Resolvendo provas antigas da UFPR

P. Kantek

31 de julho de 2020

A melhor maneira de aprender a programar é lendo código. (Manual do Hacker, disponível na Internet).

Sumário

1 Prova 1

1.1 2014

Questão 1 (50 pontos)

Escreva um programa em C++ para ler 3 números inteiros de no máximo 2 dígitos, e verificar se a soma deles é maior que 100. Caso o valor da soma seja maior que 100 o programa deve imprimir a mensagem "limite indisponível". Caso contrário, a mensagem "limite permitido" deve ser exibida. O programa não deve aceitar valores com mais que 2 dígitos.

Exemplo de execução:

```
Indique 3 números inteiros: 2 20 12
limite permitido
```

Outro exemplo de execução do programa:

```
Indique 3 números inteiros: 2 90 12
limite indisponível
```

Outro exemplo de execução do programa:

```
Indique 3 números inteiros: 2 190 12
Valores inválidos.
```

```
#include<iostream>
using namespace std;
int main() {
  int a,b,c;
  cout << "Indique 3 numeros inteiros: "<<endl;
  cin>>a>>b>>c;
  if ((a>99) || (b>99) || (c>99)) {
    cout<<"Valores invalidos"<<endl;
  }
  else {
```



```
    if ((a+b+c)>100){
      cout<<"Limite indisponivel"<<endl;
      }
    else {
      cout<<"Limite permitido"<<endl;
    }
  }
}
```

Questão 1 (50 pontos)

Uma empresa decidiu dar uma gratificação de Natal a seus funcionários. Esta gratificação é calculada com base no número de horas extras trabalhadas e o número de horas de faltas ao trabalho. A fórmula para se calcular as horas premiadas é $HorasPremiadas = HorasExtras - \frac{2}{3} \times HorasFaltas$. O prêmio é distribuído segundo a tabela a seguir:

Horas Premiadas	Prêmio (R$)
até 10 horas (inclusive)	10,00
de 11 a 20 horas (inclusive)	20,00
de 21 a 30 horas (inclusive)	30,00
de 31 a 40 horas (inclusive)	40,00
acima de 40 horas	50,00

Faça um programa em C++ , que pergunte ao usuário o número de horas extras trabalhadas e o número de horas em que faltou ao trabalho e então mostre na tela o prêmio em reais a ser recebido.

Exemplo de execução:

```
Horas extras: 30
Horas de faltas: 3
Premio: R$ 30.00
```

```
#include<iostream>
using namespace std;
int main() {
  int ht, hf;
  float hp;
  cout << "Horas extras: "<<endl;
  cin>>ht;
  cout << "Horas de faltas: "<<endl;
  cin>>hf;
  hp = ht-((2/3.0)*hf);
  if (hp<=10){ cout<<"Premio: R$ 10.00"<<endl; }
  else {
    if (hp<=20){ cout<<"Premio: R$ 20.00"<<endl; }
    else {
      if (hp<=30){ cout<<"Premio: R$ 30.00"<<endl; }
      else {
        if (hp<=40){ cout<<"Premio: R$ 40.00"<<endl; }
        else {
          cout<<"Premio: R$ 50,00"<<endl;
        }
      }
    }
  }
}
```


Questão 1 (50 pontos)

Escreva um programa em C++ que receba três números e mostre-os em ordem decrescente. O programa somente deve aceitar 3 números diferentes.

Exemplo de execução:

```
Indique 3 números: 12 2 90
90 12 2
```

Outro exemplo de execução do programa:

```
Indique 3 números: 12 190 12
Valores inválidos.
```

Outro exemplo de execução do programa:

```
Indique 3 números: 12 12 12
Valores inválidos.
```

```
#include<iostream>
using namespace std;
int main() {
  int a,b,c;
  cout<<"Indique 3 numeros: "<<endl;
  cin>>a>>b>>c;
  if ((a==b)||(a==c)||(b==c)){
    cout<<"Valores invalidos"<<endl;
  }
  else {
    if ((a<b) && (a<c) && (b<c)) {cout<<c<<" "<<b<<" "<<a;}
    if ((a<b) && (a<c) && (c<b)) {cout<<b<<" "<<c<<" "<<a;}
    if ((b<a) && (b<c) && (a<c)) {cout<<c<<" "<<a<<" "<<b;}
    if ((b<a) && (b<c) && (a>c)) {cout<<a<<" "<<c<<" "<<b;}
    if ((c<b) && (c<a) && (a<b)) {cout<<b<<" "<<a<<" "<<c;}
    if ((c<b) && (c<a) && (a>b)) {cout<<a<<" "<<b<<" "<<c;}
  }
}
```

Questão 1 (50 pontos)

Faça um programa em C++ que leia do teclado três números reais, calcule sua média aritmética e identifique qual deles é o mais distante da média calculada. O programa deverá considerar inválida a entrada de três números iguais.

Exemplo de execução:

```
Indique 3 numeros:  3 1 7
Media: 3.666
7 está mais distante da média
```

Outro exemplo de execução do programa:

```
Indique 3 numeros:  3 9 3
Media: 5
9 está mais distante da média
```

Outro exemplo de execução do programa:

```
Indique 3 numeros:  7 7 7
Dados Invalidos.
```

```
#include<iostream>
#include<cmath>
```



```
using namespace std;
int main() {
  float a,b,c,m,d1,d2,d3;
  cout<<"Indique 3 numeros: "<<endl;
  cin>>a>>b>>c;
  if ((a==b)&&(a==c)){
    cout<<"Dados invalidos"<<endl;
  }
  else {
    m=(a+b+c)/3;
    cout<<"Media: "<<m<<endl;
    d1 = abs(a-m);
    d2 = abs(b-m);
    d3 = abs(c-m);
    if ((d1>d2) && (d1>d3)) {cout<<a<<" esta mais distante da media"<<endl;}
    if ((d2>d1) && (d2>d3)) {cout<<b<<" esta mais distante da media"<<endl;}
    if ((d3>d2) && (d3>d1)) {cout<<c<<" esta mais distante da media"<<endl;}
  }
}
```

Questão 1 (50 pontos)

Escreva um programa em linguagem C++ que leia, a partir do teclado, a data de nascimento (dia, mês e ano) de uma pessoa e determine quantos anos completos ela terá no dia 25/03/2014.

Exemplo de execução:

Data nascimento: **17 05 1988**

Idade: 25 anos

```
#include<iostream>
#include<cmath>
using namespace std;
int main() {
  int d,m,a,i;
  cout<<"Data nascimento: "<<endl;
  cin>>d>>m>>a;
  if (m<3) {i=2014-a;}
  if (m>3) {i=2014-(a+1);}
  if ((m==3) and (d<=25)) {i=2014-a;}
  if ((m==3) and (d>25)) {i=2014-(a+1);}
  cout<<"Idade: "<<i<<endl;
}
```


Questão 2 (50 pontos)

Após seus treinos para uma competição de natação, Benedito quer verificar seu desempenho em relação aos seus tempos de treino e ao tempo recorde regional. Para auxiliá-lo nesta tarefa, você deve fazer um programa em C++ que solicite a quantidade N de treinos realizados, o tempo do último recorde regional (em segundos) , e para cada treino, solicite o tempo (em segundos) cronometrado. Para este conjunto de tempos, seu programa deve calcular o maior e o menor tempo, e a média aritmética dos tempos de treino. Ao final, seu programa deve mostrar os resultados na tela e, caso o tempo médio seja menor que o tempo recorde informado, deve também mostrar uma mensagem indicando tal fato.

Exemplo de execução:

```
Quantidade de Treinos: 5
Recorde Regional: 47
Tempo 1: 45
Tempo 2: 50
Tempo 3: 44
Tempo 4: 46
Tempo 5: 48
Maior tempo: 50 segundos
Menor tempo: 44 segundos
Média: 46.6 segundos
Você está abaixo do recorde!
```

Outro exemplo de execução:

```
Quantidade de Treinos: 3
Recorde Regional: 47
Tempo 1: 47
Tempo 2: 48
Tempo 3: 49
Maior tempo: 49 segundos
Menor tempo: 47 segundos
Média: 48 segundos
```

```
#include<iostream>
using namespace std;
int main() {
  int q,rr,t,so,ma,me,i;
  float media;
  cout<<"Quantidade de treinos: "<<endl;
  cin>>q;
  cout<<"Recorde Regional: "<<endl;
  cin>>rr;
  so=0;
  ma=-99999999;
  me=99999999;
  for (i=1;i<=q;i++){
    cout<<"Tempo "<<i<<" ";
    cin>>t;
    if (t<me) {me=t;}
    if (t>ma) {ma=t;}
    so=so+t;
  }
  media=so/(q*1.0);
  cout<<"Maior tempo: "<<ma<<" segundos"<<endl;
  cout<<"Menor tempo: "<<me<<" segundos"<<endl;
  cout<<"Media: "<<media<<" segundos"<<endl;
  if (me<rr) {cout<<"Voce esta abaixo do recorde"<<endl;}
}
```


Questão 2 (50 pontos)

Maria comprou vários tipos de doce de côco, de diversos fornecedores. Após provar e guardar os doces, Maria quis saber qual o doce mais barato e a economia em comprar só o mais barato. Faça um programa em C++ que receba de inicio a quantidade total de doces comprados e o peso (em gramas) e o valor (em R$) de cada doce comprado por Maria. Calcule e imprima o preço unitário (R$ por quilograma) de cada doce e no final informe o número e preço unitário (R$/kg) do doce mais barato, o gasto total realizado e a economia que seria gerada se o peso total de doces fosse comprado apenas com o doce mais barato.

Exemplo de execução:

```
Total de doces comprados: 3
Doce 1:
 Peso (g) e preço (R$): 1000 23.34
Preço unitário = R$23.34/kg
Doce 2:
 Peso (g) e preço (R$): 500 10.51
Preço unitário = R$21.02/kg
Doce 3:
 Peso (g) e preço (R$): 1500 36.12
Preço unitário = R$24.08/kg
Produto mais barato: Doce 2, R$21.02/kg
Foram comprados 3000g de doce por R$69.97
Economia de R$6.91 se comprar 3000g do doce 2
```

```
#include<iostream>
using namespace std;
int main() {
  int q,peso,speso,i,qual;
  float preco,menorpreco,punit,spreco,economia;
  menorpreco=99999999;
  spreco=0;
  speso=0;
  cout<<"Quantidade de doces comprados: "<<endl;
  cin>>q;
  for (i=1;i<=q;i++){
    cout<<"Doce "<<i<<":"<<endl;
    cout<<"Peso (g) e preco (R$): ";
    cin>>peso>>preco;
    speso=speso+peso;
    spreco=spreco+preco;
    punit=(1000.0/peso)*preco;
    cout<<"Preco unitario :"<<punit<<endl;
    if (punit<menorpreco){
        menorpreco=punit;
        qual=i;
    }
  }
  cout<<"Produto mais barato : Doce "<<qual<<", R$ "<<menorpreco<<"/kg"<<endl;
  cout<<"Foram comprados "<<speso<<"g de doce por R$"<<spreco<<endl;
  economia=spreco-((speso*menorpreco)/1000.0);
  cout<<"Economia de "<<economia<<" se comprar "<<speso<<"g do doce "<<qual<<endl;
  }
```


Questão 2 (50 pontos)

Escreva um programa em C++ que receba dois valores representando a concentração inicial de duas bactérias e outros dois valores representando as duas respectivas taxas diárias de crescimento das culturas de bactérias. O programa deve calcular e mostrar o número de dias necessários para que a cultura da primeira bactéria ultrapasse a da segunda. Assumir que o usuário SEMPRE informa a taxa de crescimento da primeira bactéria maior que a da segunda e que ele SEMPRE informa valores de taxas entre 0 (zero) e 1 (um).

Exemplo de execução:

```
Concentracao inicial da bacteria 1: 40
Taxa de crescimento da bacteria 1 (entre 0 e 1): 0.5
Concentracao inicial da bacteria 2: 100
Taxa de crescimento da bacteria 2 (entre 0 e 1): 0.1
A 1a bacteria ultrapassa a 2a em 3 dia(s).
```

```
#include<iostream>
using namespace std;
int main() {
  float b1,b2,tx1,tx2;
  int qd;
  cout<<"Concentracao inicial da bacteria 1: ";
  cin>>b1;
  cout<<"taxa de crescimento da bateria 1 (entre 0 e 1): ";
  cin>>tx1;
  cout<<"Concentracao inicial da bacteria 2: ";
  cin>>b2;
  cout<<"taxa de crescimento da bateria 2 (entre 0 e 1): ";
  cin>>tx2;
  qd=0;
  while (b1<b2) {
    qd++;
    b1=b1*(1.0+tx1);
    b2=b2*(1.0+tx2);
  }
  cout<< "A 1a. bacteria ultrapassa a segunda em "<<qd<<" dias"<<endl;
}
```

Questão 2 (50 pontos)

Escrever um programa em C++ que leia do teclado dois números inteiros positivos com 3 algarismos, respectivamente os limites inferior e superior de um intervalo de números inteiros. Validado o intervalo (vide exemplo de execução), pede-se que o programa liste e conte os números inteiros entre os dois limites (incluindo estes) em que o dígito da dezena seja 3, 5 ou 7.

Exemplo de execução:

```
Informe limites (inferior e superior): 128 115
Limites inválidos
```

Outro exemplo de execução:

```
Informe limites (inferior e superior): 23 12000
Limites invalidos.
```

Outro exemplo de execução:

```
Informe limites (inferior e superior): 110 150
130 131 132 133 134 135 136 137 138 139 150
Nos. listados: 11
```

```
#include<iostream>
```



```
using namespace std;
int main() {
  int ni,ns,x,y,qt;
  cout<<"Informe limites (inferior e superior): ";
  cin>>ni>>ns;
  if (ni>ns) {cout<<"Limites invalidos"<<endl;}
  else {
    if ((ni>999) || (ns>999)) {cout<<"Limites invalidos "<<endl;}
    else {
        qt=0;
        while (ni<=ns) {
            x=ni;
            y=(x-(x%10))/10;
            if (((y%10)==3) || ((y%10)==5) || ((y%10)==7)){
                cout<<ni<<" ";
                qt++;
            }
            ni++;
        }
        cout<<endl<<"Nos. listados "<<qt<<endl;
        }}}
```

Questão 2 (50 pontos)

Escreva um programa em linguagem C++ que leia, a partir do teclado, um quantidade arbitrária de pares de números reais, sendo um par de Zeros a identificação do final dos pares, e determine a média dos produtos positivos e dos produtos negativos destes pares.

Exemplo de execução:

```
73.5    9.8
17.3    -25.3
-8.7    19.5
-22.9   -32.7
91.2    19.2
0    0
Media dos produtos positivos:
      1073.39
Media dos produtos negativos:
      -303.67
```

```
#include<iostream>
using namespace std;
int main() {
  float p1,p2,mpos,spos,mneg,sneg;
  int qpos,qneg;
  p1=1;
  p2=1;
  spos=0;
  sneg=0;
  qpos=0;
  qneg=0;
  while (1==1) {
    cin>>p1>>p2;
    if ((p1==0) && (p2==0)) {break;}
    if ((p1*p2)>=0) {
        spos=spos+(p1*p2);
        qpos++;
    }
    else {
        sneg=sneg+(p1*p2);
        qneg++;
```



```
    }
  }
  mpos=spos/qpos;
  mneg=sneg/qneg;
  cout<<"Media dos produtos positivos: "<<endl;
  cout<<mpos<<endl;
  cout<<"Media dos produtos negativos: "<<endl;
  cout<<mneg<<endl;
}
```

1.2 2014 s2

Questão 1 (50 pontos)

Um programa para gerenciar as contas de um banco deve possuir algum mecanismo para calcular o saldo dos clientes após as movimentações diárias. Escreva um programa em C++ que receba o valor da quantia em conta, e após o código da operação realizada e valor associado. O programa deve retornar o saldo em conta. Se o saldo for negativo, ele deve imprimir mensagem informando que o cliente entrou no cheque especial.
As operações permitidas são:

100 depósito em dinheiro
101 depósito em cheque
102 depósito de outra conta
200 saque em dinheiro
201 saque com cheque
202 saque para outra agência

Exemplo de execução:
```
Saldo: 2000
202   1000
Saldo em conta: 1000
```

Outro exemplo de execução:
```
Saldo: 1000
200   1500
Você entrou no Cheque Especial !!
```

```
#include<iostream>
using namespace std;
int main() {
  int saldo,cod,valor;
  cout<<"Saldo: ";
  cin>>saldo;
  cin>>cod>>valor;
  if (cod<103) {saldo=saldo+valor;}
  if (cod>102) {saldo=saldo-valor;}
  if (saldo >=0) {
    cout<<"Saldo em conta: "<<saldo<<endl; }
  else {
    cout<<"Voce entrou no cheque especial !"<<endl;}
}
```

Questão 2 (50 pontos)

Escreva um programa em C++ que leia dois números inteiros, x, e n, sendo n não-negativo, e que calcule o valor de x^n. Não é permitido o uso da função `float pow(float x, float y)` ou de qualquer outra função pré-existente.

Exemplo de execução:
```
Digite valores x e n: 3 4
O valor de 3^4 é: 81
```

Outro exemplo de execução:
```
Digite valores x e n: -3 3
O valor de -3^3 é: -27
```

Outro exemplo de execução:
```
Digite valores x e n: 3 -3
O expoente deve ser positivo.
```

Outro exemplo de execução:
```
Digite valores x e n: 3 0
O valor de 3^0 é: 1
```

```
#include<iostream>
using namespace std;
int main() {
  int x,n,r,i;
  r=1;
  cout<<"Digite valores x e n: ";
  cin>>x>>n;
  if (n<0) {cout<<"O expoente deve ser positivo";}
  else {
    for (i=1;i<=n;i++){
      r=r*x;
```



```
    }
    cout<<"O valor de "<<x<<"^"<<n<<" e: "<<r;
  }
}
```

Questão 1 (50 pontos)

Um atleta treina em uma pista de corrida circular de 400 metros. Ao final do treino, com base na quantidade de voltas dadas e no tempo total do treino (em horas, minutos e segundos), ele quer saber:

- O percurso total do treino (em quilômetros);
- O tempo médio por quilômetro (em horas, minutos e segundos);
- O tipo de atleta, de acordo com o tempo médio por quilômetro:
  - Atleta de competição, se o tempo médio está no intervalo entre 0 e 4 minutos (inclusive);
  - Atleta participante, se o tempo médio está no intervalo acima de 4 e abaixo de 7 minutos (7 inclusive);
  - Participante de caminhada, se o tempo médio é acima de 7 minutos.

Faça um programa em C++ que auxilie o atleta a ter essas respostas. Para o tempo, se a sua solução tiver valores abaixo de segundos, ignore a parte decimal/fracionária após os segundos.

Exemplo de execução

```
No. voltas: 13
Tempo treino(hh mm ss): 0 55 23
Percurso total (km): 5.2
Tempo médio/km (hh mm ss): 0 10 39
Tipo: Particiante de caminhada
```

Outro exemplo de execução

```
No. voltas: 13
Tempo treino(hh mm ss): 0 20 30
Percurso total (km): 5.2
Tempo médio/km (hh mm ss): 0 3 57
Tipo: Atleta de competição
```

```
#include<iostream>
using namespace std;
int main() {
  int voltas,h,m,s,h1,m1,s1,tm,tt;
  float ptotal;
  cout<<"Numero de voltas: ";
  cin>>voltas;
  cout<<"Tempo treino (hh mm ss): ";
  cin>>h>>m>>s;
  ptotal=voltas*0.4;
  tt=(h*3600)+(m*60)+s;
  tm=tt/ptotal;
  h1=tm/3600;
  m1=(tm-(h1*3600))/60;
  s1=tm-((h1*3600)+(m1*60));
  cout<<"Percurso total "<<ptotal<<endl;
  cout<<"Tempo medio/km (hh mm ss): "<<h1<<" "<<m1<<" "<<s1<<endl;
  if (m1<5) {cout<<"Tipo: atleta de competicao"<<endl;}
  else {if (m1<8) {cout<<"Tipo: atleta participante"<<endl;}
    else {cout<<"Tipo: participante de caminhada"<<endl;}
  }
}
```


Questão 2 (50 pontos)

Escreva um programa em C++ que leia do teclado conjuntos de três medidas a, b, e c (reais e positivas) correspondentes aos lados de triângulos. Para cada tripla lida, o programa deverá verificar e informar se é ou não realmente possível a montagem de um triângulo. Sendo possível, o perímetro do triângulo deverá ser calculado e mostrado no monitor de vídeo. Não sendo possível, calcular e mostrar a dimensão mínima ACIMA DA QUAL a adição desta a algum dos lados menores informados possibilitará a formação de um triângulo. O conjunto de triplas se encera com a informação da primeira medida nula ou negativa; esta tripla não faz parte do conjunto de dados a processar. Ao final do programa deverão ser mostrados o total de triângulos possíveis de montagem, seu perímetro médio, o total de triângulos não possíveis de montagem, a média dos incrementos faltantes e o total de triplas lidas. **Observação:** seu programa não precisa validar se as medidas informadas são reais e positivas. Lembre-se que para formar um triângulo, a soma de dois lados quaisquer é sempre maior que a medida do terceiro lado, isto é, dados $a, b,$ e c, eles formam um triângulo se e somente se $a+b>c, b+c>a, a+c>b$

Exemplo de execução:

```
Medidas: 3 4 5
Formam um triângulo. Perímetro: 12
Medidas: 10 2 4
Não formam um triângulo.
    Incremento mínimo > 4
Medidas: 2.5 3 4.5
Formam um triângulo. Perímetro: 10
Medidas: 6 6 5
Formam um triângulo. Perímetro: 17
Medidas: 1 1 7
Não formam um triângulo.
    Incremento mínimo > 5
Medidas: -3 1 2
Triângulos montados: 3
Perímetro médio: 13
Triângulos não montados: 2
Média dos incrementos faltantes: 4.5
Total de triplas: 5
```

```
#include<iostream>
using namespace std;
int main() {
  float a,b,c,mai,men,q,sp,si,qn,mm,mn;
  sp=0; si=0; q=0; qn=0;
  while (1==1){
    cout<<"Medidas: ";
    cin>>a>>b>>c;
    if (a<0) {break;}
    if (((a+b)>c) && ((a+c)>b) && ((b+c)>a)) {
      cout<<"Formam um triangulo. ";
      cout<<"Perimetro: "<<a+b+c<<endl;
      q=q+1;
      sp=sp+a+b+c;
    }
    else {
      if ((a>=b)&&(a>=c)){mai=a; men=b+c;}
      if ((b>=c)&&(b>=a)){mai=b; men=a+c;}
      if ((c>=a)&&(c>=b)){mai=c; men=a+b;}
      cout<<"Nao formam um triangulo"<<endl;
      cout<<"Incremento minimo > "<<mai-men<<endl;
      si=si+(mai-men);
      qn=qn+1;
    }
  }
  mm=sp/q;
  mn=si/qn;
  cout<<"Triangulos montados: "<<q<<endl;;
  cout<<"Perimetro medio: "<<mm<<endl;
  cout<<"Triangulos nao montados: "<<qn<<endl;
  cout<<"Media incrementos faltantes: "<<mn<<endl;
  cout<<"Total de triplas: "<<q+qn<<endl;
}
```


Questão 1 (50 pontos)

Faça um programa em C++ que leia 4 números inteiros H_a, M_a, H_d e M_d, onde $H_a : M_a$ representa o horário atual (H_a *horas* e M_a *minutos*) e $H_d : M_d$ representa o horário programado em um despertador (H_d *horas* e M_d *minutos*). Seu programa deve calcular e mostrar quanto tempo resta até que o alarme desperte (quantas horas e quantos minutos). Se os horários forem **idênticos**, seu programa deve indicar que o alarme *está despertando*, e, se o horário programado for **anterior** ao horário atual, seu programa deve indicar que o alarme *já despertou*. Considere que os valores H_a e H_d estão sempre entre 0 e 23 e que os valores M_a e M_d estão sempre entre 0 e 59. Além disso, considere que o horário programado no despertador é referente ao **dia atual** e nunca ao dia seguinte.

Exemplo de execução:

```
10 12
14 15
4 horas e 3 minutos para despertar
```

Outro exemplo de execução:

```
16 34
22 06
5 horas e 32 minutos para despertar
```

Outro exemplo de execução:

```
23 42
03 06
Alarme já despertou!
```

Outro exemplo de execução:

```
09 42
09 42
Alarme está despertando!
```

```
#include<iostream>
using namespace std;
int main() {
  int ha,ma,hd,md,du;
  cin>>ha>>ma;
  cin>>hd>>md;
  if ((ha==hd) &&(ma==md)) {
    cout<<"Alarme esta despertando"<<endl;
    return 0;
  }
  if ((ha>hd) || ((ha==hd) && (ma > md))){
    cout<<"Alarme ja despertou"<<endl;
    return 0;
  }
  du=((hd*60)+md)-((ha*60)+ma);
  cout<<(du/60)<<" horas e "<<du%60<<" minutos para despertar"<<endl;
}
```

Questão 2 (50 pontos)

Faça um programa em C++ que leia um conjunto de valores inteiros $n_1, n_2, \ldots, n_d$ representando as faces obtidas em d lançamentos de um dado de seis faces e imprime uma mensagem indicando se o dado é *regular* ou se está *viciado*. Considere que todo valor lido n_i está entre 1 e 6, isto é, $1 \leq n_i \leq 6$, e que o final do conjunto de valores é indicado quando o usuário digitar o valor 0 (zero), que **não** deve ser contado como um lançamento do dado. Além disso, considere que o dado está viciado se o número de ocorrências de alguma de suas faces for **maior que o dobro** do valor esperado, definido pelo número total de lançamentos multiplicado pela probabilidade de ocorrência da respectiva face. Isto é,

$$\text{Dado viciado} \iff \text{Ocorrências de alguma das faces} > 2 \times \text{Número total de lançamentos} \times \frac{1}{6}$$

Exemplo de execução:

```
1
1
2
5
4
6
0
Dado regular
```

Outro exemplo de execução:

```
6
4
3
4
1
4
4
2
0
Dado viciado
```

```
#include<iostream>
using namespace std;
int main() {
  int da,a1,a2,a3,a4,a5,a6,q;
  a1=0; a2=0; a3=0; a4=0; a5=0; a6=0; q=0;
  while (1==1){
    cin>>da;
    if (da==0) {break;}
    q++;
    if (da==1) {a1++;}
    if (da==2) {a2++;}
    if (da==3) {a3++;}
```



```
    if (da==4) {a4++;}
    if (da==5) {a5++;}
    if (da==6) {a6++;}
  }
  if ((a1>((2*q)/6.0)) || (a2>((2*q)/6.0)) || (a3>((2*q)/6.0)) ||
    (a4>((2*q)/6.0)) ||
    (a5>((2*q)/6.0)) ||
    (a6>((2*q)/6.0))) {
    cout<<"Dados viciado"<<endl;
  }
  else {
    cout<<"Dados regular"<<endl;
  }
}
```

1.3 2015

Questão 1 (50 pontos)

Escreva um programa em C++ que lê dois números inteiros A e B do teclado e imprime as mensagens *Nulo*, *Sinal negativo* ou *Sinal positivo*, conforme o resultado da multiplicação de A por B, SEM REALIZAR A OPERAÇÃO.

Exemplo de execução:
10 12
Sinal positivo

Outro exemplo de execução:
16 -34
Sinal negativo

Outro exemplo de execução:
0 -34
Nulo

```
#include<iostream>
using namespace std;
int main() {
  int a,b;
  cin>>a>>b;
  if (((a>0) && (b>0)) || ((a<0) && (b<0))){
    cout<<"Sinal positivo" <<endl;
  }
  else {
    if ((a==0) || (b==0)){
      cout<<"Nulo"<<endl;
    }
    else {
      if (((a>0) && (b<0)) || ((a<0) && (b>0))) {
        cout<<"Sinal negativo"<<endl;
      }
    }
  }
}
```


Questão 2 (50 pontos)

Faça um programa em C++ que leia um conjunto de pares de valores inteiros positivos $(p_1, n_1), (p_2, n_2), \cdots, (p_k, n_k)$ onde p_i representa uma praça de pedágio saindo de Curitiba e n_i representa a quantidade de carros que passaram nas últimas 24 horas por essa estação de pedágio p_i. Foram avaliadas 5 praças de pedágios saindo de Curitiba. Considere que todos os valores lidos são inteiros e positivos, e que o valor lido p_i está entre 1 e 5, isto é, $1 \leqslant p_i \leqslant 5$, e que o final do conjunto de valores é indicado quando o usuário digitar os valores 0 0 (zero zero). No final o programa deverá informar (a) A quantidade total de veículos que passou em cada Praça de Pedágio, (b) A quantidade total de carros que passaram pelas praças de pedágio, e (c) A quantidade média de veículos por praça de pedágio. Cada praça de pedágio p_i pode receber vários lançamentos de n_i que deverão ser somados.

Exemplo de execução:

```
Informe as praças e quantidades:
1 1250
2 3020
4 3018
5 4109
3 3298
2 765
1 435
2 3278
0 0
Praça de Pedágio 1: 1685 veículos
Praça de Pedágio 2: 7063 veículos
Praça de Pedágio 3: 3298 veículos
Praça de Pedágio 4: 3018 veículos
Praça de Pedágio 5: 4109 veículos
Quantidade total: 19173 veículos
Média por Praça de Pedágio: 3834.60 veículos
```

```
#include<iostream>
using namespace std;
int main() {
  int p1,p2,p3,p4,p5;
  int ped,car,s;
  float med;
  p1=0; p2=0; p3=0; p4=0; p5=0; s=0;
  while (1==1) {
    cin>>ped>>car;
    if ((ped==0) && (car==0)){
      break;
    }
    if (ped==1) {p1=p1+car;}
    if (ped==2) {p2=p2+car;}
    if (ped==3) {p3=p3+car;}
    if (ped==4) {p4=p4+car;}
    if (ped==5) {p5=p5+car;}
    s=s+car;
  }
  med=s/5.0;
  cout<<"Praca de pedagio 1: "<<p1<<" veiculos"<<endl;
  cout<<"Praca de pedagio 2: "<<p2<<" veiculos"<<endl;
  cout<<"Praca de pedagio 3: "<<p3<<" veiculos"<<endl;
  cout<<"Praca de pedagio 4: "<<p4<<" veiculos"<<endl;
  cout<<"Praca de pedagio 5: "<<p5<<" veiculos"<<endl;
  cout<<"Quantidade total: "<<s<<" veiculos"<<endl;
  cout<<"Media por praca de pedagio : "<<med<<" veiculos"<<endl;
}
```


Questão 1 (50 pontos)

Você foi contratado por uma nova república que necessita fornecer um programa escrito em C++ para realizar o ajuste anual de rendimentos pessoa física. O programa deve ler:

- RBA: Renda bruta anual;
- NDP: Número de dependentes;
- IPG: Imposto pago no ano.

