UFRN
UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE

Linguagem de Programação

Arrays unidimensionais (vetores)

ECT2303

helton.maia@ect.ufrn.br

Arrays - Motivação

Considere a seguinte situação, você está fazendo uma pesquisa com 100 pessoas e precisa armazenar a idade de cada uma delas.

Como resolver este problema em C++?

Arrays - Motivação

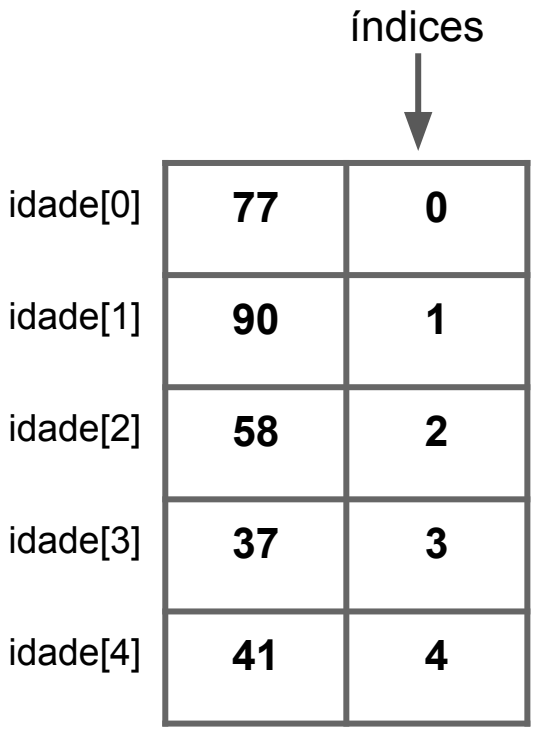
Como resolver este problema em C++?

Em vez de se declarar variáveis individuais, como idade0, idade1, ... e idade99, você pode declarar um array idade[ ], desta forma será possível guardar todas a idades na mesma variável.

Arrays - Introdução

- Todas as matrizes consistem em locais de memória contíguos. O endereço mais baixo corresponde ao primeiro elemento e o endereço mais alto ao último elemento;

- Um elemento específico em uma matriz é acessado por um índice. Exemplo com 5 elementos



Arrays - Definições

- ➔ Conjunto de espaços de memória que se relacionam. Definidos por um nome e um tipo comum entre eles;
- ➔ Para acessar um elemento da array, especificamos seu nome e a posição(índice) onde o elemento se encontra;
- ➔ Os espaços de memória são alocados linearmente. Desta forma, o número da posição serve para calcular o endereço de memória em que o elemento está localizado.

Arrays - Declaração e Inicialização

Sobre os índices:

- Este valor vai dimensionar o vetor e deve ser um inteiro ou mesmo uma expressão inteira equivalente;
- Constantes inteiras.

Declarando a array:

```
tipo identificador [tamanho];
```

Arrays - Declaração e Inicialização

```
#include ...
#define TAM 100
...
float v[TAM];
```

```
const int n = 100;
float v[n];
```

```
float v[100]; //constante inteira
```

Arrays - Declaração e Inicialização

```
tipo nome[tamanho];
```

- O dimensionamento (tamanho) do array deve ser uma constante inteira. O total de elementos contidos na array serve para informar ao compilador a quantidade de memória necessária que deve ser reservada;
- Atenção: Não é permitido utilizar variáveis, então evite a prática abaixo:

```
int num;
cin >> num;
float v[num];
```

Obs: Note que uma array é uma estrutura homogênea.

Arrays - Acessando elementos

dados

1.1	5.4	2.3	9.5	6.8	8.9	3.4	2.6	5.6	3.3
0	1	2	3	4	5	**6**	7	8	9

- Para acessar os elementos de vetor ou array é necessário especificar a sua posição (índice). Ex. **dados[6]**
- Lembre-se que o primeiro elemento de um vetor possui índice zero.

Arrays - Exemplos

```
#define TAM 100

int v[TAM];  // Declaração
v[4] = 10;   // Atribuição
cin >> v[5]; // Leitura de um valor
cout << v[50] + v[51]; //Imprimindo uma soma dos elementos
```

Arrays - Inicializando

```
//inicializa todos os elementos com o mesmo valor
int n[10] = { 0 };

//inicialização individual de cada posição do vetor
//quantidade entre chaves não pode ser maior o tamanho em []
double balance[5] = {1000.0, 2.0, 3.4, 17.0, 50.0};

//Caso seja omitido o tamanho do array, será criada uma array
//de forma a adequar todos os elementos a serem inicializados.
double balance[] = {1000.0, 2.0, 3.4, 17.0, 50.0};
```

Arrays - Consumo de memória

```
int tamanho = sizeof(<tipo_elementos>)*<tamanho_array>
```

Exemplo:

```
int vet1[5];
//sizeof(vet1) => 5 * 4 = 20

double vet2 [10];
//sizeof(vet2) => 8 * 10 = 80
```

Arrays - Em estruturas de repetição

```
int main () {
    int id , dados[10];

//Leitura de dados
for ( id = 0; id < 10; id ++) {
    cin >> dados [id];
}
//Imprimindo valores do vetor
for ( id = 0; id < 10; id ++) {
    cout << dados [id];
}
return 0;
}
```

Atenção: o C++ não verifica os limites da array. Tenha cuidado para não sobrepor dados da memória!

Arrays - Exercício 1

```cpp
#include <iostream>
#include <iomanip>
using namespace std;
using std::setw;

int main () {
   const int tam = 10;
   int n[tam]; //array possui 10 inteiros

   // inicializando elementos da array
   for ( int i = 0; i < tam; i++ ) n[i] = i + 100;
   cout << "Elementos" << setw( 13 ) << "Valor" << endl;

   // imprimindo os elementos da array
   for ( int j = 0; j < tam; j++ )
   cout << setw( 7 )<< j << setw( 13 ) << n[j] << endl;

   return 0;
}
```

O que acontece aqui?

Arrays - Exercício 2

Escreva um programa que armazena inicialmente 10 elementos. Depois, calcula o maior valor entre estes elementos e imprime seu índice.