ARRAYS

C++, CURSO COMPLETO

MÓDULO 03, AULA 4

Dr. Eng. Wagner Rambo
WR Kits

ARRAYS

- Um array consiste em um conjunto consecutivo de posições de memória que apresentam o mesmo tipo de dados.
- Os arrays são declarados seguindo as mesmas regras de nomenclatura de variáveis e funções, porém devem ser seguidos de colchetes [ ]. Dentro dos colchetes vai o número de elementos total do array.

```
int teste[10];  //declaração de um array de 10 elementos do tipo inteiro
```

- Para selecionarmos uma das posições de memória a ser lida ou escrita, utilizamos o índice do respectivo elemento, lembrando que arrays sempre começam com índice 0.

```
cout << teste[0] << endl; //imprime o conteúdo do primeiro elemento do array

teste[2] = 45; //escreve 45 na terceira posição do array
```

ARRAYS

- O índice de um array deve ser um inteiro ou qualquer expressão que utiliza qualquer tipo inteiro.
- Sendo uma variável x = 2 e y = 5:

```
teste[ x + y ] = 8; //armazena 8 na posição 7 do array
```

- Se tivermos 6, 9 e 4 armazenados nas posições 6, 8 e 9 de um array:

```
cout << teste[6]+teste[8]+teste[9] << endl; //imprimirá 19 na tela
```

DECLARAÇÃO DE UM ARRAY

- Para declarar um array utilize a seguinte sintaxe: tipo nomeArray [tamanhoArray];
- Veja alguns exemplos de declaração:

```
int array1[12]; //array de 12 inteiros declarado

short array2[3]; //array de 3 inteiros short

unsigned array3[8]; //array de 8 inteiros sem sinal
```

INICIALIZANDO ARRAYS

- O trecho de código a seguir declara um array chamado a com 10 elementos do tipo inteiro. Em seguida preenchemos todos com os valores de 1 a 10 com *for* e mandamos imprimir da na tela.

```
int a[10];

 for(int i=0; i<10; i++)
   a[i] = i+1;

 for(int i=0; i<10; i++)
   cout << setw(5) << "a[" << i << "]= "
        << setw(4) << a[i] << endl;
```

INICIALIZANDO ARRAYS

- O trecho de código a seguir declara um array chamado b com 6 elementos do tipo inteiro e já usamos uma lista de inicialização.

```
int b[6] = {4, 8, 15, 16, 23, 42};

 for(int i=0; i<6; i++)
   cout << setw(5) << "b[" << i << "]= "
        << setw(4) << b[i] << endl;
```

INICIALIZANDO ARRAYS

- Se o tamanho do array for maior que a lista, os demais elementos serão inicializados com 0 por padrão. O trecho a seguir exibe um array c com 9 elementos, mas inicializando-se apenas 3.

```
int c[9] = {7, 4, 1};

 for(int i=0; i<9; i++)
   cout << setw(5) << "c[" << i << "]= "
        << setw(4) << c[i] << endl;
```

- Porém, se o array não for inicializado com nenhum valor, ele conterá lixo de memória. O exemplo abaixo declara um array d sem inicializar. Serão impressos valores na tela desconhecidos, que denotam lixo de memória.

```
int d[5];

 for(int i=0; i<5; i++)
