U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75789 ARIEL
GUSTAVO RODRIGUES DA S
24PRG203 - 1 entregar ate 11/jun
/ _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 8
3: enquanto m < 29
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 7 ate 28 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 16
4:     para k de 5 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 19 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 16
4:     k ← 6
5:     enquanto k < 8
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 4 a 16 passo 3
3:     para  k de 6 a 08 passo 3
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 2
3: enquanto j < 13
4:     k ← 13
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 4
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 6
3: enquanto m < 29
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 7 ate 28 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 7
3: enquanto m < 18
4:     para k de 4 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 2
3: enquanto m < 20 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

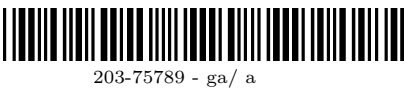
```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 12
4:     k ← 7
5:     enquanto k < 10
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.



203-75789 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75796
CASSIANO MAGNO CHAGAS E
SA 24PRG203 - 2 entregar ate
11/jun _____ / _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 8
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 5 ate 29 passo 6
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 17
4:     para k de 3 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 3
3: enquanto m < 20 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 3
3: enquanto j < 14
4:     k ← 7
5:     enquanto k < 9
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 4 a 14 passo 4
3:     para  k de 7 a 09 passo 2
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 4
3: enquanto j < 16
4:     k ← 12
5:     enquanto k > 7
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 5
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 5 ate 29 passo 6
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 5
3: enquanto m < 16
4:     para k de 3 ate 8 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 2
3: enquanto m < 16 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

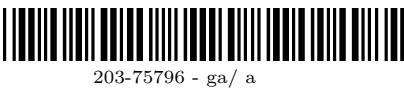
```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 15
4:     k ← 6
5:     enquanto k < 9
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.



203-75796 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75808
CHRISTOPHER BUCH FILIPAK
24PRG203 - 3 entregar ate 11/jun / _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 7 ate 28 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 7
3: enquanto m < 19
4:     para k de 3 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 3
3: enquanto m < 18 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 16
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 3 a 14 passo 4
3:     para  k de 7 a 09 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 4
3: enquanto j < 13
4:     k ← 13
5:     enquanto k > 7
6:         escreva j + k
7:         k ← k - 2
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 5
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 7 ate 26 passo 6
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 8
3: enquanto m < 19
4:     para k de 3 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 4
3: enquanto m < 16 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 14
4:     k ← 7
5:     enquanto k < 9
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75808 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75815 ERAN
MARTINEZ RAMOS 24PRG203 - 4
entregar ate 11/jun _____ / / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 27
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 17
4:     para k de 5 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 4
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 4
3: enquanto j < 14
4:     k ← 6
5:     enquanto k < 10
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 3 a 12 passo 4
3:     para  k de 7 a 10 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 3
3: enquanto j < 14
4:     k ← 11
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 5
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 7 ate 30 passo 3
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 6
3: enquanto m < 19
4:     para k de 5 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 2
3: enquanto m < 19 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 16
4:     k ← 6
5:     enquanto k < 8
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75815 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75822
HENRIQUE IVANKIO GUSSE
24PRG203 - 5 entregar ate 11/jun
/ _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 5
3: enquanto m < 30
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 7 ate 30 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 6
3: enquanto m < 18
4:     para k de 4 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 4
3: enquanto m < 17 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 14
4:     k ← 6
5:     enquanto k < 9
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 3 a 13 passo 3
3:     para  k de 5 a 10 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 3
3: enquanto j < 12
4:     k ← 11
5:     enquanto k > 8
6:         escreva j + k
7:         k ← k - 4
8:     fim{enquanto}
9:     j ← j + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 8
3: enquanto m < 29
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 7 ate 26 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 6
3: enquanto m < 18
4:     para k de 3 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 4
3: enquanto m < 18 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