A base de cálculo (BC) é dada por $RBA - NDP \times 1500,00$. O número máximo de dependentes dedutíveis é 4. Para o cálculo do imposto devido você deve utilizar tabela progressiva abaixo para definir imposto devido (IBC), que é calculado aplicando a porcentagem indicada (alíquota) sobre a Base de Cálculo:

BC	Alíquota
até 15.000,00	0% (isento)
de 15.000,01 até 35.000,00	7,5%
de 35.000,01 até 280.000,00	15,0%
acima de 280.000,00	60%

Com estes dados você deve calcular o imposto devido e, descontando deste o imposto que já foi pago no ano (IPG), informar ao usuário:

- Valor final do imposto a pagar
- Valor final da restituição a receber
- Mensagem indicando que usuário tem imposto quitado.

Exemplo de execução:

```
RBA: 55000.00
NDP: 3
IPG: 7230.00
Imposto a pagar: R$ 345.00
```

Outro exemplo de execução:

```
RBA: 35000.00
NDP: 2
IPG: 10230.00
Restituição a receber: R$ 7830.00
```

```
#include<iostream>
using namespace std;
int main() {
  float rba,ipg,bc,ali,ibc,dif;
  int ndp;
  cout<<"RBA: ";
  cin>>rba;
  cout<<"NDP: ";
  cin>>ndp;
  cout<<"IPG: ";
  cin>>ipg;
  if (ndp>4) {ndp=4;}
  bc=rba-(ndp*1500.00);
  if (bc<=15000.00) {ali=0.0;}
  else {
    if (bc<=35000.00) {ali=0.075;}
    else {
      if (bc<=280000.00) {ali=0.15;}
      else {ali=0.6;}
    }
  }
  ibc=bc*ali;
  dif=ipg-ibc;
  if (dif>0){cout<<"Restituicao a receber: R$ "<<dif<<endl;}
  if (dif<0){cout<<"Imposto a pagar: R$ "<<(-dif)<<endl;}
  if (dif==0){cout<<"Imposto quitado"<<endl;}
}
```

Questão 2 (50 pontos)

Escreva um programa em C++ para calcular e exibir os tempos gastos e alturas percorridas por uma lesma em suas escaladas em arranha-céus. Para isso, o usuário deverá informar a altura total a ser escalada (Alt), a quantidade de metros que a lesma consegue escalar por turno de 12 horas de subida (V_{subida}) e a quantidade de metros que ela escorrega por turno de 12 h de descanso ($V_{descida}$). Os valores de entrada para Alt e V_{subida} devem ser positivos e não-nulos, e o valor para $V_{descida}$ deve estar no intervalo $[0, V_{subida})$. Caso estas restrições não sejam satisfeitas, o programa deve terminar sem efetuar qualquer cálculo, exibindo uma mensagem indicando o fato transgressor. A escalada deverá ser descrita como nos exemplos a seguir.

Exemplo de execução:

```
Altura total a escalar (m): 33
Escalada de subida (m / 12h): 10
Escorregamento de descanso (m / 12h): 2
Desenvolvimento da escalada:
0h / 0m
12h / 10m
24h / 8m
36h / 18m
48h / 16m
60h / 26m
72h / 24m
82,8h / 33m <Objetivo atingido>
```

Obs.: 82,8h: 33m - 24m = 9m / 10m =
= 0,9 x 12h = 10,8h + 72h = 82,8h

Outro exemplo de execução:

```
Altura total a escalar (m): 5
Escalada de subida (m / 12h): 10
Escorregamento de descanso (m / 12h): 2
Desenvolvimento da escalada:
0h / 0m
6h / 5m <Objetivo atingido>
```

Obs.: 6h: 5m - 0m = 5m / 10m =
= 0,5 x 12h = 6h + 0h = 6h

```
#include<iostream>
using namespace std;
int main() {
  int ate,sub,esd,alt,hora,i;
  float fin;
  cout<<"Altura total a escalar (m): ";
  cin>>ate;
  cout<<"Escalada de subida (m/12h): ";
  cin>>sub;
  cout<<"Escarregamento de descanso (m/12h): ";
  cin>>esd;
  cout<<"Desenvolvimento da escalada:"<<endl;
  hora=0;
  alt=0;
  i=0;
  cout<<" 0h / 0m"<<endl;
  while (alt<ate){
    hora=hora+12;
    i++;
    if (i%2==1){
      alt=alt+sub;
    }
    else {alt=alt-esd;}
      if (alt<ate) {
        cout<<hora<<"h / "<<alt<<" m"<<endl;
      }
  }
  fin=12*((ate-(alt-sub))/(sub*1.0));
  cout<<(hora-12)+fin<<"h / "<<ate<<" m
  <objetivo atingido>"<<endl;
}
```


Questão 1 (50 pontos)

Dada a atual crise hídrica do país, as pessoas começaram a construir reservatórios para armazenar água em suas propriedades. Faça um programa em C++ que auxilie os utilizadores do reservatório a controlarem seu consumo. Obtenha do teclado as dimensões de um reservatório (altura, largura e comprimento, em centímetros) e o consumo médio diário dos utilizadores do reservatório (em litros/dia). Assuma que o reservatório esteja cheio, tenha formato cúbico e informe: (a) A capacidade total do reservatório, em litros; (b) A autonomia do reservatório, em dias; (c) A classificação do consumo, de acordo com a quantidade de dias de autonomia: **Consumo elevado**, se a autonomia for menor que 2 dias; **Consumo moderado**, se a autonomia estiver entre 2 e 7 dias; **Consumo reduzido**, se a autonomia maior que 7 dias. **Obs.:** Considere que cada litro equivale a 1000 cm^3 ou 1 dm^3.

Exemplo de execução:

```
Altura do reservatorio (cm): 100
Largura do reservatorio (cm): 100
Comprimento do reservatorio (cm): 100
Consumo medio diario (litros/dia): 400
Capacidade reservatorio: 1000 litros
Autonomia reservatorio: 2.5 dias
Consumo moderado
```

Outro exemplo de execução:

```
Altura do reservatorio (cm): 50
Largura do reservatorio (cm): 100
Comprimento do reservatorio (cm): 150
Consumo medio diario (litros/dia): 400
Capacidade reservatorio: 750 litros
Autonomia reservatorio: 1.875 dias
Consumo elevado
```

```
#include<iostream>
using namespace std;
int main() {
  int a,l,c,cd;
  float aut;
  cout<<"Altura do reservatorio (cm): ";
  cin>>a;
  cout<<"Largura do reservatorio (cm): ";
  cin>>l;
  cout<<"Comprimento do reservatorio (cm): ";
  cin>>c;
  cout<<"Consumo medio diario (litros/dia): ";
  cin>>cd;
  cout<<"Capacidade Reservatorio: "<<(a*l*c)/1000.0<<" litros"<<endl;
  aut=(a*l*c)/(cd*1000.0);
  cout<<"Autonomia do reservatorio: "<<aut<<" dias"<<endl;
  if (aut<2){ cout<<"Consumo elevado"<<endl;}
  else {
    if (aut<=7.0){
      cout<<"Consumo moderado"<<endl;}
    else {
      cout<<"Consumo reduzido"<<endl;
    }
  }
}
```

Questão 2 (50 pontos)

Escreva um programa em C++ que leia do teclado uma sequência de números inteiros até que seja lido um número que seja o dobro ou a metade do anterior. O programa deve imprimir como saída os seguintes valores: (a) A quantidade de números lidos; (b) A soma dos números lidos; (c) Os dois valores que forçaram a parada do programa.

Exemplo de execução:

```
4 3 15 8 16
5 46 8 16
```

Outro exemplo de execução:

```
-424 646 438 892 964 384 192
7 3092 384 192
```

```
#include<iostream>
using namespace std;
int main() {
  int atu,ant,q,s;
  cin>>atu;
  s=atu;
  q=1;
  while (1==1){
    ant=atu;
    cin>>atu;
```



```
    s=s+atu;
    q++;
    if ((atu==ant*2) || (atu==ant/2)) {break;}
  }
  cout<<q<<" "<<s<<" "<<ant<<" "<<atu<<endl;
}
```

Questão 1 (50 pontos)

Vamos chamar de *crescente* um número natural $n = d_1 d_2 \ldots d_k$ cujos dígitos d_i estão em ordem crescente, isto é, tal que $d_1 < d_2 < \cdots < d_k$. Faça um programa em C++ que leia um número inteiro e positivo n de 3 dígitos e verifique se n é crescente. Seu programa deve também verificar se n possui exatamente 3 dígitos e imprimir mensagens adequadas em cada caso.

Exemplo de execução:

```
Entre com um inteiro positivo:
3416
valor inválido.
```

Outro exemplo de execução:

```
Entre com um inteiro positivo: 152
152 não é crescente.
```

Outro exemplo de execução:

```
Entre com um inteiro positivo: 378
378 é crescente.
```

```
#include<iostream>
using namespace std;
int main() {
  int a,d1,d2,d3;
  cin>>a;
  if (a>999) { cout<<"Valor invalido";}
  else{
    d1=a/100;
    d2=(a-(d1*100))/10;
    d3=a%10;
    if ((d3>d2) && (d2>d1)) {cout<<a<<" e crescente";}
    else {cout<<a<<" nao e crescente";}
  }
}
```

Questão 2 (50 pontos)

O licenciamento anual de um veículo deve ser pago de acordo com o último dígito da sua placa, nos meses previstos na tabela abaixo:

Final da placa	Mês de renovação
1	Abril
2	Maio
3	Junho
4	Julho
5 e 6	Agosto
7	Setembro
8	Outubro
9	Novembro
0	Dezembro

Uma locadora de veículos precisa saber quantos licenciamentos terá de pagar a cada mês, de acordo com as placas na sua frota. Escreva um programa em C++ que leia uma sequência de placas de automóveis (-1 indica o término da sequência), cada placa representada apenas pelos 4 dígitos, e que determine (a) a quantidade de licenciamentos que deverá ser paga a cada mês, (b) a quantidade de carros na frota da locadora e (c) a média mensal de licenciamentos que deverão ser pagos.

Exemplo de execução:

```
Informe as placas dos veículos:
4444  2231  4451  3422  1230
3211  2376  6789  9820  3425  -1
Total de Abril: 3
Total de Maio: 1
Total de Junho: 0
Total de Julho: 1
Total de Agosto: 2
Total de Setembro: 0
Total de Outubro: 0
Total de Novembro: 1
Total de Dezembro: 2
Tamanho da frota: 10 veículos
Média mensal de licenciamentos pagos: 1,1
```

```
#include<iostream>
using namespace std;
int main() {
  int placa,qabr,qmai,qjun,qjul,qago,qset,qout,qnov,qdez,qcar;
  float med;
  qabr=0; qmai=0; qjun=0; qjul=0; qago=0;
  qset=0; qout=0; qnov=0; qdez=0; qcar=0;
  while (1==1){
```



```
    cin>>placa;
    if (placa==-1) {break;}
    qcar++;
    if (placa%10==1) {qabr++;}
    if (placa%10==2) {qmai++;}
    if (placa%10==3) {qjun++;}
    if (placa%10==4) {qjul++;}
    if ((placa%10==5) || (placa%10==6)) {qago++;}
    if (placa%10==7) {qset++;}
    if (placa%10==8) {qout++;}
    if (placa%10==9) {qnov++;}
    if (placa%10==0) {qdez++;}
  }
  cout<<"Total de Abril: "<<qabr<<endl;
  cout<<"Total de Maio: "<<qmai<<endl;
  cout<<"Total de Junho "<<qjun<<endl;
  cout<<"Total de Julho: "<<qjul<<endl;
  cout<<"Total de Agosto: "<<qago<<endl;
  cout<<"Total de Setembro: "<<qset<<endl;
  cout<<"Total de Outubro: "<<qout<<endl;
  cout<<"Total de Novembro: "<<qnov<<endl;
  cout<<"Total de Dezembro: "<<qdez<<endl;
  cout<<"Tamanho da frota: "<<qcar<<" veiculos"<<endl;
  med=qcar/9.0;
  cout<<"Media mensal de licenciamentos pagos: "<<med<<" veiculos"<<endl;
}
```

1.4 2016 s2

Questão 1 (50 pontos)

Escrever um programa em C++ que leia do teclado um número inteiro de 3 algarismos, construa e exiba outro número, de 4 algarismos, de acordo com a seguinte regra: (a) os 3 primeiros algarismos, contados da esquerda para a direita, são iguais aos do número dado; b) o quarto algarismo é um dígito de controle calculado da seguinte forma: primeiro algarismo + segundo algarismo * 3 + terceiro algarismo * 5; o dígito de contole será igual ao resto da divisão dessa soma por 7, caso este resto seja ímpar, caso contrário o dígito de controle deverá ser o resto de divisão obtido acrescido de 1.

Exemplo de execução:

```
No.:  123
No. obtido: 1231
```

Outro exemplo de execução:

```
No.:  1246
Entrada inválida
```

Outro exemplo de execução:

```
No.:  333
No. obtido: 3337
```


Questão 2 (50 pontos)

Após finalmente abrir sua tão sonhada pizzaria, Giovanni decide fazer uma avaliação em tempo real (conforme os pedidos vão chegando) de todos os custos envolvidos com o queijo utilizado. Quatro tamanhos de pizza são vendidos em sua pizzaria: P (pequeno - 6 fatias), M (médio - 8 fatias), G (grande - 10 fatias) e F (família - 12 fatias). Sabendo que cada fatia da pizza utiliza uma fatia de queijo, e que o queijo que Giovanni utiliza pesa em média 30 gramas, crie um programa que leia inicialmente o custo do quilo do queijo (R$/kg) e, logo após, leia uma sequência de tamanhos de pizza (P, M, G, F), e mostre a cada nova pizza qual o total de queijo gasto até o momento (em kg) e qual o respectivo custo cumulativo do queijo (em R$). A sequência de pizzas é finalizada pelo tamanho N.

Ao final do programa, mostre quantas fatias de queijo foram utilizadas no total e quanto tempo foi gasto no processo de fatiamento do queijo (em minutos e segundos, considerando que o tempo de fatiamento é de 5 segundos por fatia).

Exemplo de execução:

```
Custo do queijo (R$/Kg):  10.5
Tamanho da pizza:  P
Qtde. queijo: 0.18 kg, custo = R$ 1.89
Tamanho da pizza:  M
Qtde. queijo: 0.42 kg, custo = R$ 4.41
Tamanho da pizza:  G
Qtde. queijo: 0.72 kg, custo = R$ 7.56
Tamanho da pizza:  F
Qtde. queijo: 1.08 kg, custo = R$ 11.34
Tamanho da pizza:  m
Tamanho inválido !
Qtde. queijo: 1.08 kg, custo = R$ 11.34
Tamanho da pizza:  M
Qtde. queijo: 1.32 kg, custo = R$ 13.86
Tamanho da pizza:  N
------------------------
Fatias: 44
Tempo fatiamento: 3 min. 40 seg.
```

Questão 1 (50 pontos)

Seu patrão comprou um barco novo para a empresa de transportes e agora precisa saber se este barco conseguirá atender aos pedidos dentro dos prazos contratados pelos clientes. O barco possui a capacidade de transportar até 1 tonelada, com velocidade de navegação que reduz gradualmente dependendo da carga, de 50 km/h (vazio) até 30km/h (lotado).

Faça um programa que leia qual o peso da carga a ser transportada (em kg), a distância a ser navegada (em km) e o tempo limite da entrega (em h), e imprima se a carga chegará pontualmente ou se estará atrasada (ou adiantada), e o número de horas e minutos de atraso (ou de adiantamento). Lembre-se de verificar se as entradas estão dentro da faixa de valores válidos.

Exemplo de execução:

```
Peso a transportar (kg):  1000
Distância a navegar (km):  30
Prazo da entrega (h):  2
Entrega com adiantamento de 1h 0m.
```

Outro exemplo de execução:

```
Peso a transportar (kg):  500
Distância a navegar (km):  450
Prazo da entrega (h):  10
Entrega com atraso de 1h 15m.
```

Outro exemplo de execução:

```
Peso a transportar (kg):  250
Distância a navegar (km):  900
Prazo da entrega (h):  20
Entrega pontual.
```

Outro exemplo de execução:

```
Peso a transportar (kg):  1500
Peso inválido !
```

Questão 2 (50 pontos)

Você está visitando os pontos turísticos de Manhattan (Nova Iorque), onde as ruas são numeradas de 1 a 155 no sentido leste/oeste (eixo x), e 1 a 16 no sentido norte sul (eixo y). Por conveniência, os endereços das atratividades são dados pelos cruzamentos das ruas (exemplo: o Empire State Building fica no cruzamento da rua 5 com a rua 33). Crie um programa que leia um cruzamento de partida no formato $eixoX_1$ $eixoY_1$, e depois leia sucessivos cruzamentos de ruas em que você fez paradas para visitar algum ponto turístico, também no formato $eixoX_2$ $eixoY_2$. A cada cruzamento de parada exiba a distância percorrida desde a última parada, e também a distância total percorrida até o momento (ambas medidas em quadras). Em Manhattan, a distância entre dois cruzamentos de rua é dada por $|eixoX_2 - eixoX_1| + |eixoY_2 - eixoY_1|$ onde esta distância é medida em quadras. Não se esqueça de validar se o usuário digitou uma coordenada de rua válida. O programa deve terminar quando o usuário digitar -1 -1 para as coordenadas de rua.

Exemplo de execução:

```
Informe coordenadas:
- cruzamento de partida:  5 4
- proximo cruzamento:  5 5
Deslocamento: 1 quadras
Deslocamento total: 1 quadras
- proximo cruzamento:  10 8
Deslocamento: 8 quadras
Deslocamento total: 9 quadras
- proximo cruzamento:  9 7
Deslocamento: 2 quadras
Deslocamento total: 11 quadras
- proximo cruzamento:  156 3
Cruzamento invalido!
- proximo cruzamento:  155 3
Deslocamento: 150 quadras
Deslocamento total: 161 quadras
- proximo cruzamento:  -1 -1
Fim!
```

1.5 2017

Questão 1 (50 pontos)

A Secretaria de Meio Ambiente que controla o índice de poluição mantém 3 grupos de indústrias que são altamente poluentes do meio ambiente. O índice de poluição aceitável varia de 0,05 até 0,25. Se o índice sobe até 0,3 as indústrias do 1º grupo são intimadas a suspenderem suas atividades. Se o índice crescer até 0,4 as industrias do 1º e 2º grupo são intimadas a suspenderem suas atividades. E se o índice chegar até 0,5 todos os grupos devem ser notificados a paralisarem suas atividades. Índice de poluição acima de 0,5 implicará em evacuar a região.

Faça um programa em C++ que leia o índice de poluição medido e emita a notificação adequada aos diferentes grupos de empresas.

Exemplo de execução:

```
Indice de poluição:  0.4
Grupos 1 e 2 serão intimados.
```

Outro exemplo de execução:

```
Indice de poluição:  0.27
Grupo 1 será intimado.
```

```
#include<iostream>
using namespace std;
int main() {
  float ip;
  cout<<"Informe o IP ";
  cin>>ip;
  cout<<(ip>0.4);
  cout<<(0.4>0.4);
  if (ip>0.5){
    cout<<"Evacuar a regiao "<<endl;
  }
  if ((ip<=0.5) && (ip>0.4)){
    cout<<"Parar todas industrias "<<endl;
  }
  if ((ip<=0.4) && (ip>0.3)){
    cout<<"Parar grupo 1 e 2 "<<endl;
  }
  if ((ip<=0.3) && (ip>0.25)){
    cout<<"Parar grupo 1 "<<endl;
  }
}
```


Questão 2 (50 pontos)

Em um país remoto da Ásia Setentrional, a locadora de veículos Alugaliza Ltda. precisa saber quantos licenciamentos terá de pagar a cada mês, de acordo com as placas na sua frota. Neste país, o licenciamento anual de um veículo deve ser pago de acordo com o último dígito da sua placa, nos meses previstos na tabela abaixo:

Finais de placa	Mês de renovação
0 até 3	Maio
4 até 7	Junho
8 até 9	Julho

Escreva um programa em C++ que receba do usuário um conjunto de placas de automóveis, cada placa representada apenas pelos 4 dígitos (sem as 3 letras iniciais). O usuário indica o final da leitura de dados informando o valor -1 (um negativo). O programa deve determinar: (a) a quantidade de licenciamentos que deverá ser paga a cada mês, (b) a quantidade total de carros informada e (c) a média mensal de licenciamentos que deverão ser pagos.

Exemplo de execução:

```
Informe placas dos veículos:
4444  2231  4451  3422  1230
3211  2376  6789  9820  3425  -1
Total de Maio: 6
Total de Junho: 3
Total de Julho: 1
Total de carros: 10
Média mensal: 3.3333
```

```
#include<iostream>
#include<stdio.h>
using namespace std;
int main() {
  int placa,a,maio,junho,julho,q;
  float media;
  maio=0;
  junho=0;
  julho=0;
  while (1==1){
    cout<<"Informe a placa (-1 encerra) ";
    cin>>placa;
    if (placa<0) {
      break;
    }
    a = placa%10;
    if (a<4) {
      maio = maio + 1;
    }
    if ((a<8) && (a>3)) {
      junho = junho + 1;
    }
    if (a>7) {
      julho = julho + 1;
    }
  }
  q=maio+junho+julho;
  media=q/3;
  cout<<"Total de maio "<<maio<<endl;
  cout<<"Total de junho "<<junho<<endl;
  cout<<"Total de julho "<<julho<<endl;
  cout<<"Total de carros "<<q<<endl;
  cout.precision(10); // nao funcionou
  cout<<"Media mensal " <<media<<endl;
  printf("%5.3f \n",media);
}
```


Questão 1 (50 pontos)

Todo final de mês você prepara o orçamento do mês seguinte que está para começar. A partir do valor de seu salário (SAL), e os valores de despesa do mês corrente, quais sejam alimentação ($ALIM$), escolas dos filhos (ESC), cartão de crédito (PCC), e pensão da ex-esposa(o) (PEX), você deve tentar prever se você consegue pagar tudo no próximo mês.

Para isto você deve escrever um programa em C++ que peça ao usuário os valores reais para $SAL, ALIM, ESC, PCC, PEX$ e faça a previsão dos gastos para o próximo mês, mostrando quais despesas você conseguirá pagar no mês. Deve ser possível pagar cada despesa em sua totalidade na seguinte ordem: escola, alimentação, cartão de crédito e pensão. Ao final, o programa também deve mostrar o quanto sobrou do salário após descontar as despesas efetivamente pagas.

Exemplo de execução:

```
Salario: R$  5675.00
Escolas: R$  2199.45
Alimentacao: R$  834.87
Cartao de Credito: R$  1523.37
Pensão Ex-esposa: R$  1749.72
Você consegue pagar Escola
Você consegue pagar Alimentacao
Você consegue pagar Cartoes
Você nao consegue pagar Pensao
Saldo do salário: R$ 1117.31
```

Outro exemplo de execução:

```
Salario: R$  2100.00
Escolas: R$  2199.45
Alimentacao: R$  834.87
Cartao de Credito: R$  1523.37
Pensão Ex-esposa: R$  749.72
Você nao consegue pagar Escola
Você consegue pagar Alimentacao
Você nao consegue pagar Cartoes
Você consegue pagar Pensao
Saldo do salário: R$ 515.41
```

Questão 2 (50 pontos)

Escrever um programa em C++ que leia do usuário um número inteiro qualquer, determine e mostre na tela quantos dígitos o formam.

Obs.: A solução desenvolvida deverá utilizar while().

Exemplo de execução:

```
N:  339
3 dígitos
```

Outro exemplo de execução:

```
N:  0
1 dígitos
```

Outro exemplo de execução:

```
N:  -5000000
7 dígitos
```

```
#include<iostream>
#include<cmath>
using namespace std;
int main() {
  int q,d,a;
  q=0;
  cout << "Informe o numero " ;
  cin>>d;
  if (d==0){
    q=1;
  }
  while (d!=0) {
    q=q+1;
    d=d/10;
  }
  cout << q;
}
```

Questão 1 (50 pontos)

Faça um programa em C++ para ler 2 números naturais de 2 algarismos e verificar se eles são similares. Para serem similares estes números precisam ter os mesmos algarismos na sua formação.

Exemplo de execução:

```
Digite 2 números:  23 32
Números similares
```

Outro exemplo de execução:

```
Digite 2 números:  45 35
Números não similares
```

Outro exemplo de execução:

```
Digite 2 números:  67 67
Números similares
```

```
#include<iostream>
#include<stdio.h>
using namespace std;
int main() {
  int a,b,d0,d1,e0,e1;
  cout<<"Informe 2 numeros ";
  cin>>a>>b;
  d0=a/10;
```



```
  d1=a%10;
  e0=b/10;
  e1=b%10;
  if (((d0==e0)||(d0==e1))&&((d1==e0)||(d1==e1))){
    cout<<"similares";}
  else {
    cout<<"nao similares";}
}
```

Questão 2 (50 pontos)

Faça um programa em C++ para ler um par de numeros inteiros AB. Para este par A, B de números lidos, se A for maior do que B, imprimir a sequência $B, B+1, \cdots, A-1, A$. Caso contrário, imprimir a sequência $A, A+1, \cdots, B-1, B$. Ao final, imprimir a soma de todos os termos da sequência.

Exemplo de execução:

```
Indique par de inteiros:  4  6
Sequencia: 4 5 6
Soma: 15
```

Outro exemplo de execução:

```
Indique par de inteiros:  -2  1
Sequencia: -2 -1 0 1
Soma: -2
```

Outro exemplo de execução:

```
Indique par de inteiros:  5  -3
Sequencia: -3 -2 -1 0 1 2 3 4 5
Soma: 9
```

```
#include<iostream>
#include<stdio.h>
using namespace std;
int main() {
  int a,b,q;
  q=0;
  cout<<"Informe a e b";
  cin>>a>>b;
  if (a>b) {
    while (b<=a) {
      cout << b << " " ;
      q=q+b;
      b=b+1;
    }
  }
  else {
    while (a<=b) {
      cout << a << " " ;
      q=q+a;
      a=a+1;
    }
  }
  cout << "total " << q << endl;
}
```

1.6 2018

Questão 1 (50 pontos)

Uma categoria de imposto bastante comum é aquela baseado no lucro de operações de compra e venda. Tipicamente esses imposto são baseados em uma tabela progressiva. Considere um país que tributa o lucro usando a seguinte tabela:

Lucro (R$)	Imposto
De 0 até 1.000,00	ISENTO
De 1.000,01 até 3.729,00	10%
Acima de 3.729,00	25%

Dada essa tabela, escreva um programa em C++ que lê os valores de compra e venda, calcule o lucro obtido e o imposto devido, exibindo estes dois valores na tela.

Exemplo de execução:

```
Entre o valor de compra: 11420.00
Entre o valor de venda: 17492.00
Seu lucro foi: 6072.00
Imposto devido: 1518.00
```


Questão 2 (50 pontos)

Escreva um programa em C++ que lê um número inteiro positivo informado pelo usuário e calcule a soma dos seus dígitos.

Exemplo de execução:

```
Digite um número inteiro positivo: 1234
Soma dos dígitos: 10
```

Outro exemplo de execução:

```
Digite um número inteiro positivo: 227634
Soma dos dígitos: 24
```

Questão 1 (50 pontos)

A empresa PIRRALHOS ENDIABRADOS vende brinquedos inteligentes e tem muitos brinquedos pequenos sempre guardados em caixas no formato de cubos (paralelepípedos retos com arestas iguais). A empresa comprou muitas esferas de raios 10, 20 e 30 cm. Pretende encerrar cada brinquedo dentro de uma esfera (como se fosse um kinder ovo).

Você deve escrever um programa C++ que peça e receba a aresta do cubo de um brinquedo e deve descobrir qual a menor esfera que pode escondê-lo, imprimindo o raio da esfera. Se o brinquedo não couber nem na maior esfera, o programa deve imprimir "NÃO CABE".

DICAS: Para um cubo de aresta a e uma esfera de raio r: $V_{cubo} = a^3$, $V_{esfera} = \frac{4}{3}\pi \times r^3$, Diagonal do cubo: $a \times \sqrt{3}$, Diâmetro da esfera: $2 \times r$.

Exemplo de execução:

```
Informe a aresta do brinquedo 10
Esfera de 10cm de raio
```

Outro exemplo de execução:

```
Informe a aresta do brinquedo 40
NAO CABE
```

Outro exemplo de execução:

```
Informe a aresta do brinquedo 30
Esfera de 30cm de raio
```

Questão 2 (50 pontos)

Você trabalha em uma empresa de automação e seu chefe pediu para criar um programa para controlar as comportas que a empresa vende para tanques de piscicultura. Como os tanques recebem água de rio, na entrada do tanque há um sensor que diz quanta água entrou (em m^3). Já na saída do tanque fica a comporta, que é feita para liberar exatamente 20% da capacidade do tanque a cada vez que é aberta. A comporta deve ser aberta sempre que o tanque passar de 80% de sua capacidade. Crie um programa em C++ que leia (em m^3) a capacidade máxima do tanque e o volume atual. Depois, a cada iteração o programa deve ler quanta água entrou pelo rio (afluência em m^3) e decidir se abre ou não a comporta. O programa acaba quando a água ultrapassar a capacidade máxima do tanque mesmo após a liberação da água.

Exemplo de execução:

```
Capacidade do tanque (m^3): 100
Volume atual (m^3): 60
Afluência (m^3): 20
>  Volume atual: 80 m^3
Afluência (m^3): 20
>  Volume atual: 100 m^3
>  Liberando 20 m^3...
Afluência (m^3): 40
>  Volume atual: 120 m^3
>  Liberando 20 m^3...
Afluência (m^3): 21
>  Volume atual: 121 m^3
>  Liberando 20 m^3...
>  Capacidade do tanque excedida (101 m^3)
```

Outro exemplo de execução:

```
Capacidade do tanque (m^3): 50
Volume atual (m^3): 40
Afluência (m^3): 21
>  Volume atual: 61 m^3
>  Liberando 10 m^3...
>  Capacidade do tanque excedida (51 m^3)!
```

Outro exemplo de execução:

```
Capacidade do tanque (m^3): 50
Volume atual (m^3): 51
>  Capacidade do tanque excedida (51 m^3)!
```

Questão 1 (50 pontos)

Faça um programa em C++ que receba um caractere referente a um dos seguintes cargos: 'F' ou 'f' para funcionário e 'G' ou 'g' para gerente e receba o valor do salário. Se o usuário for funcionário calcular um desconto de 10% do valor total do salário, se escolher a opção gerente calcular um desconto de 25% do total do salário. Para um gerente, a quantidade de meses no cargo também deverá ser informada. Se o tempo no cargo for maior que 12 meses o desconto sobre o salário sobe para 30%. Após calcular o desconto no salário, o programa deve exibir o salário final, com o desconto aplicado ao salário inicial.

