CAPÍTULO 5

Variáveis Indexadas: Vetores e Matrizes

CAPÍTULO 5

VARIÁVEIS INDEXADAS

Nem sempre os tipos básicos simples (*int, float, char, string, bool*) são suficientes para exprimir estruturas de dados em programação. Por exemplo, o problema de se escrever 100 notas seguidas da média; só a declaração destas 100 variáveis tornaria impraticável a redação do programa. É necessário que se tenha uma estrutura de dados que contenha todas as 100 notas e possa ser referenciada pelo conjunto (NOTAS), acessando cada nota individualmente.

NOTAS

Valor	5.0	3.0	8.5	7.6	.....	9.0	5.5
Posição	0	1	2	3	.....	98	99

As variáveis indexadas (ou compostas) referenciam grupos de variáveis do mesmo tipo (homogêneas), pelo mesmo nome. São individualizadas entre si através de uma posição dentro desse conjunto.

Ao número de índices necessários à localização de um elemento dentro de uma variável indexada dá-se o nome de dimensão.

Variáveis indexadas homogêneas unidimensionais (único índice): vetores
Variáveis indexadas homogêneas multidimensionais (dois ou mais índices): matrizes

VARIÁVEIS INDEXADAS HOMOGÊNEAS UNIDIMENSIONAIS (VETORES)

Conjunto de dados homogêneos referenciado por um mesmo nome que necessita de um único índice para ter seus elementos individualizados. O vetor possui uma única dimensão.

SINTAXE DA VARIÁVEL INDEXADA HOMOGÊNEA UNIDIMENSIONAL (VETOR) EM C++

tipo básico lista de variáveis [**tamanho**];

onde:

tamanho representa a quantidade de elementos que vai compor o vetor
tipo básico representa qualquer um dos tipos básicos (*int, float, char, bool*)
lista de variáveis representa os nomes de variáveis indexadas que terão o mesmo tamanho e o mesmo tipo definido acima

O número de elementos de um vetor será dado pelo tamanho (LI=1 e LF=tamanho)

LI	LI + 1	LI + 2	LI + 3	.....	LF – 1	LF

As posições do vetor são identificadas a partir de LI=0, com incrementos unitários até LF=tamanho-1.

Exemplo 1) Veja a declaração abaixo:

char NOME [5];

NOME[3] O 3 é o índice, que pode ser uma constante numérica (natural), um resultado numérico (natural) de uma expressão aritmética ou, ainda, uma variável numérica (com valor natural).

Como atribui-se um valor à uma posição do vetor:

NOME[4] = 'F';

cin >> NOME[0]; P

Como se mostra o elemento contido em uma posição do vetor:

cout << NOME[4]; F

Exercício 5.1) O que será impresso pelo programa?

```
int main()
{
        int i;
        int ve [6];
        char ca [6];

        ve[0] = 1;   ve[1] = 1;   ve[2] = 2;
        ve[3] = 5;   ve[4] = 0;   ve[5] = 1;
        ca[0] = 'S';   ca[1] = 'T';   ca[2] = 'Q';
        ca[4] = 'S';   ca[5] = 'D';
        for (i = 2; i <= 4;  i = i + 1)
                cout << ca[ve[i]] << endl;
        cout << ca[ve[i]] << endl;
        cout << ca[ve[ve[3]]] << endl;

        return 0;
}
```

Observação: Atenção para os índices que não têm nenhuma atribuição de valores, pois como estes estão alocados em um determinado espaço de memória, podem conter **"lixos"** (valores que já estavam na memória antes da alocação).

Podemos ainda atribuir valores a um vetor no momento de sua declaração, sendo que, eles devem vim entre chaves e separados por vírgulas, lembrando que tais valores serão inseridos na ordem de escrita, ou seja, o primeiro elemento dentro da chave será o elemento que ocupará a posição 0 do vetor e assim sucessivamente para os demais. Como exemplo, a declaração abaixo que mostra o vetor **notas** sendo inicializado logo na sua declaração.

```
float notas [ 5 ] = { 80 , 60.5 , 38.2 , 50 , 79 };
```

No entanto tais atribuições se tornam inviáveis quando necessitamos preencher um vetor com várias posições, ou quando queremos guardar nele os valores que serão lidos. Assim, utilizamos a estrutura de repetição **for** para efetuarmos tal atribuição.

