Variáveis compostas.

Estruturas de dados homogêneas (Matrizes)

Definição de Matriz

- Matriz é uma variável composta homogênea multidimensional;
- Ela é formada por uma sequência de variáveis de mesmo tipo, que possuem o mesmo identificador (mesmo nome) e são alocadas sequencialmente na memória;
- Como as variáveis têm o mesmo nome, o que as distingue são índices que referenciam sua localização dentro da estrutura;
- Uma variável do tipo matriz precisa de um índice para cada uma das suas dimensões.

Estruturas multidimensionais (matrizes)

	0	1	2
0	a[0][0]	a[0][1]	a[0][2]
1	a[1][0]	a[1][1]	a[1][2]
2	a[2][0]	a[2][1]	a[2][2]

- Para identificar um elemento específico de uma tabela, precisamos especificar dois índices:
  - o primeiro (por convenção) identifica a linha do elemento
  - o segundo (por convenção) identifica a coluna do elemento
- As estruturas que exigem dois índices são chamadas de estruturas bidimensionais ou matrizes.

Estruturas multidimensionais (matrizes)

- As estruturas multidimensionais em C/C++ podem ter vários índices
- Um uso comum é representação das tabelas de valores que consistem em informações organizadas em linhas e colunas.
- A quantidade dos índices pode chegar até 12 (dependendo da versão da linguagem).

Matrizes

- Para criar uma matriz 2 x 2, i.e. uma matriz de 4 elementos do tipo int, podemos escrever:

```
int m[2][2];
```

- Para declarar e ao mesmo tempo inicializar (i.e. atribuir os valores iniciais) podemos escrever:

```
int m[2][2] = { { 1, 2}, { 3, 4} };
```

- Os valores são agrupados por linha e colocados entre chaves
- Se não houver inicializadores suficientes para uma determinada linha, os elementos restantes daquela linha são inicializados com 0

Matrizes

- Atribuição de valor:

```
m[0][0] = 5;
m[1][0] = 10;
```

- Impressão do elemento m[0][0]:

```
printf(" %i ", m[0][0] );
```

Matrizes

- Operações com elementos de uma matriz:

```
x = m[0][0] + 5;

m[0][0] = 5 – 10;
```

- Para calcular a soma dos valores contidos na primeira **linha** da matriz m podemos escrever:

```
sum_line = m[0][0] + m[0][1];
```

- Para calcular a soma dos valores contidos na primeira **coluna** da matriz m podemos escrever:

```
sum_column = m[0][0] + m[1][0];
```

Matrizes

- Exatamente como no caso dos vetores, devemos reservar um determinado espaço na memoria para armazenar uma matriz.
- Cuidado na hora de atribuir os valores!
- A linguagem em si não avisa quando o limite de uma matriz foi excedido.

Exemplo 1 (1p): Criação e inicialização de uma matriz 2x2

```c
// Exemplo 1: matrizes
#include <stdio.h>

int main()
{
  int i, j;
  int m[2][2] = { { 1, 2}, { 3, 4} };
  //int m[2][2] = { { 1, 2} };

  printf("\n Programa imprime uma matriz 2x2 com valores pre definidos: \n\n");
  for ( i=0; i<2; i++)
   {
     for ( j=0; j<2; j++)
          printf("%i   ", m[i][j] );
     printf("\n");
   }

  return 0;
}
```

Exemplo 2 (1p): Leitura de dados para uma matriz 2 x 2

```c
#include <stdio.h>

int main()
{
  const int line = 2, column = 2;
  int i, j;
  int m[line][column];
//-------------------------------------------------------------
 printf("\n Leitura de dados para matriz 2x2: \n\n");
 for ( i=0; i < line; i++)
   {
     for ( j=0; j < column; j++)
     {
       printf("\n Digite elemento [%i][%i] da matriz: ", i, j);
       scanf("%i", &m[i][j]);
     }
   }
//-------------------------------------------------------------
  printf("\n\n\n Matriz m: \n\n");
  for ( i=0; i < line; i++)
   {
     for ( j=0; j < column; j++)
          printf("%i   ", m[i][j] );
     printf("\n");
   }

  return 0;
}
```