Truques de programação V - C++

Nesta folha você deve resolver alguns exercícios de programação. Cada um deles sugere um truque que quando apreendido pode ser usado em inúmeros outros problemas parecidos ou não.

Para seu processamento, você deve ler o arquivo

`F187001.myd`

publicado no lugar usual.

Operações com conjuntos Imagine uma matriz de 6 linhas por 13 colunas, na qual: cada linha corresponde a um conjunto de até 12 números. O primeiro elemento de cada linha não é um elemento e sim um contador de elementos válidos naquela linha. As 3 primeiras linhas correspondem aos conjuntos A, B e C e já vêm preenchidas. As outras linhas correspondem aos conjuntos D, E e F e devem ser calculados pelo programa, usando as seguintes regras:
$D = A \cap B$, $E = D \cup C$ e $F = B \cap C$. Escreva um programa C++ que leia uma matriz deste tipo, atualize as linhas 4..6 e imprima a matriz completa.

...

Truques: O primeiro é o conceito de conjunto que elementos nos quais: i. não pode haver repetição de elementos e ii. A ordem dos elementos não interessa. Adicionalmente, a operação de intersecção (∩) devolve elementos que estão no primeiro e no segundo conjuntos. A união (∪) recupera elementos que estão no primeiro ou no segundo conjuntos.

Um truque adicional é contador de elementos válidos. É usado quando se simula um conjunto de tamanho variável (o conjunto) sobre uma estrutura de dados de tamanho fixo (a linha da matriz).

Para você No arquivo acima, há 100 matrizes (6×13 cada) Você deve processá-las e gerar em cada uma as linhas 4..6. Obtenha a soma das cardinalidades (a quantidade de elementos, ou em termos físicos a primeira coluna de cada linha) dos conjuntos D, E e F.

$\sum t(D)$	$\sum t(E)$	$\sum t(F)$

Determinante O determinante de uma matriz quadrada é um número real que informa algumas características da matriz e que pode e é usado em diversos processamentos sobre a matriz. O determinante de uma matriz 1×1 contendo um único valor é o próprio valor. Já uma matriz 2×2 tem como determinante a expressão $a.d - b.c$. Uma matriz 3×3 tem como determinante a expressão (tirada da regra de Sarrus) $aei + dhc + bfg - (gec + ahf + dbi)$. Para as ordens maiores, existe uma regra (recursiva) que vai ser adotada aqui. Para calcular o determinante de uma matriz de ordem 4, escolhe-se uma linha (ou coluna) qualquer e para todo elemento $m[i,j]$ dessa linha, calcula-se a expressão $m[i,j] \times (-1)^{i+j} \times det(m')$ onde m' é a matriz resultante quando se extrai de m a linha i e a coluna j.

```
#include<iostream>
#include<fstream>
#include<iomanip>
#define N 4
using namespace std;
void tira(float vai[N][N], float volta[N][N],
     int tam, int qual){
   int i,j,k;
   i=0; k=0;
   while (i<tam){
       if (i==qual) {i++;}
       for (j=1;j<tam;j++){
           volta[k][j-1]=vai[i][j];
       }
       k++;
       i++;
   }
}
float det(float m[N][N],int tam){
    float x,y;
    float mlin[N][N];
    int k1,k2,i,j,k;
    if (tam<4){
       x=m[0][0]*m[1][1]*m[2][2];
       x=x+m[1][0]*m[2][1]*m[0][2];
       x=x+m[0][1]*m[1][2]*m[2][0];
       y=m[2][0]*m[1][1]*m[0][2];
       y=y+m[0][0]*m[2][1]*m[1][2];
       y=y+m[1][0]*m[0][1]*m[2][2];
       return x-y;
  }
  else {
      x=0;
      i=0;
      while (i<tam){
         tira(m,mlin,tam,i);
         y=det(mlin,tam-1);
         if (((1+i)%2)==0){
            x=x+m[i][0]*-1*y;
         }
         else {
            x=x+m[i][0]*y;
         }
         i++;
      }
      return x;
   }
}
int main() {
  float m[N][N];
  int i,j;
  fstream f;
  float x;
  f.open("c:/p/n/....txt");
  for (i=0;i<N;i++){
    for (j=0;j<N;j++){
       f>>m[i][j];
    }
  }
  for (i=0;i<N;i++){
    for (j=0;j<N;j++){
       cout<<setw(5)<<m[i][j];
    }
    cout<<endl;
  }
  x=det(m,N);
  cout<<"Determinante = "<<x;
}
```