No exemplo abaixo é possível notar que estamos preenchendo cada espaço do vetor **idades** com valores recebidos da entrada de dados. Isso acontece porque o comando de entrada **cin** se encontra dentro da estrutura de repetição e a cada passo do **for** (a cada incremento da variável **i**) o valor recebido na entrada de dados é armazenado no índice do vetor correspondente ao valor da variável **i** .

```
int idades [30] ;

for ( int i = 0 ; i < 30; i ++ )
        cin >> idades [ i ] ;
```

Exercício 5.2) Dado o vetor **crr** literal (de caracteres) abaixo:

!	U	O	T	R	E	C	A
0	1	2	3	4	5	6	7

Qual será a sua configuração depois de executados os comandos:

```
:
for (i = 1; i <= 3; i = i + 1)
{
  aux = crr[i];
  crr[i] = crr[6-i+1];
  crr[6-i+1] = aux;
}
aux = crr[0];
crr[0] = crr[7];
crr[7] = aux;
:
```

Todas as manipulações de um vetor podem ser feitas através da estrutura de repetição **for** (mas também pode, ser usadas as estruturas **while** e **do-while**), pois através dele é possível acessar cada elemento (cada índice) da estrutura indexada. Exemplos:

- atribuição de valores ao vetor
- processamento dos valores lidos e colocados no vetor
- como encontrar maior valor de um vetor
- a saída dos dados do vetor, etc.

Exercício 5.3) Se você tem um vetor com 100 valores numéricos:

a) Como seria a declaração de um vetor para armazenar esses 100 números?

b) Qual seria o bloco de comandos para ler os 100 números e armazená-los neste vetor?

c) Qual seria o bloco de comandos para calcular e imprimir a média dos 100 números deste vetor?

d) Qual seria o bloco de comandos para calcular e imprimir o maior número dentre os 100 valores deste vetor?

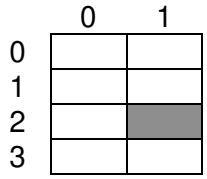
e) Qual seria o bloco de comandos para imprimir este vetor de 100 posições?

VARIÁVEIS INDEXADAS HOMOGÊNEAS MULTIDIMENSIONAIS (MATRIZES)

Conjunto de dados homogêneos referenciado por um mesmo nome que necessita de mais de um índice (2, 3....) para ter seus elementos individualizados.

Bidimensional:
1o. índice = linha
2o. índice = coluna

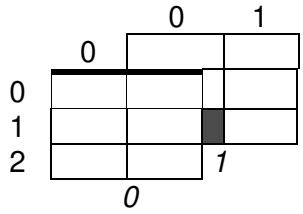
escaninho



O elemento marcado é escaninho[2][1]

Tridimensional:
1o. índice = linha
2o. índice = coluna
3o. índice = página ou face

livro



O elemento marcado é livro[2][0][1]

SINTAXE DA VARIÁVEL INDEXADA HOMOGÊNEA MULTIDIMENSIONAL (MATRIZ) EM C++

tipo básico lista de variáveis [**dimensão 1**] [**dimensão 2**]..... [**dimensão n**] ;

onde:

[**dimensão 1**] [**dimensão 2**]..... [**dimensão n**] representam as possíveis dimensões da matriz

tipo básico representa qualquer um dos tipos básicos (*int, float, char, bool*)

lista de variáveis representa os nomes de variáveis indexadas que terão a mesma dimensão e o mesmo tipo definido acima

Exemplo 2) Veja a declaração abaixo:

int patio [5] [3];

patio [1][2] os números 1 e 2 são índices, que podem ser constantes numéricas (naturais), resultados numéricos (naturais) de uma expressão aritmética ou, ainda, variáveis numéricas (com valores naturais).

Como atribui-se um valor à uma posição da matriz:

patio[4][0] = 0;

Atribui-se o valor zero à posição referente à linha 4 e coluna 0 da matriz PATIO.

cin >> patio[4][1]; 10

O usuário digitou 10 e esse valor foi atribuído à posição referente à linha 4 e coluna 1 da matriz patio.

Como se mostra o elemento contido em uma posição da matriz:

cout << patio[4][0]; 0

Executando-se este comando **cout**, na unidade de saída especificada sairá o valor zero que estava armazenado na posição referente à linha 4 e coluna 0 da matriz PATIO.