Exemplo de execução:

```
Digite o valor do salario: 6000.00
Qual seu cargo? f
Valor total a ser pago: 5400.00
```

Outro exemplo de execução:

```
Digite o valor do salario: 10000.00
Qual seu cargo? g
Quantos meses esta no cargo? 5
Valor total a ser pago: 7500.00
```

Questão 2 (50 pontos)

Faça um programa em C++ que peça ao usuário um par de coordenadas cartesianas (x, y). Após esta entrada o programa deve obter do usuário uma sequência de pares de coordenadas, mostrando na tela a distância entre cada par e o par inicial de coordenadas.

O programa deve terminar quando o usuário informar a coordenada $(0, 0)$.

DICA: A distância entre dois pontos (x_1, y_1) e (x_2, y_2) é dada por $D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.

Exemplo de execução:

```
Coordenadas iniciais:  0 0
Coordenadas: 2 3
Distância: 3.60555
Coordenadas: 5 4
Distância: 6.40312
Coordenadas: 11 20
Distância: 22.8254
Coordenadas: 0 4
Distância: 4
Coordenadas: 0 0
```

Questão 1 (50 pontos)

Um batalhão está tentando bombardear a tenda do general inimigo em um forte com um canhão. A tenda do general é circular e tem raio de 3 metros. A tenda está no meio do terreno do forte que também é circular e tem raio de 30 metros. O forte por sua vez está exatamente no centro de uma vila que tem uma área circular com raio de 100 metros. Suponha que o canhão esteja alinhado na direção da tenda, que a velocidade do inicial do projétil é de 50 m/s e $g = 10m/s^2$.

Construa um programa em C++ que leia a distância do canhão para o muro da cidade e o valor para o ângulo de disparo em radianos e diga se ele atingiu a tenda, o forte, a cidade ou nada.

$x_{max} = \frac{v_0^2 \cdot sen2\theta}{g}$

Exemplo de execução:

```
ângulo: 0.7854 (45 graus)
distância: 150 metros
acertou na tenda
```

Outro exemplo de execução:

```
ângulo: 0.7854 (45 graus)
distância: 100 metros
acertou na cidade
```

Outro exemplo de execução:

```
ângulo: 0.5235 (30 graus)
distância: 120 metros
acertou no forte
```

Questão 2 (50 pontos)

Escreva um programa em C++ para ler uma quantidade indeterminada de pares de números, via teclado, até que o par "0 0" seja digitado. O programa deve encontrar qual foi o par com a maior diferença entre seus números.

Exemplo de execução:

```
Digite pares de números:
20 50
13 26
80 12
0 0
Maior diferença: 68 - 3o. par (80 12)
```

2 Prova 2

2.1 2014 - s1

Questão 1 (50 pontos)

Escreva um programa em C++ que obtenha do usuário um par de valores reais e um caracter indicando uma das 4 operações aritméticas: soma (+), subtração (−), multiplicação (x) e divisão ($/$). Ao final, o programa deve mostrar o resultado da operação indicada pelo usuário. Para fazer o cálculo solicitado, deve ser criada e usada a função `calc()`, que recebe 2 valores reais e um caracter como parâmetros e retorna o resultado do cálculo correspondente. Esta função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Operação: 4 10 x
Resultado: 40
```

```
#include<iostream>
#include<cstdlib>
#include<cmath>
using namespace std;
float calc(float a, float b, char ope){
  if (ope=='+') {return a+b;}
  if (ope=='-') {return a-b;}
  if (ope=='x') {return a*b;}
  if (ope=='/') {return a/b;}
}

int q1(){
  float a,b;
  char o;
  cout<<"Operacao: ";
  cin>>a>>b>>o;
  cout<<"Resultado: "<<calc(a,b,o)<<endl;
}
```

Questão 1 (50 pontos)

Escreva um programa que leia as coordenadas cartesianas (x, y) de 2 pontos, A e B no espaço e imprima na tela a menor distância entre os 2 pontos e também a distância do menor caminho de A até B passando por um ponto intermediário, C com coordenadas $(0, 0)$. Deve ser definida e usada a função de nome `dist()` que retorna a distância entre 2 pontos, cujas coordenadas são passadas como parâmetros. Esta função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Obs.: Considere que a distância entre 2 pontos de coordenadas (x_a, y_a) e (x_b, y_b) é dada por $dist_{AB} = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$. A distância do caminho de A até B passando por $C = (0, 0)$ é dado por $caminho_{AB} = dist_{AC} + dist_{CB}$.

Exemplo de execução:

```
Digite coordenada de A: 10  50
Digite coordenada de B: 70  35
Menor distância entre A e B: 61.8466
Caminho de A a B, passando por (0,0): 129.253
```

```
float distab(float xa, float ya, float xb, float yb){
  float dx, dy;
  dx=xa-xb;
  dy=ya-yb;
  return sqrt((dx*dx)+(dy*dy));
}

int q2(){
  float xa,xb,ya,yb,dab,dabc;
  cout<<"Digite coordenadas de A: ";
  cin>>xa>>ya;
  cout<<"Digite coordenadas de B: ";
  cin>>xb>>yb;
  dab=distab(xa,ya,xb,yb);
  dabc=distab(xa,ya,0.0,0.0);
  dabc=dabc+distab(0.0,0.0,xb,yb);
  cout<<"Menor distancia entre A e B: "<<dab<<endl;
```



```
  cout<<"caminho de A a B, passando por (0,0): "<<dabc<<endl;
}
```

Questão 1 (50 pontos)

Escreva um programa em C++ para auxiliar um viajante a escolher a melhor opção de abastecimento de seu veículo bicombustível (flex), se álcool ou gasolina. A partir de médias de consumo na estrada analisadas pelo viajante, ele concluiu que seu carro abastecido com álcool tem um rendimento (em quilômetros rodados) de 78% a 82% do obtido com gasolina. Assim, se o preço que ele pagar pelo litro de álcool em relação ao litro de gasolina for inferior a esta faixa percentual, a melhor opção é abastecer com álcool. Acima desta faixa, a gasolina é mais econômica. Dentro da faixa (de 78% a 82%, inclusive) a opção de escolha é indiferente. O programa deverá receber do usuário os preços por litro do álcool e da gasolina e indicar a melhor opção de abastecimento. Para a escolha da melhor opção, deverá ser criada e usada a função `escolha_comb()`, que recebe como entradas os preços por litro do álcool e da gasolina e retorna um valor inteiro que indica a melhor opção: 1 (álcool); 2 (gasolina) ou 3 (indiferente). A função NÃO DEVE ler dados do teclado nem mostrar dados no monitor de vídeo.

Exemplo de execução:

```
Preço do litro de álcool:  R$ 2.10
Preço do litro de gasolina:  R$ 3.00
Sugestão de abastecimento: Álcool
```

Outro exemplo de execução do programa:

```
Preço do litro de álcool:  R$ 2.35
Preço do litro de gasolina:  R$ 2.70
Sugestão de abastecimento: Gasolina
```

Outro exemplo de execução do programa:

```
Preço do litro de álcool:  R$ 2.40
Preço do litro de gasolina:  R$ 3.00
Sugestão de abastecimento: Indiferente
```

```
int escolhe_comb(float alc, float gas){
  float xx;
  xx= alc/gas;
  if (xx<0.78) {return 1;}
  if (xx>0.82) {return 2;}
  return 3;
}

int q3(){
  int r;
  float pa,pg;
  cout<<"Preco do litro de alcool: ";
  cin>>pa;
  cout<<"Preco do litro de gasolina: ";
  cin>>pg;
  r = escolhe_comb(pa,pg);
  if (r==1){cout<<"Sugestao de abastecimento: alcool"<<endl;}
  if (r==2){cout<<"Sugestao de abastecimento: gasolina"<<endl;}
  if (r==3){cout<<"Sugestao de abastacimento: indiferente"<<endl;}
}
```

Questão 1 (50 pontos)

Escreva um programa em C++ que obtenha do usuário 4 valores inteiros e imprima na tela mensagem indicando se o primeiro valor informado é divisor da soma dos outros 3 valores informados. Para testar esta relação entre os valores deverá ser criada e usada a função `ehDivisor()`, que recebe 4 valores inteiros como parâmetros, e retorna o valor 1 (um) se a relação se verifica, ou 0 (zero) caso contrário. A função NÃO DEVE ler dados do teclado nem mostrar dados no monitor de vídeo.

Exemplo de execução:

```
Indique 4 números inteiros: 3 5 8 2
3 é divisor.
```

Outro exemplo de execução do programa:

```
Indique 3 números inteiros: 2 3 4 8
2 não é divisor.
```

```
int ehdivisor(int a, int b, int c, int d){
```



```
  if ((b+c+d)%a==0){return 1;}
  else{return 0;}
}

int q4(){
  int a,b,c,d,r;
  cout<<"Indique 4 numeros inteiros: ";
  cin>>a>>b>>c>>d;
  r=ehdivisor(a,b,c,d);
  if (r==1){cout<<a<<" e divisor"<<endl;}
  else {cout<<a<<" nao e divisor"<<endl;}
```

Questão 2 (50 pontos)

Escreva um programa em linguagem C++ que recebe via teclado um valor N com a quantidade de Produtos a serem informados e um conjunto de dados de Produto, contendo o Tipo do Produto (inteiro), seu Estoque atual (real), e seu Preço (real). Para cada conjunto digitado o programa deverá informar, se necessário, a quantidade necessária para repor o estoque mínimo. No final do processamento o programa deverá informar a quantidade total de itens listados e a quantidade de itens com estoque abaixo do mínimo. Deverá listar o valor total do estoque (somatório dos Preço x Quantidade em Estoque, de todos os produtos) e o valor total pendente de reposição (somatório dos Preço x Quantidade de Reposição, de todos os produtos com estoque abaixo do mínimo). Para o verificar se um produto existe no estoque, deve ser defina e usada a função `calc_reposicao()`, que recebe como parâmetros o Tipo (inteiro) e Quantidade (real), retornando um valor 1 (um) se a quantidade está abaixo do estoque mínimo, ou 0 (zero) caso contrário. A função deve também devolver um valor real com a quantidade necessária para atingir o estoque mínimo do Produto. Os níveis de estoque mínimo são dados pela tabela abaixo:

Tipo	Estoque Mínimo
100	40
200	20
300	10
400	5

Exemplo de execução:

```
Quantidade de produtos: 5
Em cada produto, indique tipo,estoque,preço
Produto 1: 200 18 5.00
Repor o estoque em 2 unidades !!
Produto 2: 300 10 6.00
Produto 3: 100 35 4.00
Repor o estoque em 5 unidades !!
Produto 4: 100 40 2.00
Produto 5 400 10 3.00
5 prod. no estoque, valor total = 400 reais.
2 prod. abaixo do minimo, com
       valor pendente = 30 reais.
```

```
int calc_reposicao(int tipo, int qtd, int & qtfalta){
  if (tipo==100) {qtfalta = 40 - qtd;}
  if (tipo==200) {qtfalta = 20 - qtd;}
  if (tipo==300) {qtfalta = 10 - qtd;}
  if (tipo==400) {qtfalta = 5 - qtd;}
  if (qtfalta > 0) {
    return 1;}
  else {
    return 0;
  }
}

int q5(){
  int i,t,e,qp,falta,pf=0,som=0,vpen=0,r;
  float p;
  cout<<"Quantidade de produtos: ";
  cin>>qp;
  cout<<"Para cada produto, indique tipo,estoque,preco ";
  for (i=1;i<=qp;i++){
    cin>>t>>e>>p;
    falta=0;
    r = calc_reposicao(t,e,falta);
    som = som+ (e*p);
```



```
    if (r==1){
      cout<<"Repor o estoque em "<<falta<<" unidades"<<endl;
      pf++;
      vpen=vpen+(falta*p);
    }
  }
  cout<<qp<<" produtos no estoque, valor total "<<som<<" reais"<<endl;
  cout<<pf<<" produtos abaixo do minimo, com "<<endl;
  cout<<"valor pendente = "<<vpen<<" reais."<<endl;
}
```

Questão 2 (50 pontos)

Teobaldo estuda o clima e está realizando um experimento de análise de temperaturas. Após coletar um conjunto de temperaturas em Grau Celsius (ºC), Teobaldo precisa converter cada um dos valores para Grau Fahrenheit (°F) e comparar se cada valor está acima da média correspondente. Para ajudar Teobaldo, sua tarefa é escrever um programa em C++ que receba do usuário um conjunto de N pares de valores, onde cada par consiste de uma temperatura em Grau Celsius e de um valor médio em Grau Fahrenheit. Para cada valor em Grau Celsius, seu programa deve mostrar na tela o valor convertido para Grau Fahrenheit e se está acima da média. Ao final, seu programa deve mostrar na tela o total de valores acima da média. Para auxiliar nesta tarefa, você deve definir e utilizar a função `converte_temp()`, que recebe dois parâmetros de entrada - a temperatura em Celsius e a média em Fahrenheit, e devolve dois outros valores - a temperatura Celsius convertida para Grau Fahrenheit e um número inteiro (1 - se a temperatura convertida está acima da média, 0 - caso contrário). Lembre-se: $°F = °C \times 1,8 + 32$.

Exemplo de execução:

```
Quantidade de Temperaturas:  3
Tempertura (C) e média (F): 28 80
28 C = 82.4 F --- Está acima da média
Tempertura (C) e média (F): 30 95
30 C = 86 F --- Está abaixo da média
Tempertura (C) e média (F): 25 70
25 C = 77 F --- Está acima da média
Existem 2 temperaturas acima da média
```

```
int converte_temp(float cels, float medf, float & fare){
  fare = (cels*1.8)+32;
  if (fare > medf){
    return -1;
  }
  else{
    return 0;
  }
}

int q6(){
  int i,qtam=0,qt;
  float t,m,r,vc;
  cout<<"Quantidade de temperaturas: ";
  cin>>qt;
  for (i=1;i<=qt;i++){
    cout<<"Temperatura (C) e media (F): ";
    cin>>t>>m;
    vc=0;
    r=converte_temp(t,m,vc);
    if (r==-1){
      cout<<t<<" C = "<<vc<<" F -- esta acima da media"<<endl;
      qtam++;
    }
    else {
      cout<<t<<" C = "<<vc<<" F -- esta abaixo da media"<<endl;
    }
  }
  cout<<"Existem "<<qtam<<" temperaturas acima da media"<<endl;
}
```


Questão 2 (50 pontos)

Depois de passar muito tempo programando e esquecendo-se de se exercitar, Teobaldo engordou mais do que deveria e decidiu começar a fazer atividades físicas para queimar calorias. Ele deseja construir um programa em C++ que monitore seus treinos informando quantos quilos ele perdeu por dia até que ele alcance um determinado peso desejado.

Este programa deve receber de Teobaldo seu peso inicial e sua meta final de peso, e em seguida receba dele diversos valores de calorias gastas em cada treino, mostrando na tela o peso após o treino e se ele teve ou não um bom rendimento. O programa deve solicitar novos valores de calorias até que o peso atual de Teobaldo seja menor que o peso desejado por ele, considerando que cada valor informado representa o resultado de um novo treino e, consequente, um novo gasto de calorias.

Para calcular o rendimento e o peso após o treino, deve ser definida e usada a função de nome `emagrece()` que recebe: (i) o peso atual de Teobaldo (em quilos) e (ii) a quantidade de calorias que ele gastou no dia e retorna: (a) o novo peso de Teobaldo e (b) 1 ou 0, caso Teobaldo tenha feito um treino bom ou ruim, respectivamente. Considere que a cada 7000 calorias gastas, Teobaldo emagrece 1 quilo; e, um treino bom é entendido como um treino onde o novo peso de Teobaldo é menor que 99% de seu peso anterior.

Exemplo de execução:

```
Informe seu peso: 85
Informe sua meta: 82
Informe calorias perdidas: 7000
Seu novo peso é: 84
Você teve um bom rendimento!
Informe calorias perdidas: 5000
Seu novo peso é: 83.2857
Você precisa treinar mais...
Informe calorias perdidas: 10000
Seu novo peso é: 81.8571
Você teve um bom rendimento!
Você atingiu sua meta!
```

```
int emagrece(float pa, float qc, float & np){
  np=pa - (qc/7000.0);
  if (np<(0.99*pa)) {return 1;}
  else {return 0;}
}

int q7(){
  float p,m,c,np;
  int r;
  cout<<"Informe seu peso: ";
  cin>>p;
  cout<<"Informe sua meta: ";
  cin>>m;
  while (p>m){
    cout<<"Informe calorias perdidas: ";
    cin>>c;
    r=emagrece(p,c,np);
    cout<<"Seu novo peso e: "<<np<<endl;
    if (r==1){cout<<"Voce teve um bom rendimento"<<endl;}
    else {cout<<"Voce precisa treinar mais"<<endl;}
    p=np;
  }
  cout<<"Voce atingiu sua meta!"<<endl;
}
```

Questão 2 (50 pontos)

Escreva um programa em C++ que que recebe via teclado um valor N e mostre a quantidade de mudanças de sinal no valor y e o primeiro valor de x onde houve mudança de sinal de y para um conjunto de N pares (x, y)

de valores reais fornecidos pelo usuário. Para determinar estes valores deve ser definida e usada a função de nome `raizes()`, que recebe como um parâmetro o número N, solicite do usuário o fornecimento de N de pares (x, y) de valores reais e devolva quantidade de mudanças de sinal no valor y e o primeiro valor de x onde houve mudança de sinal de y.
A função `raizes()` NÃO DEVE mostrar os resultados calculados na tela, mas note que esta função solicita do usuário APENAS os valores dos N pares (x, y) de valores reais.

Exemplo de execução:

```
N: 8
2 -15
3 -8
4 -2
5 3
6 7
7 1
8 -4
9 -7
Mudanças de sinal: 2
1o. x com mudança de sinal de y: 5
```

```
int raizes(int n, int & mu, int & px){
  int i,x,y,sa;
  for (i=1;i<=n;i++){
    cin>>x>>y;
    if (i==1){sa=y;}
    if ((y*sa)<0) {mu++;}
    if((mu==1)&&(px==0)){px=x;}
    sa=y;
  }
}

int q8(){
  int x,y,n,mudancas,primeiro;
  cout<<"N: ";
  cin>>n;
  mudancas=0; primeiro=0;
  raizes(n,mudancas,primeiro);
  cout<<"Mudancas de sinal: "<<mudancas<<endl;
  cout<<"Primeiro x com mudanca de sinal de y: "<<primeiro<<endl;
}
```

2.2 2014 - s2

Questão 1 (50 pontos)
Escreva um programa em C++ que solicita o total gasto pelo cliente de uma loja, e calcula o valor a vista com $x\%$ de desconto. O desconto é definido pelo usuário (dono da loja). O programa usa uma função `desc_a_vista()`, que recebe do programa principal o valor gasto e o desconto dado pelo usuário e retorna o valor a ser pago com desconto. O programa então deve retornar o valor a ser pago.
OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Total Gasto: 100
Desconto: 10
Total a pagar c/ desconto: 90
```

```
#include<iostream>
#include<cstdlib>
#include<cmath>
using namespace std;
float desc_a_vista(float valor, float desc){
  float diva;
  diva = 1 - (desc/100.0);
  return valor * diva;
```



```
}

int q1(){
  float tg,tp,de;
  cout<<"Total gasto: ";
  cin>>tg;
  cout<<"Desconto: ";
  cin>>de;
  tp=desc_a_vista(tg,de);
  cout<<"Total a pagar c/ desconto: "<<tp;
}

int segura_ou_vende(float compra, float atual, float & perc){
  perc=((atual/compra)-1)*100.0;
  if ((perc==5)&&(perc>0)) {return 1;}
  if (perc>=10){return 2;}
 if (perc<=-1.5){return -1;}
  return 0;
}

int q2(){
  int z,i,no;
  float vc,va,qt;
  cout<<"Numero de operacoes: ";
  cin>>no;
  cout<<"Valores de compra e atual, dois a dois: "<<endl;
  for (i=1;i<=no;i++){
    cin>>vc>>va;
    z = segura_ou_vende(vc,va,qt);
    if (z==-1){
      cout<<"Operacao "<<i<<": "<<qt<<"%, Va se benzer"<<endl;
    }
    if (z==0){
      cout<<"Operacao "<<i<<": "<<qt<<"%, aguarde"<<endl;
    }
    if (z==1){
      cout<<"Operacao "<<i<<": "<<qt<<"%, Venda a metade"<<endl;
    }
    if (z==2){
      cout<<"Operacao "<<i<<": "<<qt<<"%, Venda tudo e tire folga"<<endl;
    }
  }
}
```

Questão 1 (50 pontos)

Escreva um programa em C++ que simule um jogo de "par ou ímpar" entre o usuário e o computador. O jogo deverá solicitar a aposta do usuário – 0 para apostar em par, 1 para apostar em ímpar – e um valor inteiro qualquer que ele jogará. A jogada do computador também será um inteiro qualquer, gerado pela função `int rand()`, que já existe no sistema, e que retorna um número aleatório qualquer. Ganha o jogo aquele que adivinhar qual o tipo da soma dos valores apresentados pelo computador e pelo usuário, se par ou impar, conforme a aposta de cada um.

O seu programa deverá chamar a função `par_ou_impar()`, de sua autoria, que receba um inteiro indicando a aposta do usuário (par ou ímpar), um inteiro indicando o valor que o usuário jogou, e que retorne 1 caso o usuário tenha vencido ou 0 caso o computador tenha vencido.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Entre com a sua aposta (0:Par, 1:Ímpar) e o valor da sua jogada: 0 4
O computador ganhou
```

Outro exemplo de execução:

```
Entre com a sua aposta (0:Par, 1:Ímpar) e o valor da sua jogada: 1 33
Parabéns, você venceu!
```

```
int par_ou_impar(int pi, int val){
  int x,aa;
```



```
  x=rand();
  aa=(val+x)%2;
  if (pi==aa){return 1;}
  else {return 0;}
}

int q3() {
  int apo,val,r;
  cout<<"Entre aposta e valor: ";
  cin>>apo>>val;
  r = par_ou_impar(apo,val);
  if (r==1) {cout<<"Parabens, vc ganhou"<<endl;}
  else {cout<<"O computador ganhou"<<endl;
}

int verifica_preco(float ant, float atu, float & per){
  per = ((atu/ant)-1)*100.0;
  if (per<-10.0) {return 1;}
  if (per>10.0){return 2;}
  return 0;
}

int q4(){
  int qp,i,z;
  float vp,va,pe;
  cout<<"Quantos produtos: ";
  cin>>qp;
  for(i=1;i<=qp;i++){
    cout<<"Preco do produto (passado e atual)";
    cin>>vp>>va;
    z = verifica_preco(vp,va,pe);
    if (z==1){cout<<"Reducao de "<<pe*-1<<"% no preco do produto"<<endl;}
    if (z==2){cout<<"Aumento de "<<pe<<"% no preco do produto"<<endl;}
    if (z==0){cout<<"Nao houve mudanca significativa"<<endl;}
  }
}
```

Questão 1 (50 pontos)

Escreva um programa em C++ para calcular e mostrar a distância de um ponto (x, y) do plano cartesiano à origem dos eixos $(0, 0)$. A distância procurada é obtida pela raiz quadrada da soma dos quadrados dos valores das coordenadas x e y de cada ponto. Para resolver o problema proposto, pede-se o desenvolvimento e uso de uma função de nome `distxy()`, que recebe como entradas as coordenadas (x, y) de um ponto, calcula e devolve ao programa que a chamar a distância procurada.
OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:
```
(x,y): 3 4
Distância calculada: 5
```

Outra execução:
```
(x,y): 0  -8
Distância calculada: 8
```

Outra execução:
```
(x,y): -1  -1
Distância calculada: 1.414
```

```
float distxy(float x, float y){
  return sqrt((x*x)+(y*y));
}

int q5(){
  float xi, yi;
  cout<<"(x,y): ";
  cin>>xi>>yi;
  cout<<"Distancia calculada: "<<distxy(xi,yi)<<endl;
}
```


Questão 2 (50 pontos)

Teobaldo é um daqueles eleitores que insistem em acreditar que a eleição pode ser anulada se mais de 50% dos votos forem nulos. Como este ano nenhum dos candidatos lhe agrada, ele fará seu "voto de protesto". Contudo, gostaria de saber com antecedência se seu "protesto" não será em vão, pedindo sua ajuda para montar um programa que será utilizado para uma pesquisa eleitoral online. Faça um programa em C++ que leia um número inteiro n indicando o total de eleitores e, em seguida, leia n números inteiros representando os votos dos eleitores, codificados da seguinte maneira: 42 para o candidato A, 18 para o candidato B e 35 para o candidato C. Qualquer voto diferente desses valores deve ser considerado nulo. Faça uma função chamada $voto()$ que recebe 4 parâmetros, sendo os 3 primeiros as quantidades de votos dos candidatos A, B e C, respectivamente, e o último parâmetro o número a ser votado por um eleitor. Essa função deve atualizar a contagem dos votos, se o número votado for válido, e retornar 0, ou simplesmente retornar 1, caso o voto seja nulo. Ao final, seu programa deve indicar se a eleição será anulada segundo Teobaldo ou qual candidato está na frente (se dois ou mais estiverem empatados em primeiro lugar, mostre qualquer um deles). Você deve utilizar a função $voto()$ no seu programa principal.
OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
7
42
01
42
18
29
99
35
O candidato A está na frente.
```

Outro exemplo execução:

```
5
99
01
42
67
35
A eleição será anulada segundo Teobaldo.
```

```
int voto(int & a, int & b, int & c, int ve){
  if (ve==42) {a++; return 0;}
  if (ve==18) {b++; return 0;}
  if (ve==35) {c++; return 0;}
  return 1;
}

int q6(){
  int i,te,v,r,vnulo=0;
  cin>>te;
  int ta=0, tb=0, tc=0;
  for (i=1;i<=te;i++){
    cin>>v;
    r=voto(ta,tb,tc,v);
    if (r==1){vnulo++;}
  }
  cout<<ta<<" "<<tb<<" "<<tc<<" "<<endl;
  if (vnulo>(ta+tb+tc)) {cout<<"A eleicao sera anulada segundo Teobaldo"<<endl;}
  else{
    if ((ta>tb) && (ta>tc)) {cout<<"O candidato A esta na frente"<<endl;}
    if ((tb>ta) && (tb>tc)) {cout<<"O candidato B esta na frente"<<endl;}
    if ((tc>ta) && (tc>tb)) {cout<<"O candidato C esta na frente"<<endl;}
  }
}
```


2.3 2015

Questão 1 (50 pontos)

Sabe-se que 4 segmentos podem formar um quadrilátero apenas se a medida do maior segmento for menor que soma das medidas dos outros 3. Escreva um programa em C++ que recebe do usuário as medidas de 4 segmentos, verifica sua validade (valores reais positivos e não nulos), avalia a possibilidade de formar algum quadrilátero com as medidas informadas e, caso possível, calcula e mostra o perímetro do polígono obtido. Para a verificação da possibilidade de formar o quadrilátero, deve ser definida e usada a função `analisa4()`, que recebe como entrada as medidas de 4 segmentos e devolve ao programa que a chamou: -1, caso alguma das medidas informadas seja inválida; 0, caso as medidas sejam válidas mas não permitam a formação de um quadrilátero; OU o perímetro do quadrilátero, caso as medidas sejam válidas e possibilitem a formação do mesmo.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Medidas (4 segmentos): 10  2  -5  6
Medidas inválidas
```

Outro exemplo de execução:

```
Medidas (4 segmentos): 10  2  0.5  6
Medidas não formam um quadrilátero
```

Outro exemplo de execução:

```
Medidas (4 segmentos): 10  2  5  6
Perímetro do quadrilátero formado: 23
```

```
#include<iostream>
using namespace std;
int analisa4(float a, float b, float c, float d){
  if((a<=0) ||(b<=0) ||(c<=0)||(d<=0)){return -1;}
  if((a>(b+c+d))||(b>(a+c+d))||(c>(a+b+d))||(d>(a+b+c))){return 0;}
  return a+b+c+d;
}

int q1(){
  float a,b,c,d,p,z;
  cout<<"Medidas (4 segmentos): ";
  cin>>a>>b>>c>>d;
  z=analisa4(a,b,c,d);
  if(z==-1){cout<<"Medidas invalidas"<<endl;}
  if(z==0){cout<<"Medidas nao formam quadrilatero"<<endl;}
  if(z>0){cout<<"Perimetro do quadrilatero formado "<<z<<endl;}
}
```

Questão 2 (50 pontos)

Codifique um programa em C++ para auxiliar um leiloeiro a identificar os três maiores lances em suas atividades de vendas. O programa deverá receber vários lances para compra de um produto em leilão, onde cada lance é caracterizado por um código do comprador (inteiro, positivo e diferente de 0), e por um valor do lance, em reais. Quando um lance possuir 0 no código do comprador ou no valor da oferta , significa que o leilão acabou e o programa deverá mostrar os três maiores lances, em ordem decrescente, e encerrar o processamento. Para identificar os três maiores lances defina e utilize a função `Seleciona_Melhores_Ofertas()` que recebe cada lance (com código do comprador e valor do lance) juntamente com as váriáveis para armazenar os três melhores lances, cada um composto do código do comprador e do valor de seu lance. Caso o valor do lance seja menor ou igual ao maior valor de lance existente, a função deverá retornar 0 (significando que o lance é inválido), e em caso contrário deverá retornar +1. Caso a função retorne o valor 0, o programa deverá dar mensagem de "Lance inválido: Ignorado." , e em caso contrário deverá dar mensagem de "Lance OK: Processado." .

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Digite o Lance 1: 4 1012.50
Lance OK: Processado.
Digite o Lance 2: 1 980
Lance inválido: Ignorado.
Digite o Lance 2: 3 1050
Lance OK: Processado.
Digite o Lance 3: 4 1200.99
Lance OK: Processado.
Digite o Lance 4: 5 1250
Lance OK: Processado.
Digite o Lance 5: 3 1300
Lance OK: Processado.
Digite o Lance 6: 0 0
Foram processados 5 Lances.
1o. melhor Lance: 1300 do comprador 3
2o. melhor Lance: 1250 do comprador 5
3o. melhor Lance: 1200.99 do comprador 4
```

Outro exemplo de execução:

```
Digite o Lance 1: 4 1012.50
Lance OK: Processado.
Digite o Lance 2: 3 1050
Lance OK: Processado.
Digite o Lance 3: 0 0
Foram processados 2 Lances.
1o. melhor Lance: 1050 do comprador 3
2o. melhor Lance: 1012.50 do comprador 4
```



```
int seleciona_melhores_ofertas(int c, float l, int & c0, float & l0,
  int & c1, float & l1, int & c2, float & l2){
  if (l<=l0) {return 0;}
  if ((l>l0) && (l>l1) && (l>l2)){l2=l1; l1=l0; l0=l; c2=c1; c1=c0; c0=c; return 1;}
  if ((l>l1) && (l>l2)){l2=l1; l1=l; c2=c1; c1=c; return 1;}
  if (l>l2){l2=l; c2=c; return 1;}
}

int q2(){
  int lance=1,r;
  int c,c0,c1,c2;
  float l,l0,l1,l2;
  l0=l1=l2=0;
  c=l=1;
  while (1==1){
    cout<<"digite o lance "<<lance<<" ";
    cin>>c>>l;
    if ((c==0)||(l==0)){
      break;
    }
    r=seleciona_melhores_ofertas(c,l,c0,l0,c1,l1,c2,l2);
    if (r==0){
      cout<<"Lance invalido, ignorado"<<endl;
    }
    if (r==1){
      cout<<"Lance OK: processado"<<endl;
      lance++;
    }
  }
  cout<<"Foram processados "<<lance<<" lances."<<endl;
  if (l0>0){
    cout<<"1. melhor lance> "<<l0<<" do comprador "<<c0<<endl;
  }
  if (l1>0){
    cout<<"2. melhor lance> "<<l1<<" do comprador "<<c1<<endl;
  }
  if (l2>0){
    cout<<"3. melhor lance> "<<l2<<" do comprador "<<c2<<endl;
  }
}
```