Truque: recursividade Como a definição de determinante usa a recursividade (a solução de um determinante 4 é igual a 3 vezes a solução de um determinante 3...), tem-se o ambiente ideal para a recursividade: i.mesma solução, ii. universo menor e iii. caso básico (det=3).

Para você No arquivo acima, há 40 matrizes quadradas de 6×6. Ache o determinante delas e conte quantas dessas 40 têm determinante $\neq 0$.

det $\neq 0$

Matriz inversa A matriz inversa surge em inúmeros momentos da álgebra: por exemplo na solução de sistemas lineares. Uma matriz A poderá ter uma matriz inversa chamada A^{-1} que é única e que quando multiplicada matrizialmente à matriz original A reproduz a matriz identidade I. Quem é inversa de quem é mera convenção, já que uma é a inversa da outra. A matriz A só terá inversa se o seu determinante for diferente de zero.

```
#include<iostream>
#include<iomanip>
#define ta 5
using namespace std;
int cami(float x[ta][ta], float y[ta][ta]){
// x é a matriz A e y é a matriz A^-1
  int i,j,k;
  float pivot,alvo;
  for (i=0;i<ta;i++){
    for (j=0;j<ta;j++){
      if (i==j) {
        y[i][j]=1;
      }
      else {
        y[i][j]=0;
      }
    }
  }
  for (i=0;i<ta;i++){
    for (j=0;j<ta;j++){
      if (j!=i){
        pivot=-x[j][i]/x[i][i];
          for (k=0;k<ta;k++){
            x[j][k]=x[j][k]+x[i][k]*pivot;
            y[j][k]=y[j][k]+y[i][k]*pivot;
          }
      }
    }
    alvo=x[i][i];
    for (k=0;k<ta;k++){
      y[i][k]=y[i][k]/alvo;
      x[i][k]=x[i][k]/alvo;
    }
  }
  return 0;
}
int main() {
  float mat[ta][ta]={{...},{...},{},...};
  float res[ta][ta];
  int i,j;
  cami(mat,res);
  for (i=0;i<ta;i++){
    for (j=0;j<ta;j++){
      cout<<fixed<<setprecision(5)
            <<res[i][j]<<"  ";
    }
    cout<<endl;
  }
}
```

Para você No arquivo acima, há 20 matrizes (10 pares) de 10 linhas por 10 colunas. Você deve contar quantos pares realmente são inversos um do outro.

pares de inversos

Para ler o arquivo Para ler os dados de entrada de um arquivo (ao invés do teclado) em C++ deve-se: i.incluir o pacote `<fstream>`. ii.definir uma variável (digamos `f`) de tipo `ifstream`. iii. Abrir o arquivo passando a localização do mesmo. iv. fazer `f>>variável` (ao invés de `cin>>variável`). v. Ao final, fechar o arquivo (`f.close()`). Acompanhe

```
#include<fstream>
ifstream f;
f.open("c:/p/apl/...")
...
   f>>v[i];
...
f.close()
```

☞ Para você fazer

$\sum t(D)$	$\sum t(E)$	$\sum t(F)$
det $\neq 0$	pares inv	//////// //////// //////// ////// /////////

```
===== 04/12/2019 10:50:51.6 =====E=PL187c
    1      192  379  202       22        5
```