O número de dimensões de uma matriz é igual ao número de ([ ]) da declaração. O número de elementos da matriz (NEM) é igual ao produto do número de elementos de cada dimensão.

NEM = (tamanho da dimensão 1) x (tamanho da dimensão 2) x .... x (tamanho da dimensão n)

Exemplo 3) Declare a variável composta **a** de 4 linhas e 3 colunas constituída de elementos numéricos reais. Defina sua dimensão e a quantidade de elementos que podem ser armazenados nela.

float a [4] [3];

a possui 2 dimensões.
a possui 12 elementos = (4) x (3) = 12.

Exercício 5.4) Declare a variável indexada **paralelepipedo** com 2 linhas, 4 colunas e 6 faces, constituída de caracteres. Defina a sua dimensão e a quantidade de caracteres que ela pode armazenar.

Exercício 5.5) Qual o número de elementos e caracteres que as matrizes especificadas podem armazenar?

```
int mat1 [3] [5];
char mat2 [3] [3] [2];
```

Exercício 5.6) Dados a matriz **m** tridimensional e o vetor **vet** definidos por:

```
int m [2] [4] [3];
int vet [4];
```

e preenchidos com os seguintes números inteiros:

m:

0

	0	1	2	3
0	**0**	**1**	**2**	**3**
1	**5**	**-5**	**3**	**0**

1

	0	1	2	3
0	**1**	**2**	**-2**	**1**
1	**1**	**1**	**3**	**0**

2

	0	1	2	3
0	**0**	**0**	**1**	**1**
1	**-1**	**-1**	**2**	**2**

vet:

0	**2**
1	**3**
2	**0**
3	**1**

Determine os elementos:

a) vet [2] =

b) m [1][0][1] =

c) vet [m [0][2][2]] =

d) m [vet [3]] [vet [3]] [vet [0]] =

e) m [m [vet[2]][1][0]] [vet[m [1][2][1]]] [2] =

Exemplo 4) Seja a matriz abaixo:

```
int  vejabem [3][5];
```

vejabem

	0	1	2	3	4
0					
1					
2					

Para preenchermos a matriz, linha a linha, com elementos numéricos lidos, fixamos a linha e variamos a coluna:

```
for (i = 0 ; i <= 2 ; i = i + 1)
        for (j = 0 ; j <= 4 ; j = j + 1)
                cin >> vejabem [i][j];
```

Para preenchermos a matriz, coluna a coluna, fixamos a coluna e variamos a linha:

```
for (j = 0 ; j <= 4 ; j = j + 1)
        for (i = 0 ; i <= 2 ; i = i + 1)
                cin >> vejabem [i][j];
```

Assim como os vetores, as matrizes podem ser inicializadas tanto de forma direta, no ato de sua declaração, atribuindo elemento a elemento, ou através de uma estrutura de repetição, um conjunto de **for**'s.

```
int notas [5][4] = {{ 1 , 5 , 6 , 9 },{ 9 , 8 , 7 , 9 },{ 4 , 5 , 0 , 3 },{ 9 , 2 , 9 , 9 },{ 4 , 8 , 9 , 8 }};
```

Essa é a forma de atribuição de valores na matriz no ato da declaração. Para tal é necessário se atentar para as chaves que definem onde cada elemento se encontra, nota-se que os elementos de uma linha ficam dentro de chaves e separados por vírgula e ao fim agrupa-se todas as linhas com uma chave, definindo-se por fim uma matriz.

Uma outra forma de atribuição de valores à matriz é por meio de indexação, ou seja, referencia-se os índices e indica-se qual valor aquela posição [ ][ ] receberá, como no exemplo abaixo.

O aluno referente ao índice 0 recebeu uma nota 10 na matéria referente ao índice 1.

```
int notas [ 5 ][ 4 ] ;

notas [ 0 ][ 1 ] = 10 ;
notas [ 0 ][ 3 ] = 7 ;
notas [ 1 ][ 1 ] = 8 ;
```