Questão 1 (50 pontos)

Uma loja tem um plano de desconto progressivo para os seus clientes, onde quanto mais eles compram, mais desconto eles ganham. Além disso, a loja tem um plano de fidelidade que garante um desconto alternativo ainda maior aos clientes mais assíduos, de acordo com a tabela abaixo:

Valor da compra	Desconto Normal	Desconto Fidelidade
Até R$ 200,00	0%	2%
de 200,01 a 300,00	5%	7%
de 300,01 a 550,00	10%	13%
A partir de 550,01	15%	19%

Escreva um programa em C++ que peça ao usuário o valor de uma compra e o tipo do cliente que a efetuou. EM seguida o programa deve exibir o valor dado como desconto e o valor que foi efetivamente pago pelo cliente. Para calcular o desconto, deve ser definida e usada a funçao chamada `calc_desc()`, que recebe o valor total da compra do cliente e um valor que indique se o cliente tem plano de fidelidade (0 se não tiver, 1 se tiver) e que retorne o valor em reais do desconto aplicado ao valor da compra.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Valor da compra e tipo de cliente:
1072.30  1
Total de desconto: 203.73
Total a pagar: 868.56
```

Outro exemplo de execução:

```
Valor da compra e tipo de cliente:
247.37 0
Total de desconto: 12.36
Total a pagar: 235.01
```

```
float calc_desco(float co, int pf){
  float d=0;
  if ((co<=200)&&(pf==1)){d=0.02;}
  if ((co>200)&&(co<=300)&&(pf==0)){d=0.05;}
  if ((co>200)&&(co<=300)&&(pf==1)){d=0.07;}
```



```
  if ((co>300)&&(co<=550)&&(pf==0)){d=0.1;}
  if ((co>300)&&(co<=550)&&(pf==1)){d=0.13;}
  if ((co>550)&&(pf==0)){d=0.15;}
  if ((co>550)&&(pf==1)){d=0.19;}
  return co*d;
}

int q3(){
  int tc;
  float r,vc;
  cout<<"Valor da compra e tipo cliente"<<endl;
  cin>>vc>>tc;
  r=calc_desco(vc,tc);
  cout<<"Total do desconto: "<<r<<endl;
  cout<<"Valor a pagar: "<<vc-r;
}
```

Questão 2 (50 pontos)

Fulano, Beltrano e Cicrano estão disputando quem tem mais sorte. Para isso, eles pegaram um dado de seis lados e, um de cada vez, lançou o dado e anotou o número obtido. Eles fizeram diversas jogadas e agora querem descobrir quem foi o vencedor de cada rodada. Faça um programa completo em C++ que os auxilie a descobrir quantas rodadas cada um venceu. Obtenha do teclado o número de rodadas que foram realizadas e crie uma função chamada `vencedor_rodada()` que recebe como parâmetros os números que cada um tirou em uma determinada rodada. Faça sua função retornar 1, 2 ou 3 caso Fulano, Beltrano ou Cicrano tenha vencido aquela rodada, nessa ordem. Ao final do seu programa, mostre quantas rodadas cada jogador venceu.

Obs. 1: Assuma que houve apenas um vencedor, sem empates durante as rodadas.

OBS. 2: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Digite numero de rodadas do jogo: 3
Digite numeros obtidos pelos jogadores:
1 2 3
4 5 6
1 3 6
Fulano venceu 0 vezes
Beltrano venceu 0 vezes
Cicrano venceu 3 vezes
```

Outro exemplo de execução:

```
Digite numero de rodadas do jogo: 3
Digite numeros obtidos pelos jogadores:
6 2 1
1 2 4
4 1 5
Fulano venceu 1 vezes
Beltrano venceu 0 vezes
Cicrano venceu 2 vezes
```

```
// ------------------------------- 4 -----------------------------------
int vencedor_rodada(int f, int b, int c){
  if ((f>b)&&(f>c)){return 1;}
  if ((b>f)&&(b>c)){return 2;}
  if ((c>f)&&(c>b)){return 3;}
}

int q4(){
  int tf=0, tb=0, tc=0;
  int z,f,b,c,i,nr;
  cout<<"Digite numero de rodadas do jogo: ";
  cin>>nr;
  cout<<"Digite numeros obtidos pelos jogadores: "<<endl;
  for (i=0;i<nr;i++){
    cin>>f>>b>>c;
    z=vencedor_rodada(f,b,c);
    if (z==1){tf++;}
```



```
    if (z==2){tb++;}
    if (z==3){tc++;}
  }
  cout<<"Fulano venceu "<<tf<<" vezes"<<endl;
  cout<<"Beltrano venceu "<<tb<<" vezes"<<endl;
  cout<<"Cicrano venceu "<<tc<<" vezes"<<endl;
}
```

Questão 1 (50 pontos)

Faça um programa em C++ que verifique se a quantidade em estoque de um produto de uma loja acabou ou não toda vez que houve uma retirada. O programa deve pedir ao usuário a quantidade inicial do produto, e depois solicitar do usuário a quantidade vendida do produto, informando quando não houver quantidade suficiente em estoque para a venda. O programa termina mostrando a quantidade final do estoque. O programa deve usar a função `estoque()` para verificar se há quantidade suficiente e para atualizar a quantidade do estoque. Ela deve receber como parâmetros a quantidade em estoque e a quantidade vendida, e deve devolver o novo valor do estoque, que deve permanecer o mesmo caso o estoque não consiga cobrir a quantidade vendida.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Qtde. inicial do estoque: 20
Qtde. vendida: 12
Qtde. em estoque: 8
```

Outro exemplo execução:

```
Qtde. inicial do estoque: 13
Qtde. vendida: 20
Estoque insuficiente: 13
```

```
int estoque(int & es, int qv){
  if (qv<es) {
    es=es-qv; return 1;
  }
  else{
    return 0;
  }
}

int q5(){
  int est, v,r ;
  cout<<"Qtde inicial do estoque ";
  cin>>est;
  cout<<"Qtde vendida: ";
  cin>>v;
  r=estoque(est, v);
  if (r==0){cout<<"Estoque insuficiente: "<<est<<endl;}
  if (r==1){cout<<"Qtde. em estoque: "<<est<<endl;}
}
```

Questão 2 (50 pontos)

O *pôquer curitibano* é uma variante do jogo na qual cada jogador começa com duas cartas e, a cada rodada, pode escolher trocar uma de suas cartas por uma carta comprada do baralho. Nessa variante, apenas as cartas $2, 3, \cdots, 9, 10$ e A (Ás) são utilizadas, sendo que a pontuação de cada carta é igual ao seu valor, exceto a carta A que tem pontuação 14. Ganha o jogador que, após k rodadas, tenha na mão o par de maior valor ou as cartas de maiores valores (caso nenhum jogador tenha um par).

Faça um programa em C++ que auxilie um jogador a tomar as decisões mais lógicas numa partida de pôquer curitibano. Crie uma função chamada `troca_carta()` que recebe três parâmetros: as duas cartas que o jogador tem em mãos e a carta que foi comprada do monte nesse rodada. Essa função deve verificar se é vantajoso trocar alguma carta da mão pela comprada e retornar as cartas trocadas, se for o caso, e um inteiro representando se houve troca (1) ou não (0). No programa principal, leia o número k de rodadas da partida e dois inteiros representando as cartas que o jogador tem na mão no início do jogo (14 representa a carta **A**). Para cada uma das k rodadas, leia um inteiro representando a carta comprada. Utilizando a função `troca_carta()` seu programa deve obter, em cada rodada, uma mão melhor que a anterior, se possível. Caso haja troca de cartas a mesagem "apostar!" deve ser impressa. Caso contrário, a mensagem "mesa." é que deve ser mostrada. Se o jogador terminar a última rodada com um par de áses na mão, a mensagem "apostar tudo!!!" deve ser impressa, independente da rodada na qual o par foi obtido.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Número de rodadas: 3
Mão inicial: 2 7
Rodada 1: 3
Mão: 3 7 -- apostar!
Rodada 2: 3
Mão: 3 3 -- apostar!
Rodada 3: 14
Mão: 3 3 -- mesa.
```

Outro exemplo de execução:

```
Número de rodadas: 3
Mão inicial: 14 3
Rodada 1: 2
Mão: A 3 -- mesa.
Rodada 2: 14
Mão: A A -- apostar!
Rodada 3: 10
Mão: A A -- apostar tudo!!!
```

```
int troca_carta(int & c1, int & c2, int cc){
```



```
  if (c1==c2){return 0;}
  if ((c1==cc)&&(c2!=cc)){c2=cc; return 1;}
  if ((c2==cc)&&(c1!=cc)){c1=cc; return 1;}
  if ((cc>c1)&&(c2>c1)){c1=cc; return 1;}
  if ((cc>c2)&&(c1>c2)){c2=cc; return 1;}
  return 0;
}

int q6(){
  int i,nr,a,b,carta,z;
  cout<<"Numero de rodadas: ";
  cin>>nr;
  cout<<"Mao inicial: ";
  cin>>a>>b;
  for (i=1;i<=nr;i++){
    cout<<"Rodada "<<i<<" ";
    cin>>carta;
    cout<<endl;
    z=troca_carta(a,b,carta);
    if ((i==nr)&&(a==14)&&(b==14)){cout<<"A A -- apostar tudo!!!"<<endl;}
    if (z==1){cout<<a<<" "<<b<<" -- apostar!"<<endl;}
    if(z==0){cout<<a<<" "<<b<<" -- mesa."<<endl;}
  }
}
```

Questão 1 (50 pontos)

Faça um programa em C++ que receba do usuário a medida do raio de uma circunferência e uma escolha: 1 se o usuário deseja saber o perímetro da circunferência, 2 se o usuário deseja saber a área do círculo correspondente. O programa deverá usar uma função de nome `calcCirculo()` que recebe o raio e a escolha e retorna o valor do cálculo realizado. A entrada e saída de dados deve ser completamente manipulada pelo programa principal.
OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Informe raio da circunf.: 3
Deseja perimetro (1) ou área (2)?: 1
Perímetro = 18.8496
```

Outro exemplo de execução:

```
Informe raio da circunf.: 3
Deseja perimetro (1) ou área (2)?: 2
Área = 28.2743
```

```
float calcCirculo(float r, int tipo){
  if (tipo==1){return 2*3.14159265*r;}
  if (tipo==2){return 3.14159265*r*r;}
}

int q7(){
  float ra;
  int ti;
  cout<<"Informe o raio da circunf: ";
  cin>>ra;
  cout<<"Deseja perimetro (1) ou area (2) ?: ";
  cin>>ti;
  if (ti==1){cout<<"Perimetro: "<<calcCirculo(ra,ti)<<endl;}
  if (ti==2){cout<<"Area: "<<calcCirculo(ra,ti)<<endl;}
}
```


Questão 2 (50 pontos)

Daenerys e Drogo entendiados inventaram um jogo de advinhação. Neste jogo o oponente 1 inventa uma senha que consiste de um número de 3 dígitos não repetidos entre 1 e 4. São exemplos de senhas VÁLIDAS: 123, 231 e 412; As senhas 212 e 315 são senhas INVÁLIDAS. O oponente 2 deverá dar palpites para adivinhar a senha. Para cada palpite feito devem ser dadas as seguintes dicas: (a) quantos dígitos do palpite aparecem em qualquer posição na senha; (b) quantos dígitos do palpite aparecem na posição correta na senha (vide Exemplo de execução abaixo).

Sua missão é implementar este jogo no computador. Para isso você deve escrever uma função em C++ de nome `dicas()` que recebe como parâmetros uma *senha* e um *palpite*. A função deve devolver a quantidade de digitos do *palpite* que aparecem em *senha* (*acertdig*) e a quantidade de digitos do *palpite* que aparecem em *senha* na mesma posição (*acertdigpos*). Além disso, a função deve retornar o valor 0 (zero) se o palpite for incorreto, ou 1 (um) se o palpite estiver correto. Escreva também o programa principal que lê a senha do oponente 1 e depois lê os palpites do oponente 2 e, usando os resultados da função `dicas()`, imprime as dicas correspondentes até que este acerte a senha. O programa somente termina quando o oponente 2 adivinha a senha.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Oponente 1, entre sua senha: 341
Oponente 2, adivinhe a senha: 324
DICAS
Qtd. dígitos corretos: 2
Qtd. dígitos corretos na pos. correta: 1
Oponente 2, adivinhe a senha: 321
DICAS
Qtd. dígitos corretos: 2
Qtd. dígitos corretos na pos. correta: 2
Oponente 2, adivinhe a senha: 421
DICAS
Qtd. dígitos corretos: 2
Qtd. dígitos corretos na pos. correta: 1
Oponente 2, adivinhe a senha: 341
Você acertou. Parabéns!
```

```
int dicas(int senha, int palpi, int & adig, int & apos) {
  int s1,s2,s3,p1,p2,p3,dc=0,pc=0,saux,paux;
  saux=senha;
  paux=palpi;
  s1 = saux%10;
  saux=saux/10;
  s2 = saux%10;
  saux=saux/10;
  s3=saux;
  p1 = paux%10;
  paux = paux/10;
  p2=paux%10;
  paux=paux/10;
  p3=paux;
  if((p1==s1)||(p1==s2)||(p1==s3)){dc++;}
  if((p2==s1)||(p2==s2)||(p2==s3)){dc++;}
  if((p3==s1)||(p3==s2)||(p3==s3)){dc++;}
  if(p1==s1){pc++;}
  if(p2==s2){pc++;}
  if(p3==s3){pc++;}
  adig=dc;
  apos=pc;
  if (senha==palpi) {return 1;} else {return 0;}
}

int q8(){
  int ss,pp,qa,qp,z;
  qa=0;
  qp=0;
  cout<<"Oponente 1, entre a sua senha: ";
  cin>>ss;
  while(1==1){
    cout<<"Oponente 2, adivinhe a senha: ";
    cin>>pp;
    z=dicas(ss,pp,qa,qp);
    if (z==1){
      cout<<"Voce acertou, parabens!"<<endl;
      break;
    }
    cout<<"DICAS"<<endl;
    cout<<"Qtd digitos corretos: "<<qa<<endl;
```



```
    cout<<"Qtd na posicao correta: "<<qp<<endl;
  }
}
```

Questão 1 (50 pontos)

Escreva um programa em C++ que obtenha do usuário 4 valores inteiros e imprima na tela mensagem indicando se a soma dos dois maiores números é divisível pelo ou divisor do produto dos 3 menores valores informados. Para testar esta relação entre os valores deverá ser criada e usada a função `ehDivisor()`, que recebe 4 valores inteiros como parâmetros, e retorna o valor 1 (um) se a relação "soma divisível pelo produto" se verifica, 2 (dois) se a relação "soma divisor do produto" se verifica, ou 0 (zero) caso contrário.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

Indique 4 números inteiros: **1 7 3 2**
Propriedade não se verifica.

Outro exemplo de execução do programa:

Indique 4 números inteiros: **2 3 4 8**
12 é divisor de 24.

Outro exemplo de execução do programa:

Indique 4 números inteiros: **20 1 2 4**
24 é divisível por 8.

```
int ehdivisor(int a, int b, int c, int d, int & s, int & p){
  if (((a+b)>(a+c))&&((a+b)>(a+d))&&((a+b)>(b+c))&&((a+b)>(b+d))&&((a+b)>(c+d))){s=a+b;}
  if (((a+c)>(a+b))&&((a+c)>(a+d))&&((a+c)>(b+c))&&((a+c)>(b+d))&&((a+c)>(c+d))){s=a+c;}
  if (((a+d)>(a+c))&&((a+d)>(a+b))&&((a+d)>(b+c))&&((a+d)>(b+d))&&((a+d)>(c+d))){s=a+d;}
  if (((b+c)>(a+c))&&((b+c)>(a+d))&&((b+c)>(a+b))&&((b+c)>(b+d))&&((b+c)>(c+d))){s=b+c;}
  if (((b+d)>(a+c))&&((b+d)>(a+d))&&((b+d)>(b+c))&&((b+d)>(a+b))&&((b+d)>(c+d))){s=b+d;}
  if (((c+d)>(a+c))&&((c+d)>(a+d))&&((c+d)>(b+c))&&((c+d)>(b+d))&&((c+d)>(a+b))){s=c+d;}
  if (((a*b*c)<(a*b*d))&&((a*b*c)<(a*c*d))&&((a*b*c)<(b*c*d))){p=a*b*c;}
  if (((a*b*d)<(a*b*c))&&((a*b*d)<(a*c*d))&&((a*b*d)<(b*c*d))){p=a*b*d;}
  if (((a*c*d)<(a*b*d))&&((a*c*d)<(a*b*c))&&((a*c*d)<(b*c*d))){p=a*c*d;}
  if (((b*c*d)<(a*b*d))&&((b*c*d)<(a*c*d))&&((b*c*d)<(a*b*c))){p=b*c*d;}
  if ((s%p)==0) {return 1;}
  if ((p%s)==0) {return 2;}
  return 0;
}

int q9(){
  int a,b,c,d,s,p,x;
  cout<<"Indique 4 numeros inteiros: ";
  cin>>a>>b>>c>>d;
  x=ehdivisor(a,b,c,d,s,p);
  if (x==1){cout<<s<<" e divisivel por "<<p<<endl;}
  if (x==2){cout<<s<<" e divisor de "<<p<<endl;}
  if (x==0){cout<<"propriedade nao se verifica"<<endl;}
}
```


Questão 2 (50 pontos)

Faça um programa em C++ para realizar o agendamento de consultas ambulatoriais do SUS. Dois tipos de consultas são realizadas: (1) consulta normal e (2) consulta de retorno. Uma consulta normal possui 30 minutos de duração, enquanto uma consulta de retorno requer 15 minutos. Primeiro, o usuário deve informar a carga horária do médico em minutos. Depois, os agendamentos são realizados informando o tipo da consulta para cada paciente enquanto há carga horária suficiente para mais um agendamento. Ao final, o programa deve informar o número total de consultas agendadas.

Para resolver o problema proposto pede-se o desenvolvimento e uso de uma função de nome `agendamento()` que recebe como entrada o tipo de consulta e a carga horária disponível, e devolve esta carga horária atualizada de acordo com o tempo da consulta, devolvendo também o valor 1 (um) caso a consulta tenha sido agendada ou 0 (zero) caso contrário.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário.

Exemplo de execução:

```
Carga horaria: 90
Informe agendamentos:
1
Consulta agendada.
1
Consulta agendada.
2
Consulta agendada.
1
Consulta nao agendada.
2
Consulta agendada.
Total de consultas agendadas: 4
```

Outro exemplo de execução:

```
Carga horaria: 50
Informe agendamentos:
2
Consulta agendada.
1
Consulta agendada.
Total de consultas agendadas: 2
```

```
int agendamento(int & ch, int co){
  if ((co==1)&&(ch>=30)){ ch=ch-30; return 1;}
  if ((co==2)&&(ch>=15)){ ch=ch-15; return 1;}
  return 0;
}

int q10(){
  int r,a,t=0;
  int carga;
  cout<<"Carga Horaria ";
  cin>>carga;
  while (1==1){
    cout<<"Informe agendamentos: ";
    cin>>a;
    r=agendamento(carga,a);
    if (r==1){
      cout<<"Consulta agendada."<<endl;
      t++;
    }
    if (r==0){
      cout<<"Consulta nao agendada"<<endl;
    }
    if (carga<15){
      cout<<"Total de consultas agendadas: "<<t<<endl;
      break;
    }
  }
}
```

Questão 1 (50 pontos)

Faça um programa em C++ que ajude um usuário a calcular os juros de um investimento financeiro. Se a conta contém um saldo inferior a R$ 5.000,00, então o banco paga juros mensais de 2%. Para saldo superiores a R$ 5.000,00 o banco paga 3% de juros. O programa deve solicitar do usuário o saldo da conta e informar o saldo após a aplicação da taxa. Para fazer o cálculo solicitado, defina e chame no programa principal a função `calinveste()`, que recebe como parâmetro um valor de saldo e retorna o novo saldo.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Digite saldo atual: 12546
Novo saldo: 12922.38
```

Outro exemplo de execução:

```
Digite saldo atual: 4000
Novo saldo: 4080.00
```



```
#include<iostream>
using namespace std;
float calinveste(float saldo){
  if (saldo<5000){
    return saldo+saldo*0.02;
  }
  else {
    return saldo+saldo*0.03;
  }
}

int main() {
  float sa;
  cout<<"Digite saldo atual ";
  cin>>sa;
  sa = calinveste(sa);
  cout<<"Novo saldo "<<sa<<endl;
}
```

Questão 2 (50 pontos)

Considere as seguintes definições:

Definição 1: Um número inteiro n é um quadrado perfeito se existe outro inteiro i tal que $i * i = n$. Ou seja, n pode ser formado pelo quadrado de um número (menor) i.

Definição 2: Um número inteiro n é um cubo perfeito se existe outro inteiro i tal que $i * i * i = n$. Ou seja, n pode ser formado pelo cubo de um número (menor) i.

Usando C++, definir a função `quadsCubos()` e um programa que utilize a mesma.
A função deve receber como parâmetros dois valores inteiros a e b (estritamente positivos), e deve produzir dois resultados: $qQuadrados$, representando a quantidade de quadrados perfeitos no intervalo fechado $[a, b]$, e $qCubos$, representando a quantidade de cubos perfeitos no intervalo fechado $[a, b]$.
O seu programa principal deve ler do teclado dois números inteiros positivos (diferentes de 0) $n1$ e $n2$, e exibir o número de quadrados perfeitos e cubos perfeitos no intervalo $[n1, n2]$, valores estes calculados usando a função `quadsCubos()`.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Informe números do intervalo:  2 9
2 quadrados perfeitos nesse intervalo
1 cubos perfeitos nesse intervalo
```

Outro exemplo de execução:

```
Informe números do intervalo:  1 9
3 quadrados perfeitos nesse intervalo
2 cubos perfeitos nesse intervalo
```

Outro exemplo de execução:

```
Informe números do intervalo:  8 40
4 quadrados perfeitos nesse intervalo
2 cubos perfeitos nesse intervalo
```

```
#include<iostream>
#include<cmath>
using namespace std;
int equad(int x){
  int z;
  z=sqrt(x);
  return (z*z)==x;
}

int ecub(int x) {
  int z;
  z=cbrt(x);
  return (z*z*z)==x;
}

void quadsCubos(int a, int b, int & qtq, int & qtc){
  int i;
  qtq=0;
```



```
  qtc=0;
  i=a;
  while (i<=b) {
    if (equad(i)){
      qtq++;
    }
    if (ecub(i)){
      qtc++;
    }
    i=i+1;
  }
  return;
}

int main(){
  int k,w,z1,z2;
  cout<<"Informe numeros do intervalo ";
  cin>>k>>w;
  quadsCubos(k,w,z1,z2);
  cout<<z1<<" quadrados perfeitos"<<endl;
  cout<<z2<<" cubos perfeitos"<<endl;
}
```

Questão 1 (50 pontos)

Escreva uma função em C++ de nome `compat()` que recebe 3 números naturais e retorna o valor 1 (um) se eles são compatíveis, ou 0 (zero) caso contrário). Os números para serem compatíveis precisam ser divisíveis entre si, considerando que o divisor é sempre um número menor que o dividendo. Escreva também o programa em C++ que lê pelo teclado 3 números naturais e usando a função anteriormente especificada, verifique se são compatíveis ou não, imprimindo mensagens de acordo com o resultado.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:
Digite 3 números naturais: **5 10 20**
São compatíveis

Outro exemplo de execução:
Digite 3 números naturais: **24 3 6**
São compatíveis

Outro exemplo de execução:
Digite 3 números naturais: **4 3 7**
Não são compatíveis

```
#include<iostream>
using namespace std;
int compat(int a,int b, int c){
  if ((a<=b) && (a<=c) && (b<=c)){
    return ((b%a)==0) && ((c%a)==0) && ((c%b)==0);
  }
  if ((a<=b) && (a<=c) && (c<=b)){
    return ((b%a)==0) && ((c%a)==0) && ((b%c)==0);
  }
  if ((b<=a) && (b<=c) && (a<=c)){
    return ((a%b)==0) && ((c%b)==0) && ((c%a)==0);
  }
  if ((b<=a) && (b<=c) && (c<=a)){
    return ((a%b)==0) && ((c%b)==0) && ((a%c)==0);
  }
  if ((c<=a) && (c<=b) && (a<=b)){
    return ((a%c)==0) && ((b%c)==0) && ((b%a)==0);
  }
  if ((c<=a) && (c<=b) && (b<=a)){
    return ((a%c)==0) && ((b%c)==0) && ((a%b)==0);
  }
}

int main() {
  int x,y,z;
```



```
  cout<<"Digite 3 numeros naturais ";
  cin>>x>>y>>z;
  if (compat(x,y,z)){
    cout<<"Sao compativeis";
  }
  else {
    cout<<"Nao sao compativeis";
  }
}
```

Questão 2 (50 pontos)

Usando C++, definir a função `saqueNotas()` e um programa que utilize a mesma.

O programa principal simula um caixa eletrônico que possui somente notas de R$ 100 e notas de R$ 10. O programa deve solicitar que o usuário digite uma sequência de valores de saque e, a cada novo saque, deverá ser impresso quantas notas de R$ 100 e quantas notas de R$ 10 deverão ser entregues ao cliente. No início do programa deverá ser informado qual o valor disponível no caixa eletrônico, que deverá ser reduzido a cada novo saque efetuado com sucesso. Nenhum saque poderá entregar mais que 9 notas de R$ 10. O programa finaliza quando o caixa estiver sem saldo.

A função `saqueNotas()`, que recebe o saldo s e a quantidade r a ser retirada, deverá calcular e devolver a quantidade de notas de cada tipo de cédula, e deverá retornar também o valor 1 (um) se há saldo em caixa para o saque e 0 (zero) caso contrário.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Saldo em caixa: 1000
Valor do saque: 300
Cédulas: 3 de R$ 100 e 0 de R$ 10.
Valor do saque: 700
Cédulas: 7 de R$ 100 e 0 de R$ 10.
Sem saldo, caixa desativado.
```

Outro exemplo de execução:

```
Saldo em caixa: 1000
Valor do saque: 800
Cédulas: 8 de R$ 100 e 0 de R$ 10.
Valor do saque: 210
Saldo insuficiente.
Valor do saque: 190
Cédulas: 1 de R$ 100 e 9 de R$ 10.
Valor do saque: 10
Cédulas: 0 de R$ 100 e 1 de R$ 10.
Sem saldo, caixa desativado.
```

```
#include<iostream>
using namespace std;
int saqueNotas(float s, float q, int & n100, int & n10 ){
  if (q<=s) {
    n100 = q/100;
    n10=(q-(n100*100))/10;
    return 1;
  }
  else {
    return 0;
  }
}

int main(){
  float saldo,saque;
  int resultado,notas100,notas10;
  cout<<"Saldo em caixa ";
  cin>>saldo;
  while (saldo>0) {
    cout<<"Valor do saque ";
    cin>>saque;
    resultado=saqueNotas(saldo,saque,notas100,notas10);
    if (resultado==1){
      cout<<"Cedulas: "<<notas100<<" de 100 e "<<notas10<<" de 10"<<endl;
```



```
      saldo = saldo - saque;
    }
    else {
      cout<<"Saldo insuficiente "<<endl;
    }
  }
  if (saldo<=0){
    cout<<"Sem saldo, caixa desativado"<<endl;
  }
}
```

Questão 1 (50 pontos)

Uma data mágica é aquela em que o produto do dia pelo mês tem seus dois últimos dígitos iguais aos dois dígitos finais do ano. Por exemplo, 10 de Junho de 1960 é uma data mágica, pois Junho é o 6^o mês e 10×6 é 60, que é igual aos dois dígitos finais do ano.

Faça um programa em C++ que peça ao usuário uma data na forma *dia mes ano* (onde $1 \leqslant dia \leqslant 31$, $1 \leqslant mes \leqslant 12$, e $1900 \leqslant ano \leqslant 2500$) e mostre uma mensagem dizendo se a data informada é ou não mágica, ou se é inválida.

Para verificar se uma data é mágica ou inválida, deverá ser criada e chamada a função de nome `ehMagico()` que recebe como parâmetros três valores inteiros representando respectivamente *dia*, *mes* e *ano*, e que retorna o valor 1 (um) se a data é mágica, ou 0 (zero) caso contrário. Caso o valor de qualquer um dos parâmetros esteja fora dos intervalos indicados acima, a função deve retornar o valor -1 (um negativo).

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Data:  10 6 1960
Data é mágica
```

Outro exemplo de execução:

```
Data:  30 6 1960
Data não é mágica
```

Outro exemplo de execução:

```
Data:  30 12 1960
Data é mágica
```

```
#include<iostream>
using namespace std;
int ehMagico(int dia, int mes, int ano){
  int oba,rabo;
  if (((dia<1)||(dia>31)) || ((mes<1)||(mes>12)) || ((ano<1900)||(ano>2500))) {
    return -1;
  }
  else {
    oba = (dia*mes)%100;
    rabo = ano%100;
    if (rabo==oba){
      return 1;
    }
    else{
      return 0;
    }
  }
}

int main() {
  int d,m,a,dica;
  cout<<"Data: ";
  cin>>d>>m>>a;
  dica = ehMagico(d,m,a);
  if (dica==1){
    cout<<"Data e magica"<<endl;
  }
  else {
    if (dica==0) {
      cout<<"Data nao e magica"<<endl;
    }
    else {
      cout<<"Data invalida "<<endl;
    }
```



```
  }
}
```

Questão 2 (50 pontos)

Escrever um programa em C++ que leia do teclado um conjunto de pares de números O término da leitura de dados deve ser indicado por um par de numeros zero. Cada par é composto por um número real r e outro inteiro i, este último não devendo ser nulo. O programa deverá analisar cada par (r, i) e exibir na tela as partes inteira e fracionária de r, exibindo também mensagem indicando se a parte inteira de r é ou não divisível por i.

Ao final do programa, o total tp de pares que atenderam à propriedade verificada e o total geral tg de pares válidos deverão ser mostrados.