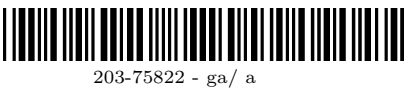
```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 9
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.



203-75822 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75839 IAN
DOLABELLA OHATA 24PRG203 -
6 entregar ate 11/jun _____ /
/ _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 6
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 6 ate 28 passo 3
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 17
4:     para k de 3 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 4
3: enquanto m < 16 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 3
3: enquanto j < 12
4:     k ← 7
5:     enquanto k < 10
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 4 a 12 passo 3
3:     para  k de 5 a 10 passo 2
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 3
3: enquanto j < 15
4:     k ← 11
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 6 ate 28 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 5
3: enquanto m < 17
4:     para k de 3 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 4
3: enquanto m < 19 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 12
4:     k ← 6
5:     enquanto k < 8
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75839 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75846 JOAO
BONFIN LINO 24PRG203 - 7
entregar ate 11/jun _____ /
/ _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 5 ate 30 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 16
4:     para k de 5 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 3
3: enquanto m < 18 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 3
3: enquanto j < 15
4:     k ← 6
5:     enquanto k < 9
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 2 a 12 passo 2
3:     para  k de 7 a 08 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 3
3: enquanto j < 13
4:     k ← 11
5:     enquanto k > 7
6:         escreva j + k
7:         k ← k - 2
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 8
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 8 ate 27 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 5
3: enquanto m < 17
4:     para k de 3 ate 7 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 3
3: enquanto m < 19 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 15
4:     k ← 6
5:     enquanto k < 8
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75846 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75853 JOAO LUCA FERNANDES DE OLIVE
24PRG203 - 8 entregar ate 11/jun / _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 29
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 5 ate 27 passo 3
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 19
4:     para k de 5 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 4
3: enquanto j < 12
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 2 a 12 passo 4
3:     para  k de 5 a 10 passo 2
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 4
3: enquanto j < 14
4:     k ← 12
5:     enquanto k > 8
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 6
3: enquanto m < 27
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 8 ate 27 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 6
3: enquanto m < 16
4:     para k de 4 ate 7 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 4
3: enquanto m < 16 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 13
4:     k ← 7
5:     enquanto k < 9
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75853 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75860 JOAO
PEDRO DE BRITO DUARTE
24PRG203 - 9 entregar ate 11/jun
/ _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 27
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 7 ate 30 passo 6
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 7
3: enquanto m < 17
4:     para k de 4 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 4
3: enquanto m < 17 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 3
3: enquanto j < 14
4:     k ← 7
5:     enquanto k < 9
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 3 a 14 passo 3
3:     para  k de 6 a 08 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 2
3: enquanto j < 14
4:     k ← 11
5:     enquanto k > 8
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 6
3: enquanto m < 27
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 7 ate 29 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 6
3: enquanto m < 16
4:     para k de 5 ate 8 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 2
3: enquanto m < 20 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 4
3: enquanto j < 13
4:     k ← 6
5:     enquanto k < 10
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75860 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75877 JOAO PEDRO MARIANO SUEKE
24PRG203 - 10 entregar ate 11/jun / _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 7 ate 30 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 3 ate 8 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 16 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 4
3: enquanto j < 16
4:     k ← 7
5:     enquanto k < 9
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 2 a 15 passo 4
3:     para  k de 7 a 08 passo 2
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 2
3: enquanto j < 14
4:     k ← 11
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 5
3: enquanto m < 30
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 6 ate 26 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 7
3: enquanto m < 20
4:     para k de 5 ate 9 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 3
3: enquanto m < 19 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 3
3: enquanto j < 13
4:     k ← 7
5:     enquanto k < 9
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75877 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75884 JOAO VICTOR GOMES VILELA GER 24PRG203 - 11 entregar ate 11/jun / _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 8
3: enquanto m < 30
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 6 ate 28 passo 3
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 4 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 4
3: enquanto m < 18 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 16
4:     k ← 6
5:     enquanto k < 8
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 4 a 13 passo 3
3:     para  k de 6 a 08 passo 3
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 4
3: enquanto j < 12
4:     k ← 11
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 4
8:     fim{enquanto}
9:     j ← j + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 7
3: enquanto m < 29
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 5 ate 29 passo 3
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 8
3: enquanto m < 20
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 2
3: enquanto m < 19 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 3
3: enquanto j < 14
4:     k ← 7
5:     enquanto k < 9
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75989 LUCAS ANTOSZCZYSZEN 24PRG203 - 12
entregar ate 11/jun _____ / / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 8
3: enquanto m < 27
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 8 ate 30 passo 6
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 19
4:     para k de 4 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 20 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 13
4:     k ← 6
5:     enquanto k < 9
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 3 a 13 passo 4
3:     para  k de 7 a 09 passo 2