No entanto, estas formas de atribuição são inviáveis para uma matriz com muitos elementos, faz-se então o uso de uma estrutura de repetição a fim de variar os índices da matriz e dessa forma tornar possível uma atribuição a partir dos elementos recebidos na entrada de dados.

```
int main ()
{
        int notas [ 5 ][ 4 ] ;

        // leitura dos elementos
        cout << "Entre com uma matriz 5x4" << endl ;
        for ( int i = 0 ; i < 5 ; i ++ )
        {
                for ( int j = 0 ; j < 4 ; j ++ )
                        cin >> notas [ i ][ j ] ;
        }
        // saida dos elementos
        cout << "SAIDA: " << endl ;
        for ( int i = 0 ; i < 5 ; i ++ )
        {
                for ( int j = 0 ; j < 4 ; j ++ )
                {
                cout << notas [ i ][ j ] << " " ;
                }
                cout << endl ;
        }
        return 0 ;
}
```

Exercício 5.7) Analise os programas (A1 e A2) e responda as questões a seguir:

```
int main ()     // programa A1
{
        int a[5];
        int b[5][5];
        int i, j, soma1, soma2;

        soma1 = 0;
        soma2 = 0;

        for (i = 0; i <= 4; i = i + 1)
        {
                cout <<  "Entre com o elemento " << i+1 << " de A: ";
                cin >> a [i];
                for (j = 0; j <= 4; j = j +1)
                {
                    cout <<  "Entre com o elemento " <<  i+1 << "," <<  j+1<< " de B: ";
                    cin >> b [i][j];
                }
        }
        for (i = 0; i <= 4; i = i + 1)
        {
                soma1 = soma1 + a[i];                   // comando C1
                for (j = 0; j <= 4; j = j +1)
                        soma2 = soma2 + b[i][j];        // comando C2
        }
        cout << "Soma 1 = " << soma1 << " e Soma 2 = " << soma2 << endl;

        return 0;
}
```

```
int main ()     // programa A2
{
        int a[5];
        int b[5][5];
        int i, j, soma1, soma2;

        soma1 = 0;
        soma2 = 0;

        for (i = 0; i <= 4; i = i + 1)
        {
                cout <<  "Entre com o elemento " << i+1 << " de A: ";
                cin >> a [i];
                for (j = 0; j <= 4; j = j +1)
                {
                    cout <<  "Entre com o elemento " <<  i+1 << "," <<  j+1<< " de B: ";
                    cin >> b [i][j];
                }
        }
        for (i = 0; i <= 4; i = i + 1)
        {
                for (j = 0; j <= 4; j = j +1)
                {
                        soma1 = soma1 + a[i];          // comando C1
                        soma2 = soma2 + b[i][j];       // comando C2
                }
        }
        cout << "Soma 1 = " << soma1 << " e Soma 2 = " << soma2 << endl;

        return 0;
}
```

a) Os dois programas fornecem a mesma saída? Se a resposta for não, justifique.

b) No programa A1, quantas vezes são executados os comandos C1 e C2?

c) No programa A2, quantas vezes são executados os comandos C1 e C2?

d) Elabore um enunciado para o programa A1.

Exercícios propostos do Capítulo 5:

P5.1) Um professor tem uma turma de 100 alunos e deseja calcular e escrever o nome e a nota de cada aluno seguida da média da turma. As notas (criticadas de 0 a 10) e os nomes são fornecidos pelo usuário.

P5.2) Elabore um programa que leia uma sequência qualquer de **n** números (n <= 100) e imprima-os em ordem inversa. **n** deve ser lido e criticado.

P5.3) Construa um programa que leia um conjunto A de 50 elementos, construa e imprima outro conjunto B da seguinte forma:
-os elementos de índice par são correspondentes aos elementos de A divididos por 2 (A/2);
-os elementos de índice ímpar são correspondentes aos elementos de A multiplicados por 3 (3A).

P5.4) Escreva um programa que leia um conjunto A de 20 elementos numéricos, calcule e imprima o valor de S, onde:

$$S = (A_0 - A_{19})^2 + (A_1 - A_{18})^2 + ... + (A_9 - A_{10})^2$$

P5.5) A partir de uma tabela dada que registra a temperatura média de todos os dias de um ano, elabore um programa que calcule e escreva:

a) a maior e a menor temperatura ocorrida no ano;
b) a temperatura média anual;
c) o número de dias no ano em que a temperatura foi inferior à média anual (calculada em b);
d) o número de dias no ano em que a temperatura foi inferior ou igual a zero grau.

P5.6) Construa um programa capaz de calcular o faturamento mensal de 1000 mercadorias onde os preços unitários, as quantidades e os nomes dessas mercadorias são fornecidos um a um; imprimir o nome da mercadoria que obteve o maior total (R$) de vendas, o faturamento total mensal e o percentual que cada mercadoria (mostrar o nome) obteve sobre o faturamento total mensal.