Para verificar se a parte inteira de um número real é ou não divisível por um outro número inteiro, deverá ser utilizada a função `analisaNumeros()`, que você deverá desenvolver. Esta função deverá receber como parâmetros um número real nr e um número inteiro div, e devolver como resultados as partes inteira pi e fracionária pf do número real nr, e um inteiro at cujo valor é 0 (zero) se pi não é divisível por div ou 1 (um) caso contrário.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Informe par (real e inteiro):  5.389 3
 Parte inteira: 5
 Parte fracionária: 0.389
 5 não é divisível por 3
Informe par (real e inteiro):  12.2525 6
 Parte inteira: 12
 Parte fracionária: 0.2525
 12 é divisível por 6
Informe par (real e inteiro):  7 0
 Entrada inválida
Informe par (real e inteiro):  -4.567 2
 Parte inteira: -4
 Parte fracionária: -0.567
 -4 é divisível por 2
Informe par (real e inteiro):  -0.5 -2
 Parte inteira: 0
 Parte fracionária: -0.5
 0 é divisível por -2
Informe par (real e inteiro):  15 6
 Parte inteira: 15
 Parte fracionária: 0
 15 não é divisível por 6
Informe par (real e inteiro):  0 0
Pares validos: 5
Pares com a propriedade: 3
```

```
#include<iostream>
using namespace std;
int analisaNumeros(float nr, int ni, int & pi, float & pf, int & enaoe) {
  pi = nr;
  pf = nr - pi;
  if ((pi%ni)==0) {
    enaoe = 1;
  }
  else {
    enaoe =0;
  }
}

int main() {
  float numeror,partef;
  int numeroi,partei,simnao;
  int tp,tg;
  tp=0;
  tg=0;
  numeror=1;
  numeroi=1;
  while (1==1) {
    cout<<"Informe par (real e inteiro): ";
    cin>>numeror>>numeroi;
    if ((numeroi==0) && (numeror!=0)){
      cout<<"Entrada invalida"<<endl;
    }
    else {
      if ((numeror==0) && (numeroi==0)){
        cout<<"Pares validos "<<tg<<endl;
        cout<<"Pares com a propriedade "<<tp<<endl;
        return 0;
      }
      else {
        tg++;
        analisaNumeros(numeror,numeroi,partei,partef,simnao);
        cout<<"Parte inteira "<<partei<<endl;
```



```
        cout<<"Parte fracionaria "<<partef<<endl;
        if ((partei%numeroi)==0) {
          cout<<partei<<" e divisivel por "<<numeroi<<endl;
          tp++;
        }
        else {
          cout<<partei<<" nao e divisivel por "<<numeroi<<endl;
        }
      }
    }
  }
}
```

2.4 2018

Questão 1 (50 pontos)

A empresa DrOBER deseja que você faça um programa para fornecer o custo de uma viagem sua em um drone, similarmente a uma bem conhecida empresa de transporte de passageiros. O custo da viagem com o DrOBER está relacionado a: distância a ser percorrida, peso e idade do passageiro.

Faça um programa em C++ que solicite do usuário idade (anos), peso (quilos) e a distância (metros) a ser percorrida por um passageiro e mostre na tela o custo desta viagem.

Para o cálculo do custo da viagem deverá ser criada uma função de nome `custo()` a ser usada em seu programa. Esta função retorna o custo em reais da viagem para um passageiro e recebe como parâmetros a idade em anos, o peso em Kg, e a distância a ser percorrida em metros. O custo por kilômetro percorrido pelo drone é de R$10,00. O valor mínimo para usar o sistema é R$1,00. Passageiros com mais de 60 anos pagam apenas metade do custo; e passageiros com mais de 100 Kg pagam 2 vezes a mais do que os demais. Para valores nulos ou negativos de entrada deverá ser cobrado o valor mínimo. Considere também que os valores calculados são sempre inteiros, para facilitar o troco.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Digite idade (anos): 80
Digite peso (Kg): 60
Digite distância (m): 2003
Seu custo será de: R$ 10
```

Outro exemplo de execução:

```
Digite idade (anos): 50
Digite peso (Kg): 120
Digite distância (m): 600
```

Questão 2 (50 pontos)

Escreva um programa em C++ que obtém de um usuário vários conjuntos de 3 valores inteiros representando as mãos de cartas de um jogador em um novo jogo chamado BRUCO. Ao final do jogo, o programa deve indicar a pontuação final do jogador e se ele ganhou ou não o jogo. Um jogador é ganhador se sua pontuação total é maior que um valor informado pelo usuário no início do programa. O final do jogo é indicado quando forem informadas 3 mãos em que cada mão totalize menos que 10 pontos.

Para calcular a pontuação em uma determinada mão e se esta pontuação é menor que 10, o programa deve definir e chamar a função `bruco()`, que recebe as três cartas de uma mão do jogo, e devolve um valor inteiro com a soma dos valores das cartas, e um segundo valor que deve ser 1 (um) se a soma calculada é menor que 10, ou 0 (zero) caso contrário.

Considere a seguinte convenção para as cartas (e também sua pontuação): $1 = AS, 2 = 2, 3 = 3 \cdots 10 = 10, 11 = Valete, 12 = Dama, 13 = Rei$.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Indique pontos para ganhar: 200
Indique suas cartas: 1 2 4
Indique suas cartas: 6 5 1
Indique suas cartas: 1 3 2
Indique suas cartas: 8 2 2
Indique suas cartas: 4 1 3
Você perdeu com 45 pontos!!
```

Outro exemplo de execução:

```
Indique pontos para ganhar: 50
Indique suas cartas: 1 2 4
Indique suas cartas: 6 5 4
Indique suas cartas: 2 3 1
Indique suas cartas: 6 5 1
Indique suas cartas: 4 3 7
Indique suas cartas: 3 1 2
Você ganhou com 60 pontos!!
```


Questão 1 (50 pontos)

Dona Gertudes comprar equipamentos de luta greco romana eletrônicamente no Site Acme. Este site oferece descontos e cobra o frete de acordo com a quantidade de produtos adquiridos de acordo com a tabela abaixo:

quantidade	desconto(%)	frete por unidade (R$)
1	0	30,00
2	10	15,00
3	20	GRÁTIS
acima de 3	30	GRÁTIS

Infelizmente o site não foi bem programado e Dona Gertudes não quer ser enganada. Vamos ajudá-la a conferir se o valor de suas compras está correto.

Para isso você deve escrever um programa em C++ que lê o valor e a quantidade que produto desejado bem como o valor calculado pelo Site Acme. Em seguida, o programa calcula o valor correto da compra e imprime este valor. Caso o valor da Acme esteja incorreto o programa indica quanto foi cobrado a menos ou a mais.

Para o cálculo do valor correto da compra, deve ser definida e usada a função `compra()`, que recebe a quantidade e o valor de um determinado produto e retorna o valor total da compra aplicando o frete e o desconto apropriado conforme a tabela apresentada. O desconto incide apenas sobre o valor do produto.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Valor do produto?: 100.0
Quantidade do produto?: 2
Valor calculado pela ACME?: 395.00
Valor correto da compra: 375.00
ACME está cobrando 20.00 a mais
```

Outro exemplo de execução:

```
Valor do produto?: 210.0
Quantidade do produto?: 3
Valor calculado pela ACME?: 500.00
Valor correto da compra: 504.00
ACME está cobrando 4.00 a menos
```

Outro exemplo de execução:

```
Valor do produto?: 50.0
Quantidade do produto?: 4
Valor calculado pela ACME?: 140.00
Valor correto da compra: 140.00
ACME está cobrando corretamente
```

Questão 2 (50 pontos)

Se multiplicarmos 37 por alguns números, obtemos números cujos algarismos, quando somados, resultam no mesmo número que foi multiplicado pelo 37. Por exemplo, se tomarmos o número 15, multiplicando-o por 37, obtemos 555. Somando-se 5 + 5 + 5 resulta em 15.

Escreva um programa C++ que lê um número inteiro positivo e mostre na tela o produto deste número por 37 e uma mensagem indicando se o número fornecido apresenta a propriedade descrita anteriormente (seguindo exatamente o formato de mensagens do exemplo de execução).

Para obter o cálculo da multiplicação de um número inteiro por 37 e a soma dos algarismos deste resultado deverá ser criada uma função de nome `mult_soma()` que deve ser usada em seu programa.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Informe um número: 24
24 x 37 = 888
Soma de digitos de 888 é 24
```

Outro exemplo de execução:

```
Informe um número: 234
234 x 37 = 8658
Soma de digitos de 8658 não é 234
```

Questão 1 (50 pontos)

Um estacionamento cobra uma taxa mínima de R$6,00 para estacionar por três horas. Um adicional de R$ 2,00 por hora não necessariamente inteira é cobrado após as três primeiras horas. O valor máximo para qualquer período de 24 horas é R$ 30,00. O programa não deve aceitar períodos de estacionamento acima de 24 horas.

Escreva um programa em C++ que calcule e exiba preço cobrado de um cliente que estacionou nessa garagem. O programa solicita do usuário as horas de estacionamento do cliente e então exibe a cobrança feita.

Para determinar o valor de cobrança para um cliente, o programa deve definir e usar a função `valorAPagar()`, que recebe do programa principal o número de horas que o cliente ficou estacionado e retorna o valor a ser pago pelo cliente.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Período do cliente (horas): 6
Cliente deve pagar R$ 12.00
```

Outro exemplo de execução:

```
Período do cliente (horas): 20
Cliente deve pagar R$ 30.00
```

Outro exemplo de execução:

```
Período do cliente (horas): 26
Período inválido !!!
```

Questão 2 (50 pontos)

Uma empresa de energia elétrica calcula quanto será cobrado por residência baseado no consumo de energia em quilowatts (kw) e no salário cadastrado, sendo que cada 100 kw custa 1/7 do salário informado. Para salários acima de 2500,00 reais, deve ser calculado um desconto de 10% em cima do valor a ser pago. Além disso, se a residência tiver mais que 3 moradores, deve ser aplicado mais 8% de desconto no valor a ser pago.

Crie um programa em C++ que receba vários conjuntos de dados de residências contendo o valor do salário cadastrado, o gasto de energia (em kw) e o número de moradores. Para cada conjunto, o programa deve calcular e exibir na tela o valor total a ser pago pela residência e o custo por morador.

Para obter estes cálculos, deve ser definida e chamada no programa a função `calcval()`, que recebe como parâmetros salário, gasto de energia e número de moradores e deve devolver o valor a ser pago, bem como o custo por morador.

A cada conjunto de dados o programa deve perguntar se o usuário deseja repetir o processo com novos valores (Digitando a opção 1) ou sair do programa (Digitando 0).

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Valor do salario (R$): 2700.00
Energia gasta (kw): 180
Número de moradores: 1
Valor a ser pago: 624.857
Custo por morador: 624.857
Digite 0 (encerrar) ou 1 (novos dados): 1

Valor do salario (R$): 2500.00
Energia gasta (kw): 120
Número de moradores: 3
Valor a ser pago: 428.571
Custo por morador: 142.857
Digite 0 (encerrar) ou 1 (novos dados): 0
```

Questão 1 (50 pontos)

Num jogo de computador existe um labirinto com 50 alvos e cada jogador tem até 100 segundos para atingir todos os alvos. Construa um programa em C++ que leia o número de acertos e o tempo decorrido e usando a função `pontuacao()`, escreva o número de pontos que o jogador obteve. Construa a função `pontuacao()` que receba como parâmetro o tempo em segundos e o número de acertos e de acordo com a tabela abaixo, retorne o número de pontos obtidos.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

acertos (n)	tempo (t)	pontos
$0 < n \leqslant 50$	$t > 100$	$n * 5$
$0 < n \leqslant 30$	$0 < t \leqslant 50$	$n * 30$
$0 < n \leqslant 30$	$50 < t \leqslant 100$	$n * 20$
$30 < n \leqslant 50$	$0 < t \leqslant 50$	$n * 50$
$30 < n \leqslant 50$	$50 < t \leqslant 100$	$n * 40$

Exemplo de execução:

```
Digite o numero de acertos: 35
Digite o tempo: 45
A pontuacao foi: 1750
```

Outro exemplo de execução:

```
Digite o numero de acertos: 25
Digite o tempo: 60
A pontuacao foi: 500
```

Outro exemplo de execução:

```
Digite o numero de acertos: 35
Digite o tempo: 60
A pontuacao foi: 1400
```

Questão 2 (50 pontos)

A empresa de taxi Vambora sempre foi considerada à frente do seu tempo. Recentemente ela comprou um lote de 80 carros biflex (gasolina e eletricidade). Como o mercado ainda está um pouco bagunçado, os preços desses 2 insumos variam diariamente, o que dificulta o planejamento de uso de combustíveis pela Vambora. Você foi contratado pela empresa para escrever uma função de nome `precos()` que receba requisições de viagem (quilometragem) e os custos dos 2 modais e devolva qual o custo dessa viagem para cada um dos 2 combustíveis.

Cada consumo e custo deve ser calculado como segue:

1. **gasolina:** 11 km/litro × preco do litro de gasolina, sem adicional;
2. **eletricidade:** 3 km/kw × preço do quilowatt, mais um custo fixo de 5 reais por viagem.

Deve também ser definido o programa principal em C++ que pede ao usuário os valores diários de custo de 1 litro de gasolina bem como o custo do kilowatt. A seguir, o programa deve efetuar um ciclo indeterminado de viagens a fazer (que acaba quando for lida uma viagem de distância negativa). Para cada viagem, o programa deve chamar a função `precos()` acima e depois imprimir os custos (calculados e devolvidos pela função) nos 2 modais.

OBS.: A função NÃO DEVE mostrar dados na tela ou solicitar valores do usuário. A exibição e leitura de dados devem ser feitas exclusivamente pelo programa principal.

Exemplo de execução:

```
Preco de gasolina (R$/l): 4.10
Preco eletricidade (R$/kw): 1.56
Distancia da viagem: 100
    Preco em gasolina 37.2727
    Preco em eletricidade 57.00
Distancia da viagem: 1000
    Preco em gasolina 372.727
    Preco em eletricidade 525.00
Distancia da viagem: -1
```

3 Prova 3

3.1 2014 - s1

Questão 1 (50 pontos)

Escreva um programa em C++ que leia um vetor de números inteiros de tamanho N (N estabelecido via diretiva `#define`). O programa deve calcular quantas vezes ocorrem valores repetidos em posições consecutivas ("vizinhas") no vetor. Além disso, o programa deve atribuir valor 0 nestas posições do vetor. Ao final, o programa deve exibir na tela o conteúdo do vetor.

OBS.: Para a exibição do conteúdo do vetor na tela, considere que já existe a função `void imprime_vetor (int vet[])`. Sua solução deve apenas chamar esta função.

Exemplo de execução (com N=10):

```
Informe vetor:
2 3 3 4 5 1 5 5 7 8
Qtde. de repetições: 2
Vetor final:
2 0 0 4 5 1 0 0 7 8
```

Outro exemplo de execução (com N=7):

```
Informe vetor:
2 3 3 3 2 1 5
Qtde. de repetições: 2
Vetor final:
2 0 0 0 2 1 5
```

```
#include<iostream>
#define N 10
using namespace std;
void imprime_vetor(int v[]){
  int i;
  for (i=0;i<N;i++){
    cout<<v[i]<<' ';
  }
}

int main() {
  int vet[N]={2,3,3,4,5,1,5,5,7,8};
  int vet[N]={1,3,3,3,3,3,3};
  int i,j,k,ctd;
  for (i=0;i<N;i++){
    cout<<"Informe "<<i+1<<" ";
    cin>>vet[i];
  }
  ctd=0;
  i=0;
  while(i<N){
    j=1;
    while ((vet[i]==vet[i+j])&&(i+j<N)){
      j=j+1;
    }
    ctd=ctd+j-1;
    if (j!=1){
      k=0;
      while (k<j){
        vet[i]=0;
        k++;
        i++;
        cout<<"valor de i "<<i<<endl;
      }
      i--; // foi incluido
    }
    i++;
    cout<<"valor de i fora"<<i<<endl;
  }
  cout<<"Quantidade de repeticoes "<<ctd<<endl;
```



```
  cout<<"Vetor final: "<<endl;
  imprime_vetor(vet);
}
```

Questão 1 (50 pontos)

A PROG (Pavês, Rocamboles e Outras Guloseimas) está fazendo uma nova promoção e Teobaldo, que é louco por promoções, quer participar juntando cupons da promoção. Para ter direito a 1 cupom, é preciso juntar embalagens de cada um dos M tipos de produto que a PROG fabrica. Por exemplo, se há 6 tipos de produto e se Teobaldo tiver 3 embalagens de cada um dos produtos P_1, P_2, P_3, P_4 e P_5, e 2 embalagens do produto P_6, ele pode obter no máximo 2 cupons da promoção, já que falta uma embalagem do produto P_6 para obter o terceiro cupom.

Escreva um programa em C++ que recebe do usuário M números inteiros (M estabelecido via diretiva `#define`), cada um representando a quantidade de embalagens de cada produto que o usuário tem. O primeiro número neste conjunto representa a quantidade de embalagens do produto P_1 que o usuário possui; o segundo número representa a quantidade de embalagens do produto P_2, e assim por diante até o produto P_M. Ao final, o programa deve imprimir o número de cupons que o usuário vai obter e quantas embalagens de cada produto restarão depois de obtidos os cupons.

OBS.: Para a exibição do conteúdo do vetor na tela, considere que já existe a função `void mostra_embalagens (int emb[])`. Sua solução deve apenas chamar esta função.

Exemplo de execução: (com M=6)

```
Qtde. embalagens: 5 3 4 6 2 2
Qtde. de cupons: 2
Embalagens restantes: 3 1 2 4 0 0
```

Outro exemplo de execução: (com M=6)

```
Qtde. embalagens: 10 5 21 3 0 11
Qtde. de cupons: 0
Embalagens restantes: 10 5 21 3 0 11
```

```
#include<iostream>
#define M 6
using namespace std;
void imprime(int vx[]){
  int j;
  for (j=0;j<M;j++){
    cout<<vx[j]<<' ';
  }
}

int main() {
  int i,j,men;
  int v[M];
  for (i=0;i<M;i++){
    cout<<"Informe "<<i+1<<' ';
    cin>>v[i];
  }
  men=99999;
  for (i=0;i<M;i++){
    if (v[i]<men){
      men=v[i];
    }
  }
  cout<<"Quantidade de embalagens: ";
  imprime(v);
  cout<<endl;
  cout<<"Quantidade de cupons: ";
  cout<<men<<endl;
  cout<<"Embalagens restantes: ";
  for (i=0;i<M;i++){
    v[i]=v[i]-men;
  }
  imprime(v);
  cout<<endl;
}
```


Questão 1 (50 pontos)

Escreva um programa em C++ para ler um vetor de reais de tamanho N (N estabelecido via diretiva `#define`) e um valor X. O programa deve exibir na tela *ACHEI* se o valor X existir pelo menos uma vez no vetor e *NÃO ACHEI* caso contrário. O programa deve também alterar o vetor dividindo todos os seus valores por X. Ao final, o programa deve exibir na tela o conteúdo do vetor.

OBS.: Para a exibição do conteúdo do vetor na tela, considere que já existe a função `void mostra_vetor (float vet[])`. Sua solução deve apenas chamar esta função.

Exemplo de execução: (com N=6)

```
Informe vetor: 4 3 12 6 9 3
Informe X: 3
ACHEI.
Vetor final: 1.33 1 4 2 3 1
```

```
#include<iostream>
#define M 6
using namespace std;
void imprime(float vx[]){
  int j;
  for (j=0;j<M;j++){
    cout<<vx[j]<<' ';
  }
}

int main() {
  int i,j,achei;
  float X;
  float v[M];
  for (i=0;i<M;i++){
    cout<<"Informe "<<i+1<<' ';
    cin>>v[i];
  }
  cout<<"Informe X ";
  cin>>X;
  achei=0;
  for (i=0;i<M;i++){
    if (v[i]==X){
      achei=1;
    }
  }
  if (achei==1){
    cout<<"ACHEI"<<endl;;
  }
  else {
    cout<<"NAO ACHEI"<<endl;
  }
  for (i=0;i<M;i++){
    v[i]=v[i]/X;
  }
  imprime(v);
}
```


Questão 1 (50 pontos)

Escreva um programa em C++ que leia um vetor de inteiros de tamanho N (N estabelecido via diretiva `#define`) e um valor X. O programa deve calcular o valor que deve ser somado a cada elemento do vetor para que este atinja o valor X. Este resultado deve substituir o valor correspondente no vetor. Além disso, deve ser calculada e exibida na tela a média aritmética dos valores finais do vetor. Ao final, o programa deve exibir na tela o conteúdo do vetor já alterado.

OBS.: Para a exibição do conteúdo do vetor na tela, considere que já existe a função `void mostra_vetor (int vet[])`. Sua solução deve apenas chamar esta função.

Exemplo de execução: (com N=6)

```
Informe vetor: 4 3 12 6 9 3
Informe X: 3
Média: -3
Vetor final: -1 0 -9 -2 -6 0
```

Outro exemplo de execução: (com N=6)

```
Informe vetor: 42 23 21 36 79 53
Informe X: 50
Média: 7.666
Vetor final: 8 27 29 14 -29 -3
```

```
#include<iostream>
#define M 6
using namespace std;
void imprime(float vx[]){
  int j;
  for (j=0;j<M;j++){
    cout<<vx[j]<<' ';
  }
}

int main() {
  int i;
  float v[M];
  float som, X;
  for (i=0;i<M;i++){
    cout<<"Informe "<<i+1<<' ';
    cin>>v[i];
  }
  cout<<"Informe X ";
  cin>>X;
  for (i=0;i<M;i++){
    v[i]=X-v[i];
  }
  som=0;
  for (i=0;i<M;i++){
    som=som+v[i];
  }
  som=som/M;
  cout<<"Media: "<<som<<endl;
  cout<<"Vetor final: ";
  imprime(v);
}
```


Questão 1 (50 pontos)

Escrever um programa em C++ que leia do teclado um vetor ordenado descendentemente (seu programa não precisa verificar isto) de N números reais (N estabelecido via `#define`), calcule e mostre a média aritmética, a mediana e a diferença entre a média e a mediana dos valores dos elementos do vetor. Para o cálculo da mediana, se o número de elementos do vetor for ímpar, a mediana é o valor do elemento central do vetor; caso par, é dada pela média entre os dois valores dos elementos centrais do vetor. Após a obtenção destes valores, elementos fora do intervalo [média, mediana] ou [mediana, média] (dependendo da ordem das medidas centrais obtidas), devem ter seus valores zerados.

OBS.: O vetor alterado deverá ser mostrado na tela utilizando a função com o protótipo `void imprime(float v[]);`, que não precisa ser codificada em sua resposta. Sua solução deve apenas chamar esta função.

Exemplo de execução: (com N=8)

```
Vetor:
10.5  9.8  8  9.4  7.3  6  5.2  5
Média aritmética: 7.65
Mediana: 8.35 (= (9.4 + 7.3) / 2 )
Média - mediana: -0.70
Vetor alterado:
0  0  8  0  0  0  0  0
```

Outro exemplo de execução: (com N=9)

```
Vetor:
9.9  9.8  9.4  8  7.3  6  5.2  5  4.9
Média aritmética: 7.344
Mediana: 7.3
Média - mediana: 0.044
Vetor alterado:
0  0  0  0  7.3  0  0  0  0
```

```
#include<iostream>
#define n 9
using namespace std;
void impvet(float x[n]){
  int i,j;
  for (i=0;i<n;i++){
    cout<<x[i]<<' ';
  }
}

int main(){
  float v[n];
  int i,j;
  float soma,media,mediana;
  soma=0;
  for (i=0;i<n;i++){
    cout<<"informe "<<i+1<<' ';
    cin>>v[i];
    soma=soma+v[i];
  }
  if (n%2==0){
    i=(n/2)-1;
    j=i+1;}
  else {
    i=j=n/2;
  }
  media=soma/n;
  mediana=(v[i]+v[j])/2;
  if (media>mediana){
    swap(media,mediana);
  } // agora mediana,media esta em ordem...
  for (i=0;i<n;i++){
    if ((v[i]<mediana)||(v[i]>media)){
      v[i]=0;
    }
  }
```



```
  cout<<"Media aritmetica: "<<media;
  cout<<"Mediana: "<<mediana;
  cout<<"Media - Mediana: "<<media-mediana;
  cout<<"Vetor alterado: "<<endl;
  impvet(v);
}
```

Questão 2 (50 pontos)

Escreva um programa em C++ que deverá ler uma matriz com M linhas x N colunas (M e N estabelecidos via `#define`) de valores inteiros, onde cada elemento da matriz representa uma cota (em centímetros) do correspondente metro quadrado de um terreno de $M \times N$ metros que precisa ser aplainado. Ao final, o programa deve imprimir, como no exemplo abaixo, o volume de terra (em metros cúbicos) que será necessário adicionar (ou retirar) para fazer a terraplenagem de toda a área do terreno, visando colocar todo o terreno em uma cota uniforme de 128 centímetros.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Sua solução deve apenas chamar esta função.

Exemplo de execução (com M=3 e N=6):

```
Informe a matriz original:
19   68   58   80   21   177
66   28   30   32   34   156
78   58   60   52   74   172
Qtde. de terra a acrescentar: 10.41 m3
```

Outro exemplo de execução (com M=3 e N=6):

```
Informe a matriz original:
219  168  158  180  125  217
166  128  130  232  234  136
278  158  160  152  214  152
Qtde. de terra a retirar: 9.03 m3
```

```
#include<iostream>
#define m 3
#define n 6
using namespace std;
void impvet(float x[m][n]){
  int i,j;
  for (i=0;i<m;i++){
    for (j=0;j<n;j++){
      cout<<x[i][j]<<' ';
    }
    cout<<endl;
  }
}

int main(){
  float v[m][n]={{19,68,58,80,21,177},{66,28,30,32,34,156},{78,58,60,52,74,172}};
  float v[m][n]={{219,168,158,180,125,217},{166,128,130,232,234,136},
  {278,158,160,152,214,152}};
  float qtd;
  int i,j;
  for (i=0;i<m;i++){
    for (j=0;j<n;j++){
      if (v[i][j]>128){
        qtd=qtd+((128-v[i][j])/100);
      }
      else{
        qtd=qtd-((v[i][j]-128)/100);
      }
    }
  }
  impvet(v);
  if (qtd>0){
    cout<<"Qtdade de terra a acrescentar: "<<qtd<<endl;}
  else{
    cout<<"Qtdade de terra a retirar: "<<-qtd<<endl;
```



```
  }
}
```

Questão 2 (50 pontos)

Escreva um programa em linguagem C++ que deverá ler uma matriz com M linhas x N colunas (M e N estabelecidos via `#define`) de valores reais **não-nulos**[]. Em seguida, o programa deve **criar uma nova matriz**[alterar a matriz] de forma a dividir os valores dos elementos de cada linha i da matriz **original** [] pelo valor do elemento da coluna i da mesma linha e, no caso de não existir a coluna, a divisão deve ser pelo último valor da mesma linha. Ao final, o programa deve mostrar na tela a matriz resultante e a quantidade de valores na matriz resultante que estão acima de 1 (um).

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (float matriz[][N])`. Para a impressão de uma matriz, considere que já existe a função `void imprime_matriz (float matriz[][N])`. Sua solução deve apenas chamar estas funções.

Exemplo de execução (com M=4 e N=3):

```
Informe a matriz:
37  28  41
17  41  53
25  93  31
93  19  26
Matriz resultante[alterada]
1.000  0.757   1.108
0.414  1.000   1.293
0.806  3.000   1.000
3.577  0.731   1.000
4 valores acima de 1.
```

```
#include<iostream>
#include<iomanip>
#define m 4
#define n 3
using namespace std;
void impvet(float x[m][n]){
  int i,j;
  for (i=0;i<m;i++){
    for (j=0;j<n;j++){
      cout<<setw(6)<<setprecision(4)<<x[i][j]<<' ';
    }
    cout<<endl;
  }
}

void levet(float x[m][n]){
  int i,j;
  for (i=0;i<m;i++){
    for (j=0;j<n;j++){
      cout<<"Informe l= "<<i+1<<" c= "<<j+1<<' ';
      cin>>x[i][j];
    }
  }
}

int main(){
  float en[m][n]={{37,28,41},{17,41,53},{25,93,31},{93,19,26}};
  float sa[m][n];
  int i,j,js,qtd;
  qtd=0;
  levet(en);
  impvet(en);
  for (i=0;i<m;i++){
    for (j=0;j<n;j++){
      if (i>=m-1){
        js=n-1; }
      else{
        js=i;
      }
```

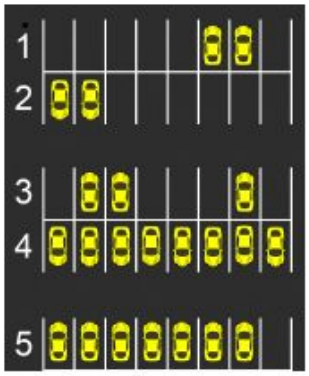


```
      sa[i][j]=en[i][j]/en[i][js];
      if (sa[i][j]>1.000){
        qtd++;}
    }
  }
  impvet(sa);
  cout<<"Quantidade "<<qtd<<" valores acima de 1."<<endl;
```

Questão 2 (50 pontos)

Um estacionamento quer controlar seu movimento de carros e instalou sensores em cada uma de suas vagas.

Os sensores informam, sempre que requisitado, se existe ou não um veículo estacionado na vaga, resultando em uma matriz com valores que representam vaga livre ou vaga ocupada.



Escreva um programa em linguagem C++ que receba uma matriz representando um estacionamento de M corredores, cada corredor com N vagas (M e N estabelecidos via `#define`), cada elemento da matriz assumindo valor 1 (vaga ocupada) ou 0 (vaga livre). O programa deve informar: (i) se existe ou não algum corredor totalmente ocupado (se existir, informar quais), e (ii) quantas vagas do estacionamento estão ocupadas e quantas estão livres.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_estacionamento (int matriz[][N])`. Sua solução deve apenas chamar esta função.

Exemplo de execução (com M=5 e N=8):

```
Informe estacionamento:
0 0 0 0 0 1 1 0
1 1 0 0 0 0 0 0
0 1 1 0 0 0 1 0
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 0
Corredor 4 está lotado.
Existem 22 vagas ocupadas e 18 livres.
```

```
#include<iostream>
#include<iomanip>
#define m 5
#define n 8
using namespace std;
void impvet(float x[m][n]){
  int i,j;
  for (i=0;i<m;i++){
    for (j=0;j<n;j++){
      cout<<setw(6)<<x[i][j]<<' ';
    }
    cout<<endl;
  }
}

void levet(float x[m][n]){
  int i,j;
  for (i=0;i<m;i++){
    for (j=0;j<n;j++){
      cout<<"Informe l= "<<i+1<<" c= "<<j+1<<' ';
```