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 4
3: enquanto j < 13
4:     k ← 11
5:     enquanto k > 8
6:         escreva j + k
7:         k ← k - 4
8:     fim{enquanto}
9:     j ← j + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 6
3: enquanto m < 29
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 5 ate 29 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 7
3: enquanto m < 19
4:     para k de 3 ate 7 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 3
3: enquanto m < 19 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75989 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75891 LUIZ FELIPE GUEDES BUCCHERI
24PRG203 - 13 entregar ate 11/jun / _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 6
3: enquanto m < 30
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 7 ate 28 passo 6
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 18
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 20 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 12
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 2 a 12 passo 4
3:     para  k de 6 a 09 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 3
3: enquanto j < 16
4:     k ← 12
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 7
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 8 ate 26 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 8
3: enquanto m < 20
4:     para k de 3 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 3
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 4
3: enquanto j < 16
4:     k ← 6
5:     enquanto k < 10
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75891 - ga/ a

U Positivo - UTFPR - PUCPr – 27/05/2024 - 10:18:11.9
Prof Dr P Kantek (pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75903
MANUELLA LEAL DE MEIRELLES F 24PRG203 - 14
entregar ate 11/jun _____ / / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 6
3: enquanto m < 29
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 7 ate 30 passo 3
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 17
4:     para k de 5 ate 8 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 3
3: enquanto m < 20 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 4
3: enquanto j < 16
4:     k ← 7
5:     enquanto k < 10
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 4 a 12 passo 3
3:     para  k de 6 a 10 passo 3
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 2
3: enquanto j < 14
4:     k ← 11
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 4
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 6
3: enquanto m < 27
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 7 ate 26 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 8
3: enquanto m < 18
4:     para k de 4 ate 7 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 3
3: enquanto m < 18 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 4
3: enquanto j < 12
4:     k ← 6
5:     enquanto k < 9
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75910 MARIA EDUARDA DA SILVA 24PRG203 - 15 entregar ate 11/jun _____ / / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 8
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 6 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 17
4:     para k de 3 ate 7 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 4
3: enquanto m < 17 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 4
3: enquanto j < 16
4:     k ← 6
5:     enquanto k < 8
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 3 a 13 passo 4
3:     para  k de 7 a 09 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 4
3: enquanto j < 12
4:     k ← 13
5:     enquanto k > 8
6:         escreva j + k
7:         k ← k - 4
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 8
3: enquanto m < 29
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 7 ate 30 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 8
3: enquanto m < 18
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 3
3: enquanto m < 18 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 12
4:     k ← 6
5:     enquanto k < 8
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75927
MATHEUS AITA FABRICIO DE
CAR 24PRG203 - 16 entregar ate
11/jun _____ / _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 5
3: enquanto m < 30
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 6 ate 29 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 18
4:     para k de 4 ate 9 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 16 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 4
3: enquanto j < 14
4:     k ← 7
5:     enquanto k < 10
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 4 a 13 passo 3
3:     para  k de 7 a 10 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 4
3: enquanto j < 13
4:     k ← 13
5:     enquanto k > 8
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 7 ate 27 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 6
3: enquanto m < 17
4:     para k de 5 ate 9 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 4
3: enquanto m < 18 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 3
3: enquanto j < 16
4:     k ← 6
5:     enquanto k < 10
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75927 - ga/ a

U Positivo - UTFPR - PUCPr – 27/05/2024 - 10:18:11.9
Prof Dr P Kantek (pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75934
MATHEUS LEANDRO DE BITENCOUR 24PRG203 - 17
entregar ate 11/jun _____ / / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 6 ate 27 passo 3
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 16
4:     para k de 3 ate 9 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 3
3: enquanto m < 17 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 14
4:     k ← 6
5:     enquanto k < 9
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 4 a 13 passo 4
3:     para  k de 6 a 10 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 4
3: enquanto j < 12
4:     k ← 13
5:     enquanto k > 7
6:         escreva j + k
7:         k ← k - 2
8:     fim{enquanto}
9:     j ← j + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 8
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 6 ate 26 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 6
3: enquanto m < 17
4:     para k de 4 ate 9 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 4
3: enquanto m < 16 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75934 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75941 MAYCON TAVARES WOLPE 24PRG203 - 18
entregar ate 11/jun _____ / / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 6
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 6
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 20
4:     para k de 4 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 18 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 12
4:     k ← 6
5:     enquanto k < 10
6:         se k mod 1 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 3 a 14 passo 3
3:     para  k de 6 a 08 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 3
3: enquanto j < 13
4:     k ← 12
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 5
3: enquanto m < 29
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 5 ate 26 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 6
3: enquanto m < 16
4:     para k de 3 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 4
3: enquanto m < 19 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

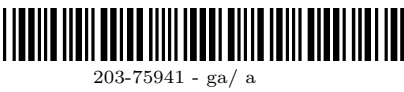
```
1: funcao Exercicio12
2: j ← 3
3: enquanto j < 13
4:     k ← 7
5:     enquanto k < 9
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.