P5.7) Dado um vetor VT de 70 elementos numéricos quaisquer, construa um programa que verifique se existe um elemento numérico qualquer Y no vetor. Se existir, escreva a(s) posição(ões) onde foi encontrado o elemento; se não, escreva “elemento não encontrado”. O vetor VT e o número Y são fornecidos pelo usuário.

P5.8) Faça um programa que leia dois vetores (A e B), contendo cada um, 25 elementos numéricos e intercale os elementos destes dois conjuntos formando um novo vetor (C) de 50 elementos da seguinte forma:

A

1800	225.9	667	.....	320	11	0.8
0	1	2		22	23	24

B

7	4	11	.....	3	1	8
0	1	2		22	23	24

C

1800	7	225.9	4	667	11	.	3	11	1	0.8	8
0	1	2	3	4	5		45	46	47	48	49

P5.9) Elabore um programa para calcular e escrever o desvio padrão de um vetor X de 50 elementos, de acordo com a fórmula:

$$D = \sqrt{\frac{\sum_{i=1}^{50}(x[i] - \bar{x})^2}{50 - 1}}$$

Onde $\bar{x}$ e a média aritmética dos elementos contidos no vetor X.

P5.10) Analise o programa abaixo e:

a) apresente os valores impressos;
b) mostre o conteúdo (quais os valores) da variável J do início ao fim do programa;
c) determine o número de repetições do laço 3.

```
int main ()          // programa VELOCIDADE
{
    int i, j, n;
    int x[5];

    n = 4;
    for (i = 0;  i <= n ; i = i + 1)            // laço 1
            x[i] = 1;
    i = 2;
    while (i < n)                               // laço 2
    {
            j = 0;
            while ((i + j) <= n)                // laço 3
            {
                    x[i+j] = 0;
                    j = j + 1;
            }
            i = i + 1;
            while ((x[i] == 0) && (i < n))  // laço 4
                    i = i + 1;
    }
    for (i = 0;  i <= n ; i = i + 1)            // laço 5
            if (x[i]  == 1)
                cout << "Posição com 1: " << i << endl;
    return 0;
}
```

P5.11) Faça um programa para corrigir provas de múltipla escolha. Cada prova tem 50 questões, cada questão valendo um ponto. O primeiro conjunto de dados a ser lido será o gabarito (de A a E) das 50 questões para a correção da prova. Os outros dados serão os 120 números de matrícula dos alunos e suas respectivas respostas às 50 questões. O programa deverá calcular e imprimir:

a) para cada aluno, o seu número de matrícula e sua nota;
b) a porcentagem de aprovação, cuja nota mínima de aprovação é 35.

P5.12) Escreva um programa que faça reserva de passagens aéreas de uma companhia. O número de vôos da companhia é igual a 37 (de 0 a 36). A quantidade de lugares disponíveis em cada um dos 37 vôos é fornecida. Vários pedidos de reserva são lidos, constituídos do

número da carteira de identidade do cliente e do número do vôo desejado (de 0 a 36). Para cada cliente, verificar se há disponibilidade no vôo desejado. Em caso afirmativo, imprimir o número da identidade do cliente e o número do vôo, atualizando o número de lugares disponíveis. Caso contrário, avisar ao cliente da inexistência de lugares.
Flag → número da carteira de identidade igual a 9999.

P5.13) Analise o programa abaixo:

```
int main ()              // programa PROVA
{
        int a[4], y , aux;
        bool b;

        a [0] = 37;
        a [1] = 22;
        a [2] = 49;
        a [3] = 24;
        b = true;
        while (b)                    // laço 1
        {
            b = false;
            for (y = 0; y <= 2; y = y + 1)          // laço 2
            {
                if (a[y] < a[y+1])
                  {
                      aux = a[y];
                      a[y] = a[y+1];
                      a[y+1] = aux;
                      b = true;
                  }
            }
        }
        for (y = 0; y <= 3; y = y + 1)
                cout << a[y] << endl;

        cout << "Y= " << y << "    B= " << b;
        return 0;
}
```

a) Apresente o que será impresso pelo programa quando executado.
b) Determine o número de repetições do LAÇO1.

P5.14) Escreva um programa que leia duas matrizes de elementos numéricos de dimensão 3 x 5, crie e escreva uma terceira com o resultado da soma das duas matrizes lidas.