```
      cin>>x[i][j];
    }
  }
}

int main(){
  int en[m][n]={{0,0,0,0,0,1,1,0},{1,1,0,0,0,0,0,0},
  {0,1,1,0,0,0,1,0},{1,1,1,1,1,1,1,1},
  {1,1,1,1,1,1,1,0}};
  int i,j,livre,qtdvo,qtdvl;
  qtdvo=qtdvl=0;
  for (i=0;i<m;i++){
    livre=0;
    for (j=0;j<n;j++){
      if (en[i][j]==1){
        qtdvo++;
      }
      else {
        qtdvl++;
        livre=1;
      }
    }
    if (livre==0){
      cout<<"Corredor "<<i+1<<" lotado"<<endl;
    }
  }
  cout<<"Existem "<<qtdvo<<" vagas ocupadas e "<<qtdvl<<" vagas livres";
}
```

Questão 2 (50 pontos)

Teobaldo é proprietário de uma concessionária de carros e precisa da sua ajuda para verificar a produtividade de N funcionários das vendas semanais de um mês. Sua tarefa é escrever um programa em C++ que receba do usuário uma matriz de N linhas x 4 colunas (N estabelecido via `#define`) de valores reais, na qual cada linha representa um vendedor e as colunas são os totais (em reais) de vendas feitas por ele em cada semana do mês (considerando-se meses de 4 semanas). Ao final, seu programa deve mostrar na tela: a) o vendedor mais produtivo, b) o total do mês, c) a média dos vendedores.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (float matriz[][4])`. Sua solução deve apenas chamar esta função.

Exemplo de execução (com N=3):

```
Informe as vendas:
30000  25450  12500  10000
11500  13550  21450  12999
40000  35000  15600  19500
Vendedor mais produtivo: 3
Total Mensal: R$ 247549.00
Média dos Vendedores: R$ 82516.33
```

```
#include<iostream>
#include<iomanip>
#define n 3
using namespace std;
void impvet(float x[n][4]){
  int i,j;
  for (i=0;i<n;i++){
    for (j=0;j<4;j++){
      cout<<setw(6)<<setprecision(4)<<x[i][j]<<' ';
    }
    cout<<endl;
  }
}

void levet(float x[n][4]){
  int i,j;
  for (i=0;i<n;i++){
```



```
    for (j=0;j<4;j++){
      cout<<"Informe l= "<<i+1<<" c= "<<j+1<<' ';
      cin>>x[i][j];
    }
  }
}

int main(){
  float en[n][4]={{30000,25450,12500,10000},{11500,13550,21450,12999},
  {40000,35000,15600,19500}};
  float mx=-9999;
  float total,media,som;
  int i,j,qual;
  total=0;
  media=0;
  for (i=0;i<n;i++){
    som=0;
    for(j=0;j<4;j++){
      som=som+en[i][j];
      total=total+en[i][j];
      media=media+en[i][j];
    }
    if (som>mx){
      mx=som;
      qual=i+1;
    }
  }
  cout<<"Vendedor mais produtivo: "<<qual<<endl;
  cout<<"Total mensal: R$ "<<total<<endl;
  media=media/n;
  cout<<"Media dos vendedores: R$ "<<media<<endl;
}
```


Questão 2 (50 pontos)

O município de Blumenau, em Santa Catarina, para proteger os cidadãos de calamidades devidas a enchentes, monitora o nível (em metros) do rio Itajaí-Açu de tempos em tempos, e criou um sistema de alerta à população.

Todos os locais do município suscetíveis a enchentes tiveram suas cotas (em metros), em relação ao fundo do rio, mapeados e registrados. Foram também identificados e catalogados abrigos para os quais a população deve deslocar-se em caso de enchente.

Para cada local do município sujeito a enchentes foram registrados o código do local, sua cota (em metros) em relação ao fundo do rio, o código do abrigo para onde seus moradores devem deslocar-se em caso de enchente e o número de avisos emitidos para aquele local (inicialmente com valor zero e incrementado de 1 a cada processamento do programa com situação de enchente detectada).

Escrever um programa em C++ que leia do usuário o nível atual do rio e uma matriz de monitoramento de 4 linhas e N colunas (N estabelecido via `#define`), cada coluna correspondente aos dados dos locais do município sujeitos a enchentes e cada linha correspondente a código do local, cota, código de abrigo e número de avisos, conforme descrito anteriormente. O programa deve mostrar, para os locais com cotas em situação de enchente, os locais, cotas e abrigos e, ainda, incrementar o número de avisos emitidos para aquele local na matriz.

Exemplo de execução (para N = 9):

```
Nível do rio (m): 10.05
Matriz de monitoramento:
30  31   32   33   34   35  36    37    40
9   9.3  9.5  9.7  9.5  11  12.3  10.9  11
90  90   90   91   91   92  55    55    55
3   2    2    1    2    2   1     0     0

Enchentes:
Locais    Cotas     Abrigos
30        9         90
31        9.3       90
32        9.5       90
33        9.7       91
34        9.5       91

Matriz de monitoramento final:
30  31   32   33   34   35  36    37    40
9   9.3  9.5  9.7  9.5  11  12.3  10.9  11
90  90   90   91   91   92  55    55    55
4   3    3    2    3    2   1     0     0
```

```
#include<iostream>
#include<iomanip>
#define N 9
using namespace std;
int main() {
  float en[4][9]={{30,31,32,33,34,35,37,40},{9,9.3,9.5,9.7,9.5,11,12.3,10.9,11},
  {90,90,90,91,91,92,55,55,55},{3,2,2,1,2,2,1,0,0}};
  float rio;
  int i,j;
  cout<<"Informe o nivel do rio em m ";
  cin>>rio;
  for (i=0;i<N;i++){
    if (en[1][i]<rio){
      cout<<en[0][i]<<" "<<en[1][i]<<" "<<en[2][i]<<endl;
      en[3][i]++;
    }
  }
  cout<<"Matriz de monitoramento final"<<endl;
  for (j=0;j<4;j++){
    for (i=0;i<N;i++){
      cout<<setw(5)<<en[j][i]<<' ';
    }
    cout<<endl;
  }
}
```


3.2 2014 - s2

Questão 1 (50 pontos)

Faça um programa em C++ que permita consultar a classificação de equipes no mundial de construtores de automobilismo. Cada equipe possui dois pilotos: um piloto principal e um piloto secundário. Considere dois vetores para representar esses pilotos: (i) vetor com a pontuação dos pilotos principais de cada equipe e (ii) vetor com a pontuação dos pilotos secundários de cada equipe. Considere também que pilotos da mesma equipe estão em posições opostas em cada vetor. Por exemplo: o primeiro piloto principal e o último piloto secundário são da mesma equipe, o segundo piloto principal e o penúltimo piloto secundário são de outra equipe, e assim por diante. A pontuação dos pilotos não está necessariamente ordenada.

O número N de pares de pilotos que disputam o campeonato é fixo e deve ser informado via `#define`. Seu programa deve obter a pontuação de cada piloto principal, de cada piloto secundário e calcular a pontuação de cada equipe. Guarde a pontuação de sua respectiva equipe no lugar da pontuação original de cada piloto. Mostre como está a classificação das equipes, imprimindo as posições dos dois vetores. Mostre também quem é a equipe que lidera o campeonato.

OBS.: Para a exibição do conteúdo do vetor na tela, considere que já existe a função `void imprime_vetor (int vet[])`. Sua solução deve apenas chamar esta função.

Exemplo de execução (para N=3):

```
Pontos p/ piloto principal: 5 2 4
Pontos p/ piloto secundário: 3 6 9
Pontuacao por equipes:
14 8 7
7 8 14
Equipe 1 na lideranca
```

Outro exemplo de execução (para N=5):

```
Pontos p/ piloto principal: 5 2 4 6 3
Pontos p/ piloto secundario: 1 9 0 10 8
Pontuacao por equipes:
13 12 4 15 4
4 15 4 12 13
Equipe 4 na lideranca
```

```
#include<iostream>
#include<iomanip>
#define N 3
using namespace std;
int lev(int xp[N], int xs[N]){
  int i;
  for (i=0;i<N;i++){
    cout<<"Informe pontuacao piloto principal - ";
    cin>>xp[i];
  }
  for (i=0;i<N;i++){
    cout<<"Informe pontuacao piloto reserva - ";
    cin>>xs[i];
  }
}

int main() {
  int vp[N];
  int vs[N];
  int i,aux,maximo,imaximo;
  lev(vp,vs);
  for (i=0;i<N;i++){
    aux=vp[i]+vs[(N-i)-1];
    vp[i]=aux;
    vs[(N-i)-1]=aux;
  }
  cout<<"Pontuacao por equipes "<<endl;
  for (i=0;i<N;i++){
    cout<<setw(4)<<vp[i]<<' ';
  }
  cout<<endl;
```



```
  for (i=0;i<N;i++){
    cout<<setw(4)<<vs[i]<<' ';
  }
  cout<<endl;
  maximo=-99999;
  for (i=0;i<N;i++){
    if (vp[i]>maximo){
      maximo=vp[i];
      imaximo=i+1;
    }
  }
  cout<<"Equipe "<<imaximo<<" na lideraca"<<endl;
}
```

Questão 2 (50 pontos)

Um comerciante deseja catalogar os M produtos que comercializa em uma matriz de M linhas. Para cada produto, ele deseja registrar como colunas (N) da matriz: o código, a quantidade em estoque, o custo unitário de aquisição e o preço unitário de venda praticado. Escrever um programa em C++ que permita ao comerciante desenvolver o cadastro desejado e, a partir deste, determinar o custo total de seu estoque, o item com maior rentabilidade absoluta (obtida pela diferença entre preço e custo unitários) e sua perspectiva de ganho se vender todo o estoque deste item. Os valores de M e N deverão ser estabelecidos via diretiva *#define*.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (float matriz[][N])`. Sua solução deve apenas CHAMAR esta função.

Exemplo de execução (com M=8, N=4):

```
CATÁLOGO DE PRODUTOS
Cód.     Estoque   Custo   Preço
.00010   35000      .025    .05
.01023   135       7.134    9.5
.23189   3345      2.74     5
.00235   15        115.285  209
.12131   52356      .3598   .85
.00123   854       12.557   13.3
.00020   2600       .472    1
.00025   1000       .125    .50
Custo total do estoque: 43646.2318
Item com maior rentabilidade: .00235
Lucro bruto previsto: 1405.725
```

```
#include<iostream>
#include<iomanip>
#define M 8
#define N 4
using namespace std;
int main() {
  float es[M][N]={{0.00010,35000,0.025,0.05},
  {0.01023,135,7.134,9.5},
  {0.23189,3345,2.74,5},
  {0.00235,15,115.285,209},
  {0.12131,52356,0.3598,0.85},
  {0.00123,854,12.557,13.3},
  {0.00020,2600,0.472,1},
  {0.00025,1000,0.125,0.50}};
  float cte,mra,pg,mai;
  int i,imai;
  cte=0;
  for (i=0;i<M;i++){
    cte=cte+es[i][1]*es[i][2];
  }
  mai=-99999;
  for (i=0;i<M;i++){
    if((es[i][3]-es[i][2])>mai){
      mai=es[i][3]-es[i][2];
      imai=i;
    }
  }
  pg=es[imai][1]*(es[imai][3]-es[imai][2]);
  cout<<"Custo total do estoque "<<cte<<endl;
  cout<<"item com maior lucratividade "<<es[imai][0]<<endl;
  cout<<"Lucro bruto previsto "<<pg<<endl;
```


Questão 1 (50 pontos)

Faça um programa em C++ que leia (do teclado) um vetor de tamanho N (N estabelecido via diretiva `#define`) de números reais e crie um novo vetor, de mesmo tamanho e tipo do original, com as seguintes características: **(a)** valores dos elementos das extremidades do novo vetor (primeiro e último) idênticos aos do original; **(b)** valores dos elementos do "miolo" do vetor (segundo ao penúltimo) obtidos a partir da média dos valores de seu antecessor e sucessor posicionais no vetor original, desde que seu valor esteja situado acima do valor de seu antecessor e abaixo do valor de seu sucessor no vetor original; não verificada esta característica, gerar o elemento do vetor de saída com o mesmo valor do original.

Ao final do processamento, o programa deverá mostrar o vetor gerado no monitor de vídeo e informar quantas substituições de valores foram feitas.

OBS.: Considere que a função `void imprime_vetor(float vet[])` para mostrar o vetor criado JÁ EXISTE. Sua solução deverá apenas CHAMAR esta função onde necessário.

Exemplo de execução (com N=10):

```
Informe vetor:
 3 4.5 2 5 6 7.3 7.5 9 12 15
Vetor gerado:
 3 4.5 2 4 6.15 6.75 8.15 9.75 12 15
Alterações: 6
```

```
#include<iostream>
#include<iomanip>
#define N 10
using namespace std;
int main() {
  float v[N]={3,4.5,2,5,6,7.3,7.5,9,12,15};
  float v[N];
  float vs[N];
  int i,ct;
  ct=0;
  cout<<"Informe vetor "<<endl;
  for (i=0;i<N;i++){
    cin>>v[i];
  }
  vs[0]=v[0];
  vs[N-1]=v[N-1];
  for (i=1;i<N-1;i++){
    if ((v[i]>v[i-1])&&(v[i]<v[i+1])){
      vs[i]=(v[i-1]+v[i+1])/2;
      ct++;
    }
    else{
      vs[i]=v[i];
    }
  }
  for (i=0;i<N;i++){
    cout<<setw(4)<<vs[i]<<' ';
  }
  cout<<endl<<"Alteracoes "<<ct;
}
```

Questão 2 (50 pontos)

Faça um programa em C++ que leia uma matriz $M \times xN$ de numeros inteiros (M e N definidos via diretiva `#define`) e calcula qual é a linha cuja soma é maior que a soma das outras linhas, informando este valor ao usuário. Caso haja mais que uma linha com soma maior que as outras, mostrar apenas a primeira que satisfaz a condição.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Sua solução deve apenas CHAMAR esta função.

Exemplo de execução (com M=4 e N=3):

```
Informe a matriz:
8 2 7
4 5 5
3 10 2
5 10 1
linha 0, soma 17
```

```
#include<iostream>
```



```
#include<iomanip>
#define M 4
#define N 3
using namespace std;
int main() {
  int m[M][N]={{8,2,7},{4,5,5},{3,10,2},{5,10,1}};
  int som,msom,imsom;
  msom=-99999;
  int i,j;
  for (i=0;i<M;i++){
    som=0;
    for(j=0;j<N;j++){
      som=som+m[i][j];
    }
    if (som>msom){
      msom=som;
      imsom=i;
    }
  }
  cout<<"linha "<<imsom<<",
  soma "<<msom<<endl;
}
```

Questão 1 (50 pontos)

Faça um programa em C++ que leia um vetor A de 6 elementos contendo o gabarito da Mega Sena. A seguir, ler um vetor B de 10 elementos contendo uma aposta. Escrever quantos pontos fez o apostador, e se ele fez a sena (6 acertos), a quina (5 acertos) ou a quadra (4 acertos). E no vetor B, mudar o valor do elemento que teve acerto para 99.

OBS.: Considere que a função `void imprime_vetor(int vet[])` para mostrar o vetor criado JÁ EXISTE. Sua solução deverá apenas CHAMAR esta função onde necessário.

Exemplo de execução:

```
Vetor A:  7 43 51 47 6 3
Vetor B:  7 43 9 1 5 8 22 47 6 34
Fez a quadra
 99 99 9 1 5 8 22 99 99 34
```

```
#include<iostream>
#include<iomanip>
#define M 4
#define N 3
using namespace std;
int main(){
  int a[6];
  int b[10];
  int i,j,ac;
  cout<<"Vetor A: ";
  for(i=0;i<6;i++){
    cin>>a[i];
  }
  cout<<"Vetor B: ";
  for(i=0;i<10;i++){
    cin>>b[i];
  }
  ac=0;
  for(i=0;i<6;i++){
    for(j=0;j<10;j++){
      if (a[i]==b[j]){
        ac++;
        b[j]=99;
      }
    }
  }
  if (ac==6){
    cout<<"Fez a sena"<<endl;
  }
  if (ac==5){
    cout<<"Fez a quina"<<endl;
```



```
  }
  if (ac==4){
    cout<<"Fez a quadra"<<endl;
  }
  for(i=0;i<10;i++){
    cout<<b[i]<<' ';
  }
}
```

Questão 2 (50 pontos)

O gabarito de cada participante de uma prova de concurso foi digitalizado e uma matriz de números inteiros correspondente foi gerada. Essa matriz tem uma linha para cada questão e uma coluna para cada alternativa. Assim, cada posição contém o valor 0 se a alternativa não foi marcada pelo participante e 1 caso contrário. Faça um programa em C++ que leia um vetor de tamanho N, onde cada posição i contém um `char` representando a alternativa correta da i-ésima questão (`'A'` se a primeira alternativa é a correta, `'B'` se a segunda, e assim por diante). Em seguida, seu programa deve ler uma matriz de dimensões $N \times M$ representando as repostas de um participante e, por fim, deve imprimir a quantidade de questões que o participante acertou. Considere que questões com 2 ou mais alternativas marcadas estão **incorretas**. Os valores N e M devem ser estabelecidos via `#define`.

Dica: O código ASCII do caractere `'A'` é 65, do `'B'` é 66, e assim por diante.

OBS.: Para a leitura da matriz e do vetor, considere que já existem as funções `void le_matriz(int mat[][M])` e `void le_vetor(int vet[])`. Seu programa deve apenas CHAMAR estas funções.

Exemplo de execução (com N=5, M=4):

```
Entre com o vetor:
A A D B C
Entre com o gabarito:
0 0 1 0
1 0 0 0
1 0 0 1
0 1 0 0
0 0 0 1
O participante acertou 2 questões.
```

```
#include<iostream>
#define N 5 //numero de linhas
#define M 4 // quantidade de alternativas
using namespace std;
int main(){
  int mat[N][M]={{0,0,1,0},{1,0,0,0},{1,0,0,1},{0,1,0,0},{0,0,0,1}};
  char gab[]="AADBC";
  int i,j,som,qual,nota=0;
  for (i=0;i<N;i++){
    som=0;
    for (j=0;j<M;j++){
      som=som+mat[i][j];
    }
    qual=gab[i]-65;
    if ((som==1)&&(mat[i][qual]==1)){
      nota=nota+1;
    }
  }
  cout<<"O participante acertou "<<nota<<" questoes";
}
```


3.3 2015

Questão 1 (50 pontos)

Considere um programa que gerencia uma prateleira de locadora de filmes. A prateleira é representada no programa por um vetor, onde cada posição equivale a um filme e à quantidade de exemplares disponíveis para a locação. Por exemplo, o número 3 na posição 7 indica que existem 3 exemplares disponíveis do filme que está na posição 7. O valor 0 indica que não há exemplar disponível para ser alugado onde quer que ele esteja.

Escreva um programa em C++ que leia N quantidades de exemplares (N estabelecido via `#define`), e que depois de efetuar a leitura da situação atual da prateleira, gerencie a locação e devolução dos filmes da seguinte forma: o usuário deverá digitar `f` para finalizar, `a` para indicar que será uma locação, e `d` para indicar uma devolução. Após selecionar `a` ou `d`, o usuário deverá digitar uma sequência de tuplas representando a posição do filme na prateleira e a quantidade de filmes a serem locados ou devolvidos. O valor `-1` encerra esta leitura. A cada etapa de aluguel ou devolução, o programa deverá mostrar a situação da prateleira até que o usuário digite a opção `f` para finalizar a execução. Se não houver quantidade suficiente para a venda, esta não deve ser feita e a prateleira fica sem mudanças.

OBS.: Considere que a função `void imprime_vetor(int vet[])` para mostrar o conteúdo de um vetor JÁ EXISTE. Sua solução deverá apenas CHAMAR esta função onde necessário.

Exemplo de execução (com N=5):

```
Entre situação da prateleira:
2 4 0 1 2
(a)lugar? (d)evolver ou (f)inalizar? a
0 1
Alugado filme 0.
2 1
Filme 2 sem exemplares.
3 1
Alugado filme 3.
-1 -1
Situação da prateleira: 1 4 0 0 2
(a)lugar? (d)evolver ou (f)inalizar? d
2 2
Devolvido filme 2.
3 1
Devolvido filme 3.
-1 -1
Situação da prateleira: 1 4 2 1 2
(a)lugar? (d)evolver ou (f)inalizar? a
4 4
Exemplares do filme 4 insuficientes.
-1 -1
Situação da prateleira: 1 4 2 1 2
(a)lugar? (d)evolver ou (f)inalizar? f
Fim da execução.
```

```
#include<iostream>
#define N 5
using namespace std;
void le_pra(int p[N]){
  int i;
  for (i=0;i<N;i++){
    cin>>p[i];
  }
}

void imp_pra(int p[N]){
  int i;
  for (i=0;i<N;i++){
    cout<<p[i]<<" ";
  }
  cout<<endl;
}

int main() {
```



```
  int p[N];
  int fil,qtd;
  char op='x';
  cout<<"Entre a situacao da prateleira: "<<endl;
  le_pra(p);
  imp_pra(p);
  while (op != 'f'){
    cout<<"(a) alugar: (d) devolver ou (f) finalizar ";
    cin>>op;
    if (op=='f') {
      cout<<"Fim da execucao"<<endl;
      break;
    }
    if (op=='a'){
      while (1==1){
        cin>>fil>>qtd;
        if (fil==-1){
          break;
        }
        if (p[fil]>=qtd){
          cout<<"Alugado filme "<<fil<<endl;
          p[fil]=p[fil]-qtd;
        }
        else {
          cout<<"Filme "<<fil<<" sem exemplares"<<endl;
        }
      }
    }
    if (op=='d'){
      while (1==1){
        cin>>fil>>qtd;
        if (fil==-1){
          break;
        }
        p[fil]=p[fil]+qtd;
        cout<<"Devolvido filme "<<fil<<endl;
      }
    }
    cout<<"Situacao da prateleira: "<<endl;
    imp_pra(p);
  }
}
```

Questão 2 (50 pontos)

Faça um programa em C++ que leia uma matriz $M \times N$ de números inteiros (M e N estabelecidos via `#define`), calcule e imprima quantos elementos 2-maior existem nessa matriz. Um elemento é 2-maior caso seja maior que os seus vizinhos de linha (da direita e da esquerda). Caso não exista vizinho, considere que este possui o valor 0 (zero). Para resolver o problema proposto pede-se o desenvolvimento e uso de uma função de nome `calcula_2maior()` que recebe como entrada uma matriz, calcula e retorna o número de elementos 2-maior.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Sua solução deve apenas CHAMAR esta função.

Exemplo de execução (com M=3, N=4):

```
Informe a matriz
7   15   4    11
5   3    20    0
1   6    9     8
Qtde. elementos 2-maior: 5
```

```
#include<iostream>
#define M 3
#define N 4
using namespace std;
int calcula_2maior(int mat[M][N]){
  int i,j,ve,vd,qtd=0;
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
```



```
      if (j==0){
        ve=0;
      }
      else{
        ve=mat[i][j-1];
      }
      if (j==N){
        vd=0;
      }
      else {
        vd=mat[i][j+1];
      }
      if ((mat[i][j]>ve)&&(mat[i][j]>vd)){
        qtd++;
      }
    }
  }
  return qtd;
}

int main() {
  int mat[M][N]={{7,15,4,11},{5,3,20,0},{1,6,9,8}};
  int xx;
  xx=calcula_2maior(mat);
  cout<<"Qtde de elementos 2-maior: "<<xx<<endl;
}
```

Questão 1 (50 pontos)

Escreva um programa em C++ para ler um vetor de N elementos inteiros (N estabelecido via `#define`). Em seguida, pergunte ao usuário qual posição deve ser eliminada. A partir dessa posição o vetor terá seus elementos deslocados uma posição à esquerda (na direção do inicio do vetor). A posição vaga no final do vetor após o deslocamento devem receber valor 0 (zero). Ao final, o programa deve imprimir o valor do elemento excluído e o conteúdo do vetor resultante.

OBS.: Considere que a função `void imprime_vetor(int vet[])` para mostrar conteúdo de um vetor JÁ EXISTE. Sua solução deverá apenas CHAMAR esta função onde necessário.

Exemplo de execução (com N=10):

```
Informe vetor:
 5 6 8 2 1 3 9 4 7 10
Indique posicao a eliminar: 4
Vetor final: 5 6 8 2 3 9 4 7 10 0
Valor eliminado: 1
```

```
#include<iostream>
#define N 10
using namespace std;
int imprime_vetor(int vet[N]){
  int i;
  for (i=0;i<N;i++){
    cout<<vet[i]<<" ";
  }
  cout<<endl;
}

int main() {
  int vet[N];
  int i,pos,qual;
  cout<<"Informe vetor: "<<endl;
  for (i=0;i<N;i++){
    cin>>vet[i];
  }
  cout<<"Indique posicao a eliminar: ";
  cin>>pos;
  qual=vet[pos];
  for (i=pos;i<N-1;i++){
    vet[i]=vet[i+1];
  }
  vet[N-1]=0; //99 para testar
```



```
  cout<<"Vetor final: ";
  imprime_vetor(vet);
  cout<<"Valor eliminado: "<<qual;
}
```

Questão 2 (50 pontos)

Escrever um programa em C++ que leia uma matriz de número reais de ordem $M \times N$ (M e N estabelecidos via *#define*, sendo $N > 2$), verifique e mostre as linhas nas quais os dois primeiros valores da linha e qualquer dos valores restantes da mesma linha podem formar um triângulo. Indicar os casos de linhas que não permitam a formação de triângulo algum. Para a fomação de um triângulo, a soma de dois lados quaisquer deve ser maior que o terceiro.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (float matriz[][N])`. Sua solução deve apenas CHAMAR esta função.

Exemplo de execução (com M=5, N=4):

```
Indique a matriz:
3 4 5 8
0 0 0 0
1 2 3 4
4 3 2 3
1 1 1 3
Resultado:
Linha 0 colunas 0 e 1 com 2
Linha 1 não permite formar triângulos
Linha 2 não permite formar triângulos
Linha 3 colunas 0 e 1 com 2 e 3
Linha 4 colunas 0 e 1 com 2
```

```
#include<iostream>
#define M 5
#define N 4
using namespace std;
int triangulo(float a, float b, float c){
  if ((a<b+c)&&(b<a+c)&&(c<a+b)){ return 1; }
  else { return 0; }
}

int main() {
  float mat[M][N]={{3,4,5,8},{0,0,0,0},{1,2,3,4},{4,3,2,3},{1,1,1,3}};
  int sentinela[N];
  int i,j,qual;
  for (i=0;i<M;i++){
    qual=0;
    for(j=2;j<N;j++){
      if (triangulo(mat[i][0], mat[i][1], mat[i][j])){
        sentinela[qual]=j;
        qual++;
      }
    }
    if (qual==0){
      cout<<"Linha "<<i<<" nao permite formar triangulos"<<endl;
    }
    else{
      cout<<"Linha "<<i<<" colunas 0 e 1 com ";
      for (j=0;j<qual;j++){
        cout<<sentinela[j]<<" ";
      }
      cout<<endl;
    }
  }
}
```


Questão 1 (50 pontos)

Uma companhia de eletricidade implantou uma política que reajusta a conta de energia elétrica daqueles que gastam além de um determinado valor limite. O reajuste varia de acordo com uma das seguintes situações: (i) reajuste de 10% para quem gastou até 40% além do valor limite; (ii) reajuste de 25% para quem gastou de acima de 40% e até 70% do valor limite; (iii) reajuste de 50% para quem gastou acima de 70% do valor limite.

Faça um programa em C++ que obtenha um valor limite para cálculos de reajustes, e um vetor de N valores reais (N estabelecido via `#define`), cada valor representando o valor gasto por cada uma das N residências avaliadas. Atualize o valor das contas de cada residência no vetor caso elas ultrapassem o valor limite. Por fim, mostre os valores reajustados que cada residência deve pagar para saldar sua conta de energia e o total acumulado que será arrecadado a mais pela companhia elétrica.

OBS.: Considere que a função `void imprime_vetor(float vet[])` para mostrar o conteúdo de um vetor JÁ EXISTE. Sua solução deverá apenas CHAMAR esta função onde necessário.

Exemplo de execução (com N=10):

```
Digite valor limite: 100
Digite contas de residencias:
20 40 60 80 100 120 140 160 180 200
Valores antes do reajuste:
20 40 60 80 100 120 140 160 180 200
Valores depois do reajuste:
20 40 60 80 100 132 154 200 270 300
Valor extra arrecadado: 256
```

Exemplo de execução (com N=5):

```
Digite valor limite: 300
Digite contas de residencias:
100 200 300 400 500
Valores antes do reajuste:
100 200 300 400 500
Valores depois do reajuste:
100 200 300 440 625
Valor extra arrecadado: 165
```

```
#include<iostream>
#define N 10
using namespace std;
int main() {
  float res[N]={20,40,60,80,100,120,140,160,180,200};
  float lim,extr=0,nv;
  int i;
  cout<<"Digite valor limite: "<<endl;
  cin>>lim;
  cout<<"Valores antes do reajuste: "<<endl;
  for (i=0;i<N;i++){
    cout<<res[i]<<" ";
  }
  cout<<endl;
  for (i=0;i<N;i++){
    nv=res[i];
    if ((res[i]>lim)&&(res[i]<=lim*1.4)) {nv=res[i]*1.1;}
    if ((res[i]>lim*1.4)&&(res[i]<=lim*1.7)){ nv=res[i]*1.25;}
    if (res[i]>lim*1.7) {nv=res[i]*1.5;}
    extr=extr+nv-res[i];
    res[i]=nv;
  }
  cout<<"Valores depois do reajuste: "<<endl;
  for (i=0;i<N;i++){
    cout<<res[i]<<" ";
  }
  cout<<endl;
  cout<<"Valor extra arredacadado "<<extr;
}
```


Questão 2 (50 pontos)

Faça um programa em C++ que leia uma matriz $M \times N$ de números inteiros (M e N estabelecidos via `#define`) e: imprime **Borda** se a soma dos elementos da borda da matriz é maior que soma do elementos do miolo da matriz; ou imprime **Miolo** caso contrário. A borda da matriz são todos os elementos das primeiras e últimas colunas e linhas; o miolo são todos os outros elementos da matriz. Você deve escrever a função `MioloMoleCascaDura()`, que recebe uma matriz $M \times N$ e retorna 1 (um) se a soma dos elementos da borda da matriz é maior ou igual do que a soma do elementos do miolo e 0 (zero) caso contrário.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Sua solução deve apenas CHAMAR esta função.

Exemplo de execução (com M=3, N=4):

```
Informe matriz:
1  4 2 5
23 5 4 9
9  3 7 5
Borda
```

Outro exemplo de execução (com M=3, N=4):

```
Informe matriz:
1  4  1 5
1 25 14 2
2  3  7 5
Miolo
```

```
#include<iostream>
#define M 3
#define N 4
using namespace std;
int miolomolecascadura(int mat[M][N]){
  int i,j,den=0,fora=0;
  for (i=0;i<M;i++){
    for(j=0;j<N;j++){
      if ((i==0)||(i==M-1)||(j==0)||(j==N-1)){
        fora=fora+mat[i][j];
      }
      else {
        den=den+mat[i][j];
      }
    }
  }
  if (fora>=den) {return 1;} else {return 0;}
}

int main() {
  int mat[M][N]={{1,4,2,5},{23,5,4,9},{9,3,7,5}};
  int mat[M][N]={{1,4,1,5},{1,25,14,2},{2,3,7,5}};
  int res;
  res=miolomolecascadura(mat);
  if (res==1){
    cout<<"Borda"<<endl;
  }
  else {
    cout<<"Miolo"<<endl;
  }
}
```


Questão 1 (50 pontos)

Faça um programa em C++ que leia um vetor de tamanho N (N estabelecido via `#define`) contendo inteiros $x_1, x_2, \ldots, x_N$ representando o resultado da rodada de um campeonato de futebol com N times. Cada valor x_k pode ser: (a) **negativo** (ou nulo), representando a diferença de gols pela qual o time k perdeu (ou empatou) o jogo dessa rodada; (b) **positivo**, representando o índice (de 1 a N) do time que foi adversário do time k nessa rodada. Por exemplo, para sequência -2 1 4 0, temos que: o time 1 perdeu o jogo por uma diferença de 2 gols, o time 2 jogou contra o time 1, o time 3 jogou contra o time 4 e o time 4 empatou a partida. Seu programa deve alterar o vetor lido de forma que toda posição passe a ter o saldo de gols do respectivo time nessa rodada. Para o exemplo, o 2º elemento deve ser alterado para 2 (pois o time 2 jogou contra o time 1 que tem um saldo de -2 gols) e o 3º elemento deve ser alterado para 0 (pois o time 3 jogou contra o time 4 que tem um saldo de 0 gols). Ao final, seu programa deve imprimir o vetor resultante, a maior diferença de gols que houve nas partidas da rodada e a quantidade de partidas que terminaram empatadas. Considere que todo time perdedor apresenta no vetor o seu saldo de gols e todo time ganhador apresenta o índice do seu adversário, variando de 1 a N. No caso de empate, apenas um deles tem o índice do adversário.