U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75958 PEDRO TOLEDO LEAL 24PRG203 - 19
entregar ate 11/jun _____ / / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 8
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 6 ate 28 passo 3
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 16
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 20 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 3
3: enquanto j < 14
4:     k ← 7
5:     enquanto k < 10
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 2 a 12 passo 3
3:     para  k de 5 a 09 passo 3
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 2
3: enquanto j < 14
4:     k ← 11
5:     enquanto k > 8
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 6
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 5
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 7 ate 27 passo 4
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 8
3: enquanto m < 18
4:     para k de 5 ate 9 passo 2
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 2
3: enquanto m < 16 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 4
3: enquanto j < 12
4:     k ← 6
5:     enquanto k < 10
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

203-75958 - ga/ a

U Positivo - UTFPR - PUCPr –
27/05/2024 - 10:18:11.9
Prof Dr P Kantek
(pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75965 PHELIPE GABRIEL LIMA DA SILV
24PRG203 - 20 entregar ate 11/jun / _____ / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 6
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 6 ate 27 passo 6
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 19
4:     para k de 4 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 3
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 4
3: enquanto m < 20 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 16
4:     k ← 6
5:     enquanto k < 9
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 2 a 15 passo 4
3:     para  k de 5 a 08 passo 2
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 3
3: enquanto j < 14
4:     k ← 13
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 3
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 6
3: enquanto m < 30
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 5 ate 30 passo 6
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 6
3: enquanto m < 19
4:     para k de 3 ate 7 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 2
3: enquanto m < 18 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

```
1: funcao Exercicio12
2: j ← 2
3: enquanto j < 14
4:     k ← 6
5:     enquanto k < 9
6:         se k mod 3 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 4
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.

U Positivo - UTFPR - PUCPr – 27/05/2024 - 10:18:11.9
Prof Dr P Kantek (pkantek@gmail.com)
Estruturas Básicas de Repetição
VIVX023a, V: 2.12 75972 REBECA CABRAL DOS SANTOS 24PRG203 - 21 entregar ate 11/jun _____ / / _____

Comandos de Repetição

Enquanto Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: enquanto condição
2:     comando 1
3:     comando 2
4:     ...
5: fim{enquanto}
```

Quando o comando enquanto é encontrado a condição é avaliada. Se for falsa, ocorre um desvio para o próximo comando após o fimenquanto. Se for verdadeira, os comandos seguintes são executados e quando o comando fimenquanto é achado, ocorre um desvio para o enquanto (e a condição associada é avaliada de novo. Por exemplo:

```
1: J ← 5
2: enquanto J < 16
3:     escreva(J)
4:     J ← J + 3
5: fim{enquanto}
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14.

Repita Serve para iterar (repetir) partes de código de programação dependendo de alguma condição. Seu formato é

```
1: repita
2:     comando 1
3:     comando 2
4:     ...
5: até condição
```

Quando o comando repita é encontrado ele simplesmente é ignorado e os comandos seguintes são executados normalmente. Quando o comando até é achado, a condição que o acompanha é avaliada. Se for verdadeira, ocorre um desvio para o comando seguinte ao do até. Se for falsa, ocorre um desvio para o comando repita no alto. Por exemplo:

```
1: J ← 5
2: repita
3:     escreva(J)
4:     J ← J + 3
5: até J > 16
```

Este trecho dará origem às seguintes impressões: 5, 8, 11 e 14. Note que para obter os mesmos resultados foi necessário inverter o sinal da condição (de < para >). Este comando tal como foi descrito é aquele da linguagem PASCAL e da maioria dos processadores de algoritmos. Para C, o comando é `do { ... } while(condição)`, com o detalhe é que a saída é quando a condição é FALSA.

Para Este comando serve para repetir (iterar) com bastante controle sobre a quantidade de repetições. Só pode ser usado quando se conhece a quantidade de repetições necessária. Seu formato:

```
1: para variável FROM valor-1 TO
   valor-2 [PASSO valor-3]
2:     comando-1
3:     comando-2
4:     ...
5: fim{para}
```

A variável de controle é inicializada com o valor-1. A seguir, ela é testada contra o valor-2. Se a variável é ≤ ao valor, o conjunto de comandos é executado. Senão desvia-se para o comando seguinte ao fimpara. Quando o comando fimpara é encontrado, a variável de controle é incrementada com o valor-3, e ocorre um desvio para o início do comando.

Se o valor-3 é negativo, a variável de controle é decrementada a cada laço e o teste de parar/continuar é: ≥.

Acompanhe no exemplo a seguir:

```
1: para J FROM 4 TO 12 PASSO 3
2:     escreva J
3: fim{para}
```

Serão impressos: 4, 7, 10 e 12. O mesmo programa, agora escrito com enquanto, fica:

```
1: J ← 4
2: enquanto J ≤ 12
3:     escreva J
4:     J ← J + 3
5: fim{enquanto}
```

Alguns exemplos Similares aos exercícios que terão que ser feitos à frente.

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 28
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio2
2: para m de 8 ate 26 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio3
2: m ← 5
3: enquanto m < 17
4:     para k de 5 ate 9 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio4
2: m ← 2
3: enquanto m < 17 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×3)
8:     fim{se}
9:     m ← m + 2
10: fim{enquanto}
11: fimfuncao
```

Cuja resposta é:

```
1: funcao Exercicio5
2: j ← 2
3: enquanto j < 15
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 3 = 1
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 2
14: fim{enquanto}
15: fimfuncao
```

Cuja resposta é:

☞ Para você fazer

Para os exercícios a seguir, descubra a sequencia de números impressos e registre no gabarito o LTIMO deles. Note que todas as séries tem menos de 11 números. Se este número for alcançado, significa que há algum erro de interpretação.

Exercício 1

```
1: funcao Exercicio1
2: m ← 7
3: enquanto m < 27
4:     escreva (m)
5:     m ← m+ 3
6: fim{enquanto}
7: fimfuncao
```

Exercício 2

```
1: funcao Exercicio2
2: para m de 7 ate 28 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 3

```
1: funcao Exercicio3
2: m ← 8
3: enquanto m < 20
4:     para k de 4 ate 7 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 4
8: fim{enquanto}
9: fimfuncao
```

Exercício 4

```
1: funcao Exercicio4
2: m ← 4
3: enquanto m < 18 faca
4:     se m e divisivel por 3
5:         escreva (m)
6:     senão
7:         escreva (m ×4)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 5

```
1: funcao Exercicio5
2: j ← 4
3: enquanto j < 14
4:     k ← 6
5:     enquanto k < 8
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 3
12:    fim{enquanto}
13:    j ← j + 4
14: fim{enquanto}
15: fimfuncao
```

Exercício 6

```
1: funcao Exercicio6
2: para  i de 4 a 13 passo 4
3:     para  k de 7 a 09 passo 4
4:         imprima i+k
5:     fim{para}
6: fim{para}
7: fimfuncao
```

Exercício 7

```
1: funcao Exercicio7
2: j ← 4
3: enquanto j < 13
4:     k ← 12
5:     enquanto k > 6
6:         escreva j + k
7:         k ← k - 2
8:     fim{enquanto}
9:     j ← j + 4
10: fim{enquanto}
11: fimfuncao
```

Exercício 8

```
1: funcao Exercicio8
2: m ← 8
3: enquanto m < 26
4:     escreva (m)
5:     m ← m+ 4
6: fim{enquanto}
7: fimfuncao
```

Exercício 9

```
1: funcao Exercicio9
2: para m de 5 ate 30 passo 5
3:     escreva (m)
4: fim{para}
5: fimfuncao
```

Exercício 10

```
1: funcao Exercicio10
2: m ← 5
3: enquanto m < 19
4:     para k de 3 ate 8 passo 3
5:         escreva (m+k)
6:     fim{para}
7:     m ← m+ 5
8: fim{enquanto}
9: fimfuncao
```

Exercício 11

```
1: funcao Exercicio11
2: m ← 3
3: enquanto m < 19 faca
4:     se m e divisivel por 2
5:         escreva (m)
6:     senão
7:         escreva (m ×2)
8:     fim{se}
9:     m ← m + 3
10: fim{enquanto}
11: fimfuncao
```

Exercício 12

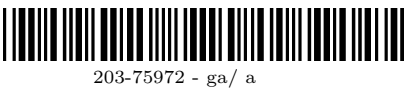
```
1: funcao Exercicio12
2: j ← 4
3: enquanto j < 12
4:     k ← 7
5:     enquanto k < 8
6:         se k mod 2 = 0
7:             escreva k-j
8:         senão
9:             escreva k+j
10:        fim{se}
11:        k ← k + 2
12:    fim{enquanto}
13:    j ← j + 3
14: fim{enquanto}
15: fimfuncao
```

.

Responda a seguir, os últimos números de cada sequência:

1	2	3	4	5	6
7	8	9	10	11	12

.



203-75972 - ga/ a