P5.15) Faça um programa que carregue uma matriz 5x5 de elementos inteiros e verifique quantos elementos desta matriz são ímpares.

P5.16) Faça um programa que carregue uma matriz 2 x 4 com números inteiros, calcule e mostre:
-a quantidade de elementos entre 12 e 20 em cada linha;
-a média dos elementos pares da matriz.

P5.17) Faça um programa que leia os elementos de uma matriz 2x2, calcule e imprima o seu determinante.

P5.18) Faça um programa que carregue uma matriz 6 x 3, de elementos numéricos, calcule e mostre:

-o maior elemento da matriz e sua respectiva posição, ou seja, linha e coluna;
-o menor elemento da matriz e sua respectiva posição, ou seja, linha e coluna.

P5.19) Escreva um programa que leia uma matriz **X**x**Y** de elementos numéricos, onde ($X \leq 10 \, e \, Y \leq 10$), calcule e escreva o produto de todos os elementos da primeira coluna e da última linha desta matriz. O número de linhas (X) e o número de colunas (Y) da matriz são fornecidos pelo usuário. O elemento [X-1,0] não poderá ser multiplicado 2 vezes.

P5.20) Escreva um programa que leia uma matriz quadrada M de 20x20 elementos numéricos, gere e imprima uma 2a. matriz onde cada elemento desta nova matriz é o mesmo da matriz lida, multiplicado pelo elemento da diagonal principal da linha onde está o elemento, com exceção do próprio elemento da diagonal principal que não é modificado. (Elementos da diagonal principal: M[0,0], M[1,1], M[2,2],....., M[19,19]).

P5.21) Elabore um programa que:

a) leia um valor M natural, 1<M≤10;
b) leia uma matriz quadrada numérica A, de dimensão MxM;
c) verifique se a matriz é simétrica, ou seja, se A[i][j] = A[j][i], para todo i,j < M;
d) imprima a mensagem “simétrica”, se A for simétrica e “não simétrica”, caso contrário.

P5.22) Escreva um programa para a geração e impressão da seguinte matriz:

1	1	1	1	1	1
1	2	2	2	2	1
1	2	3	3	2	1
1	2	3	3	2	1
1	2	2	2	2	1
1	1	1	1	1	1

P5.23) Faça um programa que carregue uma matriz 6 x 4. Recalcule a matriz digitada, onde cada linha será multiplicada pelo maior elemento da linha em questão. Mostre a matriz resultante.

P5.24) Uma fábrica produziu dois tipos de motores M1 e M2 nos meses de janeiro, fevereiro, ... , dezembro e o número de motores produzidos foi registrado em uma matriz (12x2). O setor de produção tem uma tabela do valor gasto para se produzir cada motor (M1 e M2). Faça um programa que, a partir da produção mensal de motores M1 e M2, e seus respectivos custos, calcule o valor gasto em cada um dos meses e o gasto anual para se produzir os motores.

P5.25) Faça um programa que carregue uma matriz 12 x 4 com os valores das vendas de uma loja, onde cada linha representa um mês do ano e cada coluna representa uma semana do mês. Calcule e mostre:

-o total vendido em cada mês do ano, mostrando o nome do mês por extenso;
-o total vendido pela loja no ano.

P5.26) Faça um programa que leia uma variável composta bidimensional (27 x 10) cujo conteúdo é a população dos 10 municípios mais populosos de cada um dos 27 estados brasileiros. Determine e imprima o número do município mais populoso e o número do estado a que pertence. Considerando que a primeira coluna contém sempre a população da capital do estado, calcule e imprima a média da população das capitais dos 27 estados.

P5.27) A matriz abaixo, representa o pátio de uma indústria e contém um estoque de matéria prima. Cada um dos valores é uma combinação de números, que representam códigos de produtos e quantidades disponíveis no estoque.

Exemplo: 1072 107-quantidade 2-código do produto

Tipos de produto:1-cobre, 2-carvão, 3-chumbo, 4-ferro, 5-alumínio, 6-níquel.

7012	0000	0000	9824
2363	4595	0000	3015
2561	0000	5302	6671
0000	6546	9454	3003

Escreva um programa capaz de contar quantas toneladas de cada material existem no pátio. A saída das informações deverá ser:

PRODUTO	QUANTIDADE
cobre	xxxx
carvão	xxxx