```
-2 0 2 1 -1 0
Maior diferença de gols: 2
Qtd. de jogos empatados: 1
```

Outro exemplo de execução (com N=10):

```
Resultado da rodada:
0 0 -3 9 1 -3 6 2 -1 3
Saldo de gols:
0 0 -3 1 0 -3 3 0 -1 3
Maior diferença de gols: 3
Qtd. de jogos empatados: 2
```

OBS.: Considere que a função `void imprime_vetor(int vet[])` para mostrar o conteúdo de um vetor JÁ EXISTE. Sua solução deverá apenas CHAMAR esta função onde necessário.

Exemplo de execução (com N=6):

```
Resultado da rodada:
-2 6 1 5 -1 0
Saldo de gols:
```

```
#include<iostream>
#include<cmath>
#define N 10
using namespace std;
int main(){
  int vet[N]={-2, 6, 1, 5, -1, 0};
  // time 1 perdeu de 2 gols, time 2=jogou com 6, time 3 jogou com 1,
  // 4 jogou com 5, 5 perdeu de 1 e 6 empatou a 0
  int vet[N]={0,0,-3,9,1,-3,6,2,-1,3};
  int mdif=-99999, qemp=0, onde;
  int i;
  cout<<"Resultado da rodada: "<<endl;
  for (i=0;i<N;i++){
    cout<<vet[i]<<" ";
  }
  cout<<endl<<"--------------------"<<endl;
  for (i=0;i<N;i++){
    if ((vet[i]<0)&&(abs(vet[i])>mdif)){mdif=abs(vet[i]);}
    if (vet[i]>0){
      onde = vet[i]-1;
      vet[i]=abs(vet[onde]);
    }
    if (vet[i]==0){qemp++;}
  }
  cout<<"Saldo de gols: "<<endl;
  for (i=0;i<N;i++){
```



```
    cout<<vet[i]<<" ";
  }
  cout<<endl<<"Maior diferenca de gols: "<<mdif<<endl;
  cout<<"Quantidade de empates: "<<qemp/2<<endl;
}
```

Questão 2 (50 pontos)

Faça um programa em C++ que leia uma matriz quadrada $M \times M$ de números inteiros (M estabelecido via `#define`), calcule e imprima a soma dos menores elementos de cada linha e de cada coluna da matriz. Crie e utilize no programa a função `smenorColLin()`, que recebe uma matriz $M \times M$ e um valor k e retorna a soma do menor elemento da linha k com o menor elemento da coluna k da matriz.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][M])`. Sua solução deve apenas CHAMAR esta função.

Exemplo de execução (com M=4):

```
Informe matriz:
1  4  2  5
23 5  4  9
9  3  7  5
6  5  4  2
Soma dos menores: 18
```

```
#include<iostream>
#define M 4
using namespace std;
int smenorColLin(int mat[M][M], int k){
  int i,menorl,menorc;
  menorl=menorc=999999;
  for (i=0;i<M;i++){
    if (mat[i][k]<menorc){
      menorc=mat[i][k];
    }
    if (mat[k][i]<menorl){
      menorl=mat[k][i];
    }
  }
  return menorc+menorl;
}

int main() {
  int mat[M][M]={{1,4,2,5},{23,5,4,9},{9,3,7,5},{6,5,4,2}};
  int j,soma=0;
  for (j=0;j<M;j++){
    soma=soma+smenorColLin(mat,j);
  }
  cout<<"Soma dos menores: "<<soma<<endl;
}
```

Questão 1 (50 pontos)

Faça um programa em C++ que leia (do teclado) um vetor de tamanho N (N estabelecido via `#define`) de números inteiros e altere este vetor, de forma que todos os elementos divisíveis pelo primeiro valor do vetor recebam o novo valor -1. Ao final do processamento, o programa deverá mostrar o novo conteúdo do vetor no monitor de vídeo e informar a soma dos valores que foram substituídos por -1.

OBS.: Considere que a função `void imprime_vetor(int vet[])` para mostrar o vetor criado JÁ EXISTE. Sua solução deverá apenas CHAMAR esta função onde necessário.

Exemplo de execução (com N=10):

```
Informe vetor:
 3 4 2 5 6 3 7 9 12 15
Vetor gerado:
 -1 4 2 5 -1 -1 7 -1 -1 -1
Soma: 48
```

```
#include<iostream>
#define N 10
using namespace std;
int main() {
  int vet[N]={3,4,2,5,6,3,7,9,12,15};
  int j,soma=0,divisor;
  divisor=vet[0];
  for (j=0;j<N;j++){
    if (vet[j]%divisor==0){
      soma=soma+vet[j];
```



```
      vet[j]=-1;
    }
  }
  cout<<"Vetor gerado: "<<endl;
  for (j=0;j<N;j++){
    cout<<vet[j]<<" ";
  }
  cout<<endl<<"Soma: "<<soma<<endl;
}
```

Questão 2 (50 pontos)

Fazer programa em C++ que lê uma matriz $M \times N$ (M e N estabelecidos via diretiva *#define*) a partir do teclado, e informa se a matriz lida atende o Modelo de Vandermonde Totalmente, Parcialmente, ou não Atende. A avaliação deve ser feita por uma função chamada `Avalia_Vandermonde` (que deverá ser codificada junto com o resto do código) que recebe a matriz e retorna o valor 1 (indicando que atende Totalmente), o valor 0 (indicando que atende Parcialmente) ou o valor -1 (indicando que não Atende Vandermode).

Uma matriz atende totalmente o Modelo de Vandermonde quando M for igual a N e cada item da matriz está dentro do Modelo abaixo. Uma matriz atende parcialmente o modelo de Vandermonde quando está dentro do modelo abaixo, mas $M \neq N$. E a matriz não atende o modelo de Vandermonde quando não atende o modelo abaixo com quaisquer dos valores da matriz.

No modelo abaixo, $X_0, X_1, \cdots X_{M-1}$ são os valores básicos da matriz e podem possuir valores repetidos (por exemplo, X_0 pode ser igual a X_6).

$$\begin{vmatrix} 1 & {X_0}^1 & {X_0}^2 & {X_0}^3 & {X_0}^4 & \cdots & {X_0}^{N-1} \\ 1 & {X_1}^1 & {X_1}^2 & {X_1}^3 & {X_1}^4 & \cdots & {X_1}^{N-1} \\ 1 & {X_2}^1 & {X_2}^2 & {X_2}^3 & {X_2}^4 & \cdots & {X_2}^{N-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & {X_{M-1}}^1 & {X_{M-1}}^2 & {X_{M-1}}^3 & {X_{M-1}}^4 & \cdots & {X_{M-1}}^{N-1} \end{vmatrix}$$

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Sua solução deve apenas CHAMAR esta função.

Exemplo de execução (com M=3, N=3):

```
Digite matriz 3x3
3  4  5
2  3  6
7  1  2
Matriz não atende Vandermonde
```

Outro exemplo de execução (com M=3, N=3):

```
Digite matriz 3x3
1  4  16
1  3  9
1  2  4
Matriz atende Vandermonde totalmente
```

Outro exemplo de execução (com M=2, N=3):

```
Digite matriz 2x3
1  4  16
1  7  49
Matriz atende Vandermonde parcialmente
```

```
#include<iostream>
#define M 2 // numero de linhas
#define N 3 // numero de colunas
#include<cmath>
using namespace std;
int avalia_vandermonde(float mat[M][N]){
  int simnao=1,j,k; // comeca com sim}
  for (j=0;j<M;j++){
    for (k=1;k<N;k++){
      if (mat[j][k]!=pow(mat[j][1],k)){simnao=0;}
    }
  }
  if (simnao==0) {return -1;}
  if ((simnao==1)&&(j!=k)){return 0;}
  if ((simnao==1)&&(j==k)){return 1;}
}
```



```
int main(){
  // float mat[M][N]={{3,4,5},{2,3,6},{7,1,2} }; nao atende
  // float mat[M][N]={{1,4,16},{1,3,9},{1,2,4}}; atende totalmente
  float mat[M][N]={{1,4,16},{1,7,49}}; // atende parcialmente
  int z;
  z=avalia_vandermonde(mat);
  if (z==-1){cout<<"Matriz nao atende vandermonde";}
  if (z==0){cout<<"Matriz atende vandermonde parcialmente";}
  if (z==1){cout<<"Matriz atende vandermonde totalmente";}
}
```

3.4 2016

Questão 1 (50 pontos)

Crie um programa que lê N valores inteiros do teclado e os armazena em um vetor $v1$, e depois lê outros K valores inteiros, que são armazenados em outro vetor $v2$. O programa deve então ler um inteiro indicando uma posição, e inserir o conteúdo de $v2$ em $v1$ a partir dessa posição. O conteúdo de $v1$ não deve ser apagado, mas sim deslocado para a direita a partir da posição definida pelo usuário (veja os exemplos). Os tamanhos dos vetores devem ser fixados via diretivas `#define`. Não esqueça de realizar as validações necessárias. No final, o programa deve imprimir $v1$ com os valores de $v2$ devidamente inseridos, e também o valor do elemento que está na última posição de $v1$.

OBS.: Considere que a função `void imprime_vetor(int vet[], int tamanho)` para mostrar o conteúdo de um vetor JÁ EXISTE. Sua solução deverá apenas CHAMAR esta função onde necessário.

```
Digite vetor 2: 50  51  52
Digite posicao: 8
Posicao invalida
```

Exemplo de execução (com N = 7 e K = 3):

```
Digite vetor 1: 0  1  2  3  4  5  6
Digite vetor 2: 50  51  52
Digite posicao: 2
Resultado insercao:
0  1  50  51  52  2  3  4  5  6
Ultimo elemento do vetor: 6
```

Exemplo de execução (com N = 7 e K = 3):

```
Digite vetor 1: 0  1  2  3  4  5  6
Digite vetor 2: 50  51  52
Digite posicao: 7
Resultado insercao:
0  1  2  3  4  5  6  50  51  52
Ultimo elemento do vetor: 52
```

Exemplo de execução (com N = 7 e K = 3):

```
Digite vetor 1: 0  1  2  3  4  5  6
```

```
#include<iostream>
#define N 7
#define K 3
using namespace std;
int main(){
  int vet1[N];
  int vet2[K];
  int vet3[N+K];
  int i,j,k,pos;
  cout<<"Informe o vetor 1: "<<endl;
  for (i=0;i<N;i++){
    cin>>vet1[i];
  }
  cout<<"Informe o vetor 2: "<<endl;
  for (i=0;i<K;i++){
    cin>>vet2[i];
  }
  cout<<"Informe a posição: ";
  cin>>pos;
  if (pos>N) {
    cout<<"Posicao invalida";
    return 0;
  }
  i=0;
  j=0;
  while (i<pos){
    vet3[i]=vet1[j];
    i++;
```



```
    j++;
  }
  k=0;
  while (k<K){
    vet3[i]=vet2[k];
    i++;
    k++;
  }
  while (i<N+K){
    vet3[i]=vet1[j];
    i++;
    j++;
  }
  cout<<"Resultado da insercao: "<<endl;
  for (i=0;i<N+K;i++){
    cout<<vet3[i]<<" ";
  }
  cout<<"Ultimo elemento do vetor e: "<<vet3[N+K-1];
}
```

Questão 2 (50 pontos)

Escreva um programa em linguagem C++ que deverá ler uma matriz $M \times N$ (M e N estabelecidos via *#define*) de valores inteiros. Em seguida o programa deverá encontrar e imprimir qual é o maior e o menor elemento da matriz. Em seguida o programa deverá aplicar a seguinte regra para cada elemento da matriz: (a) se o elemento for positivo subtraia o maior valor; (b) caso o elemento seja negativo adicione o menor valor; (c) caso seja 0 não faça nada. Depois imprima a matriz resultante.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Para a impressão de uma matriz, considere que já existe a função `void imprime_matriz (int matriz[][N])`. Sua solução deve apenas **CHAMAR** estas funções onde for apropriado.

Exemplo de execução (com M=3, N=4):

```
Digite a matriz inicial:
10   1  -5  0
-10  5  -2  0
1    0  -4  5
Menor valor: -10
Maior valor: 10
Nova matriz:
0    -9  -15  0
-20  -5  -12  0
-9    0  -14 -5
```

```
#include<iostream>
#include<iomanip>
#define M 3
#define N 4
using namespace std;
int main(){
  int mat[M][N]={{10,1,-5,0},{-10,5,-2,0},{1,0,-4,5}};
  int maior=-9999999,menor=99999999,i,j,sentinela;
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      if (mat[i][j]>maior) {maior=mat[i][j];}
      if (mat[i][j]<menor) {menor=mat[i][j];}
    }
  }
  cout<<"Menor valor: "<<menor<<endl;
  cout<<"Maior valor: "<<maior<<endl;
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      sentinela=mat[i][j];
      if (sentinela>0) {
        mat[i][j]=mat[i][j]-maior;
      }
      if (sentinela<0) {
        mat[i][j]=mat[i][j]+menor;
      }
    }
```



```
  }
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      cout<<setw(4)<<mat[i][j];
    }
    cout<<endl;
  }
}
```

Questão 1 (50 pontos)

Faça um programa que leia um vetor V de inteiros com um número fixo N de elementos indicado via `#define`, e rotacione seus elementos para a direita, fazendo com que o valor de V_{i+1} receba o valor de V_i . Cuidado com as extremidades, o primeiro elemento deve receber o último elemento do vetor. Além disso, o programa deve imprimir o somatório de todos os elementos rotacionados.

OBS.: Considere que a função `void imprime_vetor(int vet[])` para mostrar o conteúdo de um vetor JÁ EXISTE. Sua solução deverá apenas CHAMAR esta função onde necessário.

Exemplo de execução (com N=6):

```
Entre vetor:
7  2  3  -1  9  0
Vetor rotacionado:
0  7  2  3  -1  9
Soma dos elementos: 20
```

```
#include<iostream>
#define N 6
using namespace std;
int main(){
  int vet[N]={7,2,3,-1,9,0};
  int i,salva,soma=0;
  salva=vet[N-1];
  for (i=N-1;i>=0;i--){
    vet[i+1]=vet[i];
  }
  vet[0]=salva;
  cout<<"Vetor rotacionado: "<<endl;
  for (i=0;i<N;i++){
    cout<<vet[i]<<" ";
    soma=soma+vet[i];
  }
  cout<<endl<<"Soma dos elementos :"<<soma;
}
```

Questão 2 (50 pontos)

Crie uma função chamada `borrarImagem()`, que recebe uma matriz de inteiros de tamanho $M \times N$ representando uma imagem, e armazena uma versão borrada da imagem original em uma segunda matriz, também de $M \times N$ (utilize `#define` para definir M e N). A versão borrada de um elemento a_{ij} da matriz é dado por $(a_{ij-1} + a_{ij} + a_{ij+1})/3$, onde a é um elemento da matriz original, i indica a linha, e j a coluna. Em outras palavras, estamos tirando a média do elemento com seu vizinho a esquerda e seu vizinho a direita. Tome cuidado com as bordas, sendo que na borda esquerda não há vizinho a esquerda e portanto calculamos $(a_{ij} + a_{ij+1})/2$, e na borda direita não há vizinho a direita, e portanto calculamos $(a_{ij-1} + a_{ij})/2$ como a versão borrada do elemento. A função `borrarImagem()` também deve retornar o valor médio dos elementos da matriz borrada. A função `borrarImagem()` não deve ler dados do teclado nem imprimir informações na tela. Utilize no programa principal a função que você criou, sendo que você deve imprimir a matriz resultante (borrada) na tela. Note que todos os cálculos são feitos com inteiros, portanto as partes fracionárias são descartadas no exemplo.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void lerMatriz (int matriz[][N])`. Para imprimir os valores da matriz na tela, considere que já existe a função `void imprimirMatriz (int matriz[][N])`. Sua solução deve apenas CHAMAR estas funções onde apropriado.

Exemplo de execução (com M=4 e N=5:

```
Digite a matriz da imagem:
103  198  105  115  81
255  74   236  41   205
186  171  242  251  227
70   124  194  84   248
Imagem borrada:
150  135  139  100  98
164  188  117  160  123
178  199  221  240  239
97   129  134  175  166
Média elementos da matriz: 157
```

```
#include<iostream>
#include<iomanip>
#define M 4
#define N 5
using namespace std;
int borrarimagem(int mo[M][N], int mb[M][N]){
  int i,j,soma;
  for (i=0;i<M;i++){
    mb[i][0]=(mo[i][0]+mo[i][1])/2;
    mb[i][N-1]=(mo[i][N-1]+mo[i][N-2])/2;
  }
  for (i=0;i<M;i++){
    for (j=1;j<N-1;j++){
      mb[i][j]=(mo[i][j-1]+mo[i][j]+mo[i][j+1])/3;
    }
  }
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      soma=soma+mb[i][j];
    }
  }
  return soma/(M*N);
}

int main(){
  int mo[M][N]={{103,198,105,115,81},{255,74, 236, 41, 205},
  {186, 171, 242, 251, 227},{70, 124, 194, 84, 248}};
  int mb[M][N];
  int med,i,j;
  med=borrarimagem(mo,mb);
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      cout<<setw(4)<<mb[i][j];
    }
    cout<<endl;
  }
  cout<<"Media elementos da matriz: "<<med;
}
```


4 Final

4.1 2014 - s1

Questão 1 (50 pontos)

Escreva um programa em C++ que deverá ler do teclado um conjunto de valores reais, onde os dois primeiros valores representam a largura e comprimento (em metros) de uma plataforma de terra existente em um supermercado, e os valores restantes (mínimo de 5 valores) representam uma cota (em centímetros) correspondente aos metros lineares desta plataforma. O final do conjunto de valores é indicado ao se digitar uma cota de valor nulo ou negativo (que NÃO DEVE SER CONSIDERADO nos calculos solicitados). O proprietário quer fazer o asfaltamento dessa plataforma, aplainando-a através do acréscimo ou retirada de terra do terreno existente, de forma que a plataforma tenha ao final o valor da primeira cota informada originalmente. O programa deve imprimir o volume total (em m^3) de terra retirada ou adicionada, conforme exemplos abaixo.

Exemplo de execução:

```
Largura e compr. da plataforma:
10 50
Digite as cotas da plataforma:
94 81 121 94 92 0
Volume de terra retirada: 60 m3
```

Outro Exemplo de execução:

```
Largura e compr. da plataforma:
10 50
Digite as cotas da plataforma:
120 94 121 92 81 0
Volume de terra adicionada: 460 m3
```

```
// falta fazer
```

Questão 1 (50 pontos)

Escreva um programa em C++ que deverá ler do teclado um conjunto de valores reais positivos, até que seja digitado um valor negativo.

Devem ser determinados e impressos na tela o valor médio destes valores e o quadrado da diferença entre a soma dos valores e o valor médio calculado. Nestes cálculos, o valor negativo digitado NÃO DEVE SER USADO.

Exemplo de execução:

```
Digite valores reais:
81 94 121 92 94 -10
Valor médio: 96.4
Quadrado da diferença: 148687.36
```

```
#include<iostream>
#include<cmath>
using namespace std;
int main(){
  float val, som=0, qua, med;
  int ctd=0;
  cout<<"Digite valores reais: "<<endl;
  while (1==1){
    cin>>val;
    if (val<0){
      break;
    }
    som=som+val;
    ctd++;
  }
  med=som/ctd;
  qua=pow((som-med),2);
  cout<<"Valor medio: "<<med<<endl;
  cout<<"Quadrado das diferencas: "<<qua<<endl;
}
```


Questão 1 (50 pontos)

Teobaldo adora ver televisão. No entanto, sempre que ele muda de canal o volume do som deste novo canal fica diferente do anterior, às vezes com volume menor, outras vezes com volume maior. Então ele sempre precisa ajustar o volume pressionando uma quantidade de vezes seguidas os botões de aumentar e diminuir o volume. Além disso, a TV de Teobaldo possui um volume mínimo, com valor 0, e volume máximo, com valor 100; a TV nunca ultrapassa os volumes máximo e mínimo.

Depois de tantas mudanças de volume, Teobaldo não sabe qual é o volume atual da TV.

Escreva um programa em C++ que receba o volume inicial da TV, a quantidade de mudanças feitas e a quantidade de vezes que o botão foi apertado em cada mudança; e, ao final, mostre o volume atual da TV.

Exemplo de execução:

```
Volume inicial: 50
Quantidade de mudanças: 4
Valores das mudanças: 11 20 -15 -13
Volume atual: 53
```

Outro exemplo de execução:

```
Volume inicial: 50
Quantidade de mudanças: 5
Valores das mudanças: 30 30 30 40 -78
Volume atual: 22
```

```
#include<iostream>
#include<cmath>
using namespace std;
int main(){
  int vi, qv, nv;
  cout<<"Volume inicial: ";
  cin>>vi;
  cout<<"Quantidade de mudancas: ";
  cin>>qv;
  cout<<"Valores das mudancas: ";
  while (qv>0){
    cin>>nv;
    vi=vi+nv;
    if (vi>100){
      vi=100;
    }
    if (vi<0){
      vi=0;
    }
    qv--;
    cout<<"V="<<vi<<endl;
  }
  cout<<"Volume atual: "<<vi;
}
```


Questão 1 (50 pontos)

Em época de eleição, sua tarefa é escrever um programa em C++ para simular uma urna eletrônica. Na eleição para presidente, existem 3 candidatos que utilizam respectivamente os códigos: 10, 11, 12. Adicionalmente é utilizado o código 1 para voto nulo e 2 para voto em branco. Seu programa deve receber votos até que o valor 0 seja informado. Ao final, seu programa deve mostrar na tela o total de votos de cada candidato, nulos e em branco e o total de votos na eleição.

Exemplo de execução:

```
Informe seu voto: 10
Informe seu voto: 11
Informe seu voto: 11
Informe seu voto: 12
Informe seu voto: 11
Informe seu voto: 10
Informe seu voto: 1
Informe seu voto: 2
Informe seu voto: 2
Informe seu voto: 0
Candidato 1: 2 votos
Candidato 2: 3 votos
Candidato 3: 1 voto
Votos Nulos: 1
Votos em branco: 2
Total de votos: 9
```

```
#include<iostream>
#include<cmath>
using namespace std;
int main(){
  int vot,c10=0, c11=0, c12=0, bra=0, nul=0, tot=0;
  while (1==1){
    cout<<"Informe seu voto: ";
    cin>>vot;
    if (vot==0){
      break;
    }
    if (vot==10){c10++; tot++;}
    if (vot==11){c11++; tot++;}
    if (vot==12){c12++; tot++;}
    if (vot==1){nul++; tot++;}
    if (vot==2){bra++; tot++;}
  }
  cout<<"Candidato 1: "<<c10<<" votos"<<endl;
  cout<<"Candidato 2: "<<c11<<" votos"<<endl;
  cout<<"Candidato 3: "<<c12<<" votos"<<endl;
  cout<<"Votos nulos: "<<nul<<endl;
  cout<<"Votos em branco: "<<bra<<endl;
  cout<<"Total de votos: "<<tot<<endl;
}
```


Questão 2 (50 pontos)

Escreva um programa em C++ que faz a leitura de uma matriz quadrada de números inteiros de tamanho ímpar $N \times N$ (N estabelecido via `#define`). O programa deve alterar para valor 0 todas as posições cujo valor original seja maior que o VMLA (valor médio da linha anterior). Considere $vmla = 5$ para alterar os valores da primeira linha da matriz, pois ela não tem linha anterior. Ao final, o programa deve exibir na tela o novo conteúdo da matriz.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Sua solução deve apenas chamar esta função.

Exemplo de execução (com N=3):

```
Informe matriz:
2 8 3
6 1 4
4 7 8
Matriz resultante:
2 0 3
0 1 4
0 0 0
```

Outro exemplo de execução (com N=5):

```
Informe matriz:
1 2 8 3 9
0 6 1 4 7
9 4 7 8 2
2 7 1 3 8
1 8 2 5 9
Matriz resultante:
1 2 0 3 0
0 0 1 4 0
0 0 0 0 2
2 0 1 3 0
1 0 2 0 0
```

```
#include<iostream>
#define N 5
using namespace std;
int main() {
  int m[N][N];
  int i,j,som;
  float vmla=5;
  cout<<"Informe a matriz N";
  for (i=0;i<N;i++){
    for (j=0;j<N;j++){
      cin>>m[i][j];
    }
  }
  for (i=0;i<N;i++){
    som=0;
    for (j=0;j<N;j++){
      som=som+m[i][j];
      if (m[i][j]>vmla){
        m[i][j]=0;
      }
    }
    vmla=som/N;
  }
  for (i=0;i<N;i++){
    for (j=0;j<N;j++){
      cout<<m[i][j]<<' ';
    }
    cout<<endl;
  }
}
```


Questão 2 (50 pontos)

Escreva um programa em C++ que deverá ler uma matriz com M linhas x N colunas (M e N estabelecido via `#define`) de valores reais. O programa devera calcular o número de linhas e o número de colunas nulas da matriz, estes valores devem ser mostrados na tela. Além disso, os valores da matriz devem ser divididos pelo número de linhas nulas. Ao final, a matriz resultante deve ser impressa na tela.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void le_matriz (float matriz[][N])`, que preencherá toda a matriz. Sua solução deve apenas chamar esta função.

Exemplo de execução (com M=4 e N=4):

```
Informe matriz:
1 0 2 3
4 0 5 6
0 0 0 0
0 0 0 0
Matriz tem 2 linhas e 1 coluna nulas
Matriz resultante:
0.5  0  1  1.5
2   0  2.5  3
0   0   0   0
0   0   0   0
```

```
#include<iostream>
#include<iomanip>
#define M 4
#define N 4
using namespace std;
int main() {
  float m[M][N];
  int i,j;
  float cln, ccn, som;
  cln=0;
  ccn=0;
  cout<<"Informe a matriz N";
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      cin>>m[i][j];
    }
  }
  for (i=0;i<M;i++){
    som=0;
    for (j=0;j<N;j++){
      som=som+m[i][j];
    }
    if (som==0){
      cln++;
    }
  }
  for (j=0;j<N;j++){
    som=0;
    for (i=0;i<M;i++){
      som=som+m[i][j];
    }
    if (som==0){
      ccn++;
    }
  }
  cout<<"Matriz tem "<<cln<<" linhas e "<<ccn<<" colunas zeradas"<<endl;
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      m[i][j]=m[i][j]/cln;
    }
  }
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      cout<<setw(5)<<setprecision(2)<<m[i][j];
```



```
    }
    cout<<endl;
  }
}
```

Questão 2 (50 pontos)

O município de Blumenau, em Santa Catarina, para proteger os cidadãos de calamidades devidas a enchentes, monitora o nível (em metros) do rio Itajaí-Açu de tempos em tempos, e criou um sistema de alerta à população.

Todos os locais do município suscetíveis a enchentes tiveram suas cotas (em metros), em relação ao fundo do rio, mapeados e registrados. Foram também identificados e catalogados abrigos para os quais a população deve deslocar-se em caso de enchente.

Para cada local do município sujeito a enchentes foram registrados o código do local, sua cota (em metros) em relação ao fundo do rio, o código do abrigo para onde seus moradores devem deslocar-se em caso de enchente e o número de avisos emitidos para aquele local (inicialmente com valor zero e incrementado de 1 a cada processamento do programa com situação de enchente detectada).

Escrever um programa em C++ que leia do usuário o nível atual do rio e uma matriz de monitoramento de 4 linhas e N colunas (N estabelecido via `#define`), cada coluna correspondente aos dados dos locais do município sujeitos a enchentes e cada linha correspondente a código do local, cota, código de abrigo e número de avisos, conforme descrito anteriormente. O programa deve mostrar, para os locais com cotas em situação de enchente, os locais, cotas e abrigos e, ainda, incrementar o número de avisos emitidos para aquele local na matriz.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void le_matriz (float matriz[][N])`, que preencherá toda a matriz. Sua solução deve apenas chamar esta função.

Exemplo de execução (para N = 9):

```
Nível do rio (m): 10.05
Matriz de monitoramento:
30  31   32   33   34   35  36    37    40
9   9.3  9.5  9.7  9.5  11  12.3  10.9  11
90  90   90   91   91   92  55    55    55
3   2    2    1    2    2    1    0     0

Enchentes:
Locais    Cotas     Abrigos
30        9         90
31        9.3       90
32        9.5       90
33        9.7       91
34        9.5       91

Matriz de monitoramento final:
30  31   32   33   34   35  36    37    40
9   9.3  9.5  9.7  9.5  11  12.3  10.9  11
90  90   90   91   91   92  55    55    55
4   3    3    2    3    2    1    0     0
```

```
// falta fazer
```


Questão 2 (50 pontos)

Escreva um programa em C++ que deverá ler uma matriz com M linhas x N colunas (M e N estabelecidos via `#define`) de valores inteiros, onde cada elemento da matriz representa uma cota (em centímetros) do correspondente metro quadrado de um terreno de $M \times N$ metros que precisa ser aplainado. Ao final, o programa deve imprimir, como no exemplo abaixo, o volume de terra (em metros cúbicos) que será necessário adicionar (ou retirar) para fazer a terraplenagem de toda a área do terreno, visando colocar todo o terreno em uma cota uniforme de 128 centímetros.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Sua solução deve apenas chamar esta função.

Exemplo de execução (com M=3 e N=6):

```
Informe a matriz original:
19   68   58   80   21   177
66   28   30   32   34   156
78   58   60   52   74   172
Qtde. de terra a acrescentar: 10.41 m3
```

Outro exemplo de execução (com M=3 e N=6):

```
Informe a matriz original:
219  168  158  180  125  217
166  128  130  232  234  136
278  158  160  152  214  152
Qtde. de terra a retirar: 9.03 m3
```

```
#include<iostream>
#define M 3
#define N 6
using namespace std;
int main(){
  int m[M][N]={{19,68,58,80,21,177},{66,28,30,32,34,156},{78,58,60,52,74,172}};
  //int m[M][N]={{219,168,158,180,125,217},{166,128,130,232,234,136},{278,158,160,152,214,152}};
  int i,j;
  float qtt=0,xx;
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      xx=1.28-(m[i][j]*0.01);
      qtt=qtt+xx;
    }
  }
  if (qtt<0){
    cout<<"Quantidade de terra a retirar: "<<qtt*-1<<" m3"<<endl;
  }
  else {
    cout<<"Quantidade de terra a acrescentar: "<<qtt<<" m3"<<endl;
  }
}
```

4.2 2014 - s2

Questão 1 (50 pontos)

A sequência de *Fibonacci* é uma sequência de números que começa com 0, 1 e que daí por diante, cada elemento (**e**) terá um valor (**v**) igual à soma do último (**u**) com o penúltimo (**p**) elemento, como demonstrado na tabela 1. Escreva um programa em C++ que leia uma sequência de números, e que para cada número n lido, calcule a sequência dos n primeiros números de Fibonacci. Quando o usuário inserir um número negativo na entrada, o programa deverá parar a sua execução.

elemento	1	2	3	4	5	6	7	...
ultimo	-	-	1	1	2	3	5	...
penúltimo	-	-	0	1	1	2	3	...
valor (u + p)	0	1	1	2	3	5	8	...

Tabela 1: Valores da sequência de *Fibonacci*.

Exemplo de execução:

```
Entre com o valor: 1
Sequência: 0
Entre com o valor: 2
Sequência: 0 1
Entre com o valor: 3
Sequência: 0 1 1
Entre com o valor: 4
Sequência: 0 1 1 2
Entre com o valor: 11
Sequência: 0 1 1 2 3 5 8 13 21 34 55
Entre com o valor: -1
Fim de execução.
```



```
#include<iostream>
using namespace std;
int main() {
  int i,u,p,v,n;
  while (1==1){
    cout<<"Entre com o valor ";
    cin>>n;
    if (n<0){
      cout<<"Fim da execucao "<<endl;
      return 0;
    }
    p=0;
    u=1;
    if (n==1){
      cout<<"0"<<endl;
    }
    if (n==2){
      cout<<"0 1"<<endl;
    }
    if (n>2){
      cout<<"0 1 ";
      i=2;
      while(i<n){
        v=p+u;
        cout<<v<<' ';
        p=u;
        u=v;
        i++;
      }
      cout<<endl;
    }
  }
}
```

Questão 2 (50 pontos)

Faça um programa em C++ que crie uma matriz simétrica. O programa deve calcular a soma dos valores que estão acima da diagonal principal e a soma dos valores que estão abaixo da diagonal e compará-los. A parte que possuir a maior soma será espelhada na outra parte da matriz. Considere para essa operação que a matriz utilizada será quadrada de dimensões N linhas e N colunas, (N estabelecido via diretiva `#define`).

Neste programa deve ser criada e usada uma função chamada `matriz_simetrica` que recebe como argumento a matriz a ser transformada em simétrica. Essa função deve calcular a soma dos valores acima da diagonal principal, a soma dos valores abaixo da diagonal principal e espelhar a matriz original com os valores da parte que tiver a maior soma na outra parte da matriz, a partir da diagonal principal.

OBS.: Para obter os dados da matriz e para imprimir os dados da matriz, considere que já existem as funções `void ler_matriz (int matriz[][N])` e `void imprimir_matriz(int mat[][N])` respectivamente, já codificadas e disponíveis para uso. Sua solução deve apenas chamar estas funções onde necessário.

Exemplo de execução (para N = 4):

```
Entre com a matriz:
1  2  3  4
5  6  7  8
9  10 11 12
13 14 15 16
Soma acima da diagonal principal: 36
Soma abaixo da diagonal principal: 66
Matriz simétrica:
1 5 9 13
5 6 10 14
9 10 11 15
13 14 15 16
```

```
#include<iostream>
#define N 4
using namespace std;
int main() {
  int m[N][N];
  int i,j;
  cout<<"Informe a matriz ";
  for (i=0;i<N;i++){
    for(j=0;j<N;j++){
      cin>>m[i][j];
    }
  }
  int aba,aci;
  aba=0;
  aci=0;
  for (i=0;i<N;i++){
    for(j=0;j<N;j++){
      if (i>j){
        aba=aba+m[i][j];
      }
      if (j>i){
        aci=aci+m[i][j];
      }
    }
  }
  cout<<"Soma acima "<<aci<<endl;
  cout<<"Soma abaixo "<<aba<<endl;
  if (aci<aba){
    for (i=0;i<N;i++){
      for(j=0;j<N;j++){
        if (i<j){
          m[i][j]=m[j][i];
        }
      }
    }
  }
  else {
    for (i=0;i<N;i++){
      for(j=0;j<N;j++){
        if (j<i){
          m[i][j]=m[j][i];
        }
      }
```



```
    }
  }
  for (i=0;i<N;i++){
    for(j=0;j<N;j++){
      cout<<m[i][j]<<' ';
    }
    cout<<endl;
  }
}
```

Questão 1 (50 pontos)

Escrever um programa em C++ que leia do teclado um par de números inteiros não negativos a e b, correspondentes aos limites inferior e superior, nesta ordem, de um intervalo $[a, b]$, com $a \leqslant b$. A seguir, ler ainda do teclado um número inteiro $N(1 \leqslant N \leqslant 5)$ e um conjunto de N números inteiros (que denominaremos divisores). Na seqüência, o programa deverá listar os números do intervalo lido, incluindo seus limites, que não são divisíveis por nenhum dos N números do conjunto de divisores lido. Ao final, mostrar a quantidade de números do intervalo que foram listados. Seu programa deve validar o intervalo informado, o valor de N e também a inexistência de zeros no conjunto de divisores informado.

Exemplo de execução:

```
[a,b]: 25 15
Limites inválidos
```

Outro exemplo de execução:

```
[a,b]: 15 25
Qtde. divisores (N): 8
Qtde. inválida
```

Outro exemplo de execução:

```
[a,b]: 15 25
Qtde. divisores (N): 3
Divisor 1: 2
Divisor 2: 0
    Inválido - reentre
Divisor 2: -3
Divisor 3: 4
Valores em [a,b] não divisíveis
   pelos divisores informados:
      17  19  23  25
Números listados: 4
```

```
#include<iostream>
using namespace std;
int main() {
  int a,b,n,i,qd,listados;
  int divs[5];
  listados=0;
  cout<<"[a,b] ";
  cin>>a>>b;
  if (b<a){
    cout<<"Limites invalidos "<<endl;
    return 0;
  }
  cout<<"Qtde divisores (N): ";
  cin>>n;
  if ((n<1)||(n>5)){
    cout<<"Qtd. invalida ";
    return 0;
  }
  i=0;
  for (i=0;i<n;i++){
    cout<<"Divisor "<<i+1<<": ";
    cin>>divs[i];
    if (divs[i]==0){
      cout<<"Invalido - reentre "<<endl;
      i--;
    }
  }
  cout<<"valores nao divisiveis "<<endl;
  while (a<=b){
    qd=0;
    for (i=0;i<n;i++){
      if ((a%divs[i])==0){
        qd++;
      }
    }
```



```
    if (qd==0){
      cout<<a<<' ';
      listados++;
    }
    a++;
  }
  cout<<endl<<"Numeros listados "<<listados<<endl;
}
```

Questão 2 (50 pontos)
Fazer programa em C++ que lê uma matriz quadrada $N \times N$, (N estabelecido via diretiva `#define`), calcula a soma dos valores na diagonal principal e a soma dos valores na diagonal secundária, imprimindo na tela o resultado. O cálculo deve ser feito por uma função que recebe a matriz e retorna os dois resultados.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Sua solução deve apenas chamar esta função onde necessário.

Exemplo de execução (com N=3):

```
Informe Matriz:
3  4  5
2  3  6
7  1  2
Soma diagonal principal = 8
Soma diagonal secundaria = 15
```

```
#include<iostream>
#define N 3
using namespace std;
int soma2 (int m[N][N], int & ss){
  int i,j,sp=0;
  ss=0;
  for (i=0;i<N;i++){
    for (j=0;j<N;j++){
      if (i==j){
        sp=sp+m[i][j];
      }
      if (i+j==N-1){
        ss=ss+m[i][j];
      }
    }
  }
  return sp;
}

int main(){
  int m[N][N]={{3,4,5},{2,3,6},{7,1,2}};
  int sp,ss;
  sp=soma2(m,ss);
  cout<<"Soma diagonal princial: "<<sp<<endl;
  cout<<"Soma diagonal secundária: "<<ss<<endl;
}
```


Questão 1 (50 pontos)

Um inteiro n é dito um **número perfeito** se ele é igual a soma de todos os seus divisores positivos, excluindo ele mesmo. Por exemplo, 6 é um número perfeito, pois seus divisores são 1, 2 e 3 e 1+2+3 = 6. O número 28 também é perfeito, pois 1+2+4+7+14=28. Faça um programa em C++ que leia uma sequência de números inteiros $n_1, n_2, \ldots, n_k$ terminada por zero e informe ao usuário, para cada n_i lido, se ele é um número perfeito ou não. O número 0 apenas indica o final da sequência e não deve ser considerado para a verificação da propriedade.

Exemplo de execução:

```
Entre com um inteiro: 2
2 não é um número perfeito.
Entre com um inteiro: 28
28 é um número perfeito.
Entre com um inteiro: 79
79 não é um número perfeito.
Entre com um inteiro: 8128
8128 é um número perfeito.
Entre com um inteiro: 0
```

```
#include<iostream>
using namespace std;
int main(){
  int num,dii,som;
  while(1==1){
    cout<<"Entre com um inteiro: ";
    cin>>num;
    if (num==0){
      break;
    }
    dii=1+(num/2);
    som=0;
    while(dii>0){
      if (num%dii==0){
        som=som+dii;
      }
      dii--;
    }
    if ((num==som)&&(num!=1)){
      cout<<num<<" e um numero perfeito."<<endl;
    }
    else {
      cout<<num<<" nao e um numero perfeito."<<endl;
    }
  }
}
```

Questão 2 (50 pontos)

Em um engradado de bebidas, pode-se ter: Água (1), refrigerante (2), cerveja (3) ou nada (0). Um refrigerante possui, em média, 10,8 gramas de açúcar para cada 100 ml de bebida, uma cerveja tem em média 5% de álcool na sua composição e a água não possui nada de álcool ou açúcar.

Dado um engradado de bebidas, representado por uma matriz $M \times N$, sendo os valores de M e N estabelecidos pela diretriz `#define`, escreva um programa em C++ que leia a matriz e **utilize uma função** que receba a matriz e calcule: o número total de garrafas, o total de litros de bebida; o total de gramas de açúcar; o total em litros de álcool; e a concentração por litro de açúcar em todo o engradado. Todas as garrafas do engradado possuem a mesma capacidade e ela é fornecida pelo usuário no início do programa.

OBS.: Para a leitura da matriz que representa o engradado, considere que já existe a função `void ler_matriz (int matriz[][N])` já codificada e disponível para uso. Sua solução deve apenas chamar estas funções onde necessário.

Exemplo de execução (para M=2 e N=6):

```
Quantos litros têm as garrafas? 0.6
Entre com o engradado:
1 2 3 3 2 0
0 0 2 2 1 1
Total de garrafas: 9
Total de líquido: 5.4 l
Total de açúcar: 259.2 g
Total de álcool: 0.06 l
Concentração total de açúcar: 48 g/l.
```

Outro exemplo de execução (para M=2 e N=6):

```
Quantos litros têm as garrafas? 1
Entre com o engradado:
3 3 3 3 2 3
0 0 2 2 1 1
Total de garrafas: 10
Total de líquido: 10 l
Total de açúcar: 324 g
Total de álcool: 0.25 l
Concentração total de açúcar: 32.4 g/l.
```

```
#include<iostream>
#define M 2
#define N 6
using namespace std;
int bebida(int m[M][N], float capa, float & tlit, float & tacu, float & talc,
  float & conc, int & totg){
  int i,j;
  totg=0;
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      if (m[i][j]!=0){
        tlit=tlit+capa;
        totg++;
      }
      if (m[i][j]==2){
        tacu=tacu+(capa/0.1)*10.8;
      }
      if (m[i][j]==3){
        talc=talc+capa*0.05;
      }
    }
  }
  conc=tacu/tlit;
}

int main(){
  int i,j,totg;
  int m[M][N]={{1,2,3,3,2,0},{0,0,2,2,1,1}};
  int m[M][N]={{3,3,3,3,2,3},{0,0,2,2,1,1}};
  float capa,tlit,tacu,talc,conc;
  cout<<"Quantos litros tem as garrafas ? ";
  cin>>capa;
  tlit=tacu=talc=conc=0;
  bebida(m,capa,tlit,tacu,talc,conc,totg);
  cout<<"Total de garrafas: "<<totg<<endl;
  cout<<"Total de liquido: "<<tlit<<" l."<<endl;
  cout<<"Total de acucar: "<<tacu<<" g."<<endl;
  cout<<"Total de alcool: "<<talc<<" l."<<endl;
  cout<<"Concentracao total de acucar: "<<conc<<" g/l."<<endl;
}
```


4.3 2015

Questão 1 (50 pontos)

Sua grana estava curta e então decidiu trabalhar temporariamente para o IBGE como agente do censo brasileiro. Sua remuneração depende de quantas residências são avaliadas por dia. Em cada residência, você faz o levantamento de quantas pessoas são homens e quantas são mulheres. Porém, o IBGE não pode pagar além de uma determinada quantidade de residências por dia.

Escreva um programa em C++ que, após obter do usuário a quantidade de dias trabalhados e a cota máxima diária de residências que o IBGE pode pagar ao agente, obtenha para cada um dos dias trabalhados, pares de valores representando a quantidade de pessoas entrevistadas no levantamento de cada residência, sendo contabilizado primeiro os homens e depois as mulheres. O final dos dados de um dia é indicado quando o usuário fornece o par de valores (-1, -1). Ao final da execução, mostre a quantidade de casas avaliadas e de homens e mulheres entrevistados. Mostre também qual a média de homens e de mulheres por residência considerados na entrevista.

OBS.: Residências avaliadas durante um mesmo dia que ultrapassem a cota diária devem ser ignoradas nos cálculos de valores totais e médias.

Exemplo de execução:

```
Digite a quantidade de dias trabalhados: 3
Digite a cota diaria: 2
Dia 1:
1 2
3 4
5 6
-1 -1
Dia 2:
0 3
3 0
-1 -1
Dia 3:
5 1
-1 -1
Total de casas: 5
Total de homens: 12
Total de mulheres: 10
Media de homens por casa: 2.4
Media de mulheres por casa: 2
```

```
#include<iostream>
using namespace std;
int main(){
  int hom=0,mul=0,cas=0;
  int qtdias=0,cotad=0,dia=1,qm,qh,qtc;
  cout<<"Digite a qtd dias: ";
  cin>>qtdias;
  cout<<"Digite a quota diaria: ";
  cin>>cotad;
  while (dia<=qtdias){
    cout<<"Dia "<<dia<<": "<<endl;
    qh=0;qm=0;qtc=0;
    while (1==1){
      cin>>qh>>qm;
      if ((qh==-1)&&(qm==-1)){
        break;
      }
      if (qtc<cotad){
        hom=hom+qh;
        mul=mul+qm;
        cas=cas+1;
      }
      qtc=qtc+1;
    }
    dia=dia+1;
  }
  float mh,mm;
  cout<<"Total de casas: "<<cas<<endl;
  cout<<"Total de homens: "<<hom<<endl;
  cout<<"Total de mulheres: "<<mul<<endl;
```



```
  mh=hom/(cas*1.0);
  mm=mul/(cas*1.0);
  cout<<"Media h/c: "<<mh<<endl;
  cout<<"Media m/c: "<<mm<<endl;
}
```

Questão 2 (50 pontos)

Faça um programa em C++ que leia uma matriz quadrada $N \times N$ de números inteiros (N estabelecido via `#define`). Lida a matriz, o programa deve mostrar o valor da soma dos elementos acima da diagonal e o conteúdo da matriz. Para produzir estes resultados, deve ser criada e usada no programa a função de nome `calculo_matriz` que recebe como argumento uma matriz $N \times N$ e retorna a soma dos valores que estão acima da diagonal principal, além de alterar o valor de todos os elementos da mesma matriz, multiplicando cada valor original da matriz pelo valor da soma calculada.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Para imprimir os valores da matriz na tela, considere que já existe a função `void imprimir_matriz (int matriz[][N])`. Sua solução deve apenas CHAMAR estas funções onde apropriado.

Exemplo de execução (com N=4):

```
Informe matriz:
1  0  0  0
5  6  1  1
9  2  3  0
1  0  4  8
Soma acima da diagonal princ.: 2
Matriz final:
2  0  0  0
10 12 2  2
18 4  6  0
2  0  8  16
```

```
#include<iostream>
#include<iomanip>
#define N 4
using namespace std;
int calculo_matriz(int m[N][N]){
  int i,j,som=0;
  for (i=0;i<N;i++){
    for(j=0;j<N;j++){
      if (j>i){
        som=som+m[i][j];
      }
    }
  }
  for (i=0;i<N;i++){
    for(j=0;j<N;j++){
      m[i][j]=m[i][j]*som;
    }
  }
  return som;
}

int main(){
  int m[N][N]={{1,0,0,0},{5,6,1,1},{9,2,3,0},
  {1,0,4,8}};
  int s,i,j;
  s=calculo_matriz(m);
  cout<<"Soma acima da d.p.: "<<s<<endl;
  for (i=0;i<N;i++){
    for(j=0;j<N;j++){
      cout<<setw(4)<<m[i][j];
    }
    cout<<endl;
  }
}
```


Questão 1 (50 pontos)

Escreva um programa em C++ que leia do teclado um conjunto de pares de números inteiros. O primeiro número de cada par poderá ser negativo, positivo ou nulo. O segundo número deverá ser diferente de zero e serve para indicar, de acordo com seu sinal, se uma série de números que deve ser gerada termina (caso de sinal negativo) ou inicia (sinal positivo) com o primeiro número informado. A série a ser gerada e exibida no monitor de vídeo deverá ser do tipo progressão aritmética, com razão e número de termos total dados pelo valor absoluto do segundo número digitado. Caso o segundo número do par informado seja nulo, deverá ser exibida mensagem ao usuário (vide exemplo) e desconsiderado o par. O final do conjunto de pares é assinalado com a informação de zeros para ambos os números do par, e este par não faz parte do conjunto. Ao final do processamento de cada par de números, o programa deverá mostrar a soma dos termos da série gerada e, ao final do programa, a quantidade de séries geradas (a partir de pares válidos).

Exemplo de execução:

```
Par de inteiros 1: -5 0
Segundo no. inválido
Par de inteiros 1: 0 -5
Série gerada:
-20 -15 -10 -5 0
Soma dos termos da série: -50
Par de inteiros 2: -20 10
Série gerada:
-20 -10 0 10 20 30 40 50 60 70
Soma dos temos da série: 250
Par de inteiros 3: 100 7
Série gerada:
100 107 114 121 128 135 142
Soma dos termos da série: 847
Par de inteiros 4: 0 0

Detectado o final dos pares de números
Séries geradas: 3
```

```
#include<iostream>
#include<iomanip>
#include<cmath>
#define N 4
using namespace std;
int main() {
  int p,s,qt=1,q,se,so;
  while (1==1){
    cout<<"Par: "<<qt<<" ";
    cin>>p>>s;
    if ((p==0)&&(s==0)){
      break;
    }
    if (s==0){
      cout<<endl<<"Segundo invalido"<<endl;;
      continue;
    }
    qt++;
    cout<<"Serie gerada "<<endl;
    so=0;
    if (s>0){
      se=p;
      q=1;
      while (q<=s){
        cout<<se<<" ";
        so=so+se;
        se=se+s;
        q=q+1;
      }
      cout<<endl<<"Soma: "<<so<<endl;
    }
    else {
      se=p+s*(s+1)*-1;
      q=1;
      while (q<=s*-1){
        cout<<se<<" ";
        so=so+se;
        se=se-s;
        q++;
      }
```



```
      cout<<endl<<"Soma: "<<so<<endl;
    }
  }
  cout<<"detectado fim"<<endl;
  cout<<"Series geradas"<<qt<<endl;
}
```

Questão 2 (50 pontos)

Faça um programa em C++ que leia a partir do teclado uma matriz $M \times N$ de números inteiros (M e N estabelecido via `#define`) representando notas de escolas de samba. A matriz possui uma linha para cada escola e uma coluna para cada jurado. O programa deve descartar a menor nota atribuída pelos jurados às escolas de samba e calcular a quantidade de escolas nota 10. Uma escola é nota 10 se possui todas as notas iguais a 10 (dez) após o descarte. Para resolver o problema proposto pede-se o desenvolvimento e uso de uma função de nome `apuracao()` que recebe como argumento uma matriz $M \times N$ de números inteiros, substitui a menor nota de cada escola pelo valor 0 (zero), calcula e retorna a quantidade de escolas nota 10.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Para imprimir os valores da matriz na tela, considere que já existe a função `void imprimir_matriz (int matriz[][N])`. Sua solução deve apenas CHAMAR estas funções onde apropriado.

Exemplo de execução (com M=3 e N=4):

```
Informe matriz:
7  10 9  10
10 10 10 8
10 10 10 10
Menores notas descartadas:
0  10 9  10
10 10 10 0
0  10 10 10
Qtde. escolas nota 10: 2
```

```
#include<iostream>
#include<iomanip>
#include<cmath>
#define M 3
#define N 4
using namespace std;
int apurac(int m[M][N]){
  int i,j,qt,escolas=0;
  int menor,jmen;
  for (i=0;i<M;i++){
    menor=9999999;
    for (j=0;j<N;j++){
      if (m[i][j]<menor){
        menor=m[i][j];
        jmen=j;
      }
    }
    m[i][jmen]=0;
  }
  for (i=0;i<M;i++){
    qt=0;
    for (j=0;j<N;j++){
      if (m[i][j]==10){
        qt++;
      }
    }
    if (qt==N-1){
      escolas++;
    }
  }
  return escolas;
}

int main() {
  int i,j,s;
  int m[M][N]={{7,10,9,10},{10,10,10,8},
  {10,10,10,10}};
  s=apurac(m);
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
```



```
      cout<<setw(3)<<m[i][j];
    }
    cout<<endl;
  }
  cout<<"Quantidade de escolas: "<<s;
}
```

4.4 2016

Questão 1 (50 pontos)

Escrever um programa em C++ para o cálculo da soma de um número N de termos da série a seguir:

$$1+\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\cdots$$

Exemplo de execução:

Quantidade de termos a somar: **3**
Soma: 1.75

Outro exemplo de execução:

Quantidade de termos a somar: **0**
Entrada inválida

```
#include<iostream>
#include<cmath>
#include<iomanip>
#define M 4
#define N 4
using namespace std;
int q1(){
  int i,n;
  float s;
  s=0;
  cout<<"Quantidade de termos a somar: ";
  cin>>n;
  if (n<1){
    cout<<"Entrada invalida"<<endl;
  }
  else {
    for (i=1;i<=n;i++){
      s=s+1/pow(2,i-1);
    }
    cout<<"Soma: "<<s<<endl;
  }
}
```


Questão 2 (50 pontos)

Crie um programa que lê uma matriz de $M \times N$ valores inteiros do teclado, onde M e N devem ser pares (M e N definidos via diretiva `#define`). Sabendo que se dividirmos a matriz ao meio horizontalmente e verticalmente, obteremos quatro quadrantes, crie uma função chamada `trocaQuadrantes()` que recebe uma matriz de valores inteiros $M \times N$. O objetivo da função é trocar os elementos do quadrante superior esquerdo com os do inferior direito da matriz, e os elementos do quadrante inferior esquerdo com os elementos do superior direito. A função também deve retornar a média dos elementos do quadrante superior esquerdo. A função `trocaQuadrantes()` não deve ler dados do teclado nem imprimir informações na tela. Utilize no programa principal (int main) a função que você criou, sendo que você deve imprimir a matriz resultante na tela.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int matriz[][N])`. Para a impressão de uma matriz, considere que já existe a função `void imprime_matriz (int matriz[][N])`.

Sua solução deve apenas CHAMAR estas funções onde for apropriado.

Exemplo de execução (com M=4, N=4):

```
Matriz original
0   1   2   3
4   5   6   7
8   9   10  11
12  13  14  15
Resultado troca quadrantes:
10  11  8   9
14  15  12  13
2   3   0   1
6   7   4   5
Media quadrante sup. esq.: 12.5
```

Exemplo de execução (com M=4, N=6):

```
Matriz original
0   1   2   3   4   5
4   5   6   7   8   9
8   9   10  11  12  13
12  13  14  15  16  17
Resultado troca quadrantes:
11  12  13  8   9   10
15  16  17  12  13  14
3   4   5   0   1   2
7   8   9   4   5   6
Media quadrante sup. esq.: 14
```

```
int mostramat(int ma[M][N]){
  int i,j;
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      cout<<setw(4);
      cout<<ma[i][j];
    }
    cout<<endl;
  }
}

int trocaquadrantes(int mat[M][N]){
  int metm,metn,i,j,aux,som;
  float med;
  metm=M/2;
  metn=N/2;
  som=0;
  for (i=0;i<metm;i++){
    for (j=0;j<metn;j++){
      aux=mat[i][j];
      mat[i][j]=mat[i+metm][j+metn];
      mat[i+metm][j+metn]=aux;
      som=som+mat[i][j];
    }
  }
  for (i=metm;i<M;i++){
    for (j=0;j<metn;j++){
      aux=mat[i][j];
      mat[i][j]=mat[i-metm][j+metn];
      mat[i-metm][j+metn]=aux;
    }
  }
  med= som/(metm*metn*1.0);
  cout<<"Media quadrante sup. esq.: "<<med<<endl;
}

int q2(){
```



```
  int x[M][N]={{0,1,2,3},{4,5,6,7},{8,9,10,11},{12,13,14,15}};
  mostramat(x);
  trocaquadrantes(x);
  cout<<endl;
  mostramat(x);
}
```

Questão 1 (50 pontos)

Faça um programa em C++ que leia um polinômio arbitrário (cujos expoentes de x são valores inteiros no intervalo $[0, 99]$) e calcule o seu valor (veja abaixo os exemplos de como deve ser feito). O valor de x a ser utilizado no cálculo do polinômio deve ser também entrado via teclado. A ordem de entrada dos expoentes deverá ser obrigatoriamente decrescente, sendo que ao atingir o grau zero (x^0) somente a constante do polinômio deverá ser requisitada, finalizando então a função.

OBS.: (a) Os coeficientes do polinômio são números reais quaisquer; (b) Os expoentes são inteiros entre $[0, 99]$, decrescentes; (c) Vetores **não** deverão ser utilizados neste programa.

Exemplo de execução:

```
Valor de x: 2
Entre com o polinômio:
x^500
x^5
 * 1
+   x^3
 * 1
+   x^3
x^1
 * 1
+   1
Resultado: 43
```

Outro exemplo de execução:

```
Valor de x: 1.5
Entre com o polinômio:
x^2
 * 2
+   x^1
 * -1.5
+   0
Resultado: 2.25
```

Outro exemplo de execução:

```
Valor de x: 2
Entre com o polinômio:
x^4
 * 5
+   x^1
 * 3
+   -1
Resultado: 85
```

```
#include<iostream>
#include<cmath>
using namespace std;
int main(){
  int ex;
  float t=0,x,coef,fin;
  ex=100;
  cout<<"Entre com o valor de x: ";
  cin>>x;
  while ((ex<0) || (ex>99)) {
    cout<<"x^";
    cin>>ex;
  }
  while (ex>=1){
    cout<<"c * ";
    cin>>coef;
    t=t+coef*pow(x,ex);
    if (ex==1) {break;}
    cout<<"x^";
    cin>>ex;
  }
  cout<<"+ ";
```



```
  cin>>fin;
  t=t+fin;
  cout<<"Resultado: "<<t<<endl;
}
```

Questão 2 (50 pontos)

Faça um programa em C++ que leia uma matriz de inteiros $A_{M\times N}$ (M e N definidos via `#define`) e que atribua zero a todos os elementos da matriz da forma $A_{i,j}$ se a soma de seus índices for par, ou seja, se $i+j$ for par (considere 0 par). Depois disso defina uma função que retorne o maior elemento da matriz A transformada, imprimindo a linha e coluna na qual ele se encontra (imprima o primeiro encontrado caso existam outros elementos de igual valor). O programa deve chamar esta função e imprimir seu resultado no programa principal.

OBS.: Para a leitura de uma matriz, considere que já existe a função `void ler_matriz (int A[][N])`. Para a impressão de uma matriz, considere que já existe a função `void imprime_matriz (int A[][N])`. Sua solução deve apenas CHAMAR estas funções onde for apropriado.

Exemplo de execução (com M=2 e N=5):

```
Informe matriz:
2   5    4   5   6
1  -2   -3   0   1
Matriz transformada:
0   5    0   5   0
1   0   -3   0   1
Maior valor de A: 5, linha 0, coluna 1
```

```
#include<iostream>
#include<iomanip>
#define M 2
#define N 5
using namespace std;
int transf(int m[M][N], int & lin, int & col){
  int i,j,maior=-999999;
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      if (m[i][j]>maior){
        maior=m[i][j];
        lin=i;
        col=j;
      }
    }
  }
  return maior;
}

int main(){
  int i,j,ima,jma,mai;
  int m[M][N]={{2,5,4,5,6},{1,-2,-3,0,1}};
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      if (((i+j)%2)==0) {
        m[i][j]=0;
      }
    }
  }
  mai=transf(m,ima,jma);
  cout<<"Matriz transformada: "<<endl;
  for (i=0;i<M;i++){
    for (j=0;j<N;j++){
      cout<<setw(3)<<m[i][j];
    }
    cout<<endl;
  }
  cout<<"Maior valor: "<<mai<<" linha: "<<ima<<" coluna: "<<jma;
}
```

