
 problema 61, 166
 problema 62, 171
 problema 63, 173
 problema 64, 175
 problema 65, 178
 problema 66, 181
 problema 67, 183
 problema 68, 186
 problema 69, 188
 problema 7, 36
 problema 70, 189
 problema 71, 191
 problema 72, 192
 problema 73, 194
 problema 74, 196
 problema 75, 198
 problema 76, 200
 problema 77, 202
 problema 78, 204
 problema 79, 206
 problema 8, 39
 problema 80, 207
 problema 81, 210
 problema 82, 212
 problema 83, 214
 problema 84, 217
 problema 85, 220
 problema 86, 222
 problema 87, 224
 problema 88, 226
 problema 89, 229

problema 9, 42
problema 90, 231
problema 91, 233, 237
problema 93, 239
problema 94, 242
problema 95, 247
problema 96, 250
problema 97, 253
problema 98, 255
problema 99, 257
produtos pandigitais, 101
Python, 11

quadrados anagrâmicos, 255
quantos domingos caíram no 1^o do mês, 69

radicais ordenados, 305
recursão, 11
recíprocas diofantinas I, 276
rede mínima, 273
restos de quadrados primos, 304
restos quadrados, 298
rota cubóide, 222

seleções combinatórias, 144
sequência de Collatz, 59
soma das diagonais, 92
soma de não abundantes, 81
soma de primos consecutivos, 138
soma de termos fibonacci, 19
soma do caminho máximo I em triângulo numérico, 66
soma do caminho máximo II, 183
soma do caminho: 4 caminhos, 214
soma do caminho: três caminhos, 212
soma dos caminhos: duas maneiras, 210
soma dos dígitos de uma potência, 63
soma dos dígitos do fatorial de 100, 75
soma dos dígitos potenciados, 296
soma dos primos abaixo de 2 milhões, 47
somadas de conjuntos especiais ótimas, 266
somadas palindrômicas, 307
sudoku, 250
sumarizações de primos, 202

termo fibonacci a ter mais de 1000 dígitos, 85
termos distintos em $a*b$ e $b*a$, 94
thinking apl, 11
TIOBE, 11
totiente máximo, 188
triangular, pentagonal e hexagonal, 127
triangulares codificados, 121
tripla pitagoreana com $a + b + c = 1000$, 42
triplas potências de primos, 224
triângulos quase equiláteros, 242
triângulos retângulos com coordenadas inteiras, 233
triângulos retângulos inteiros, 116
triângulos retângulos inteiros singulares, 198
troca de dígitos em primos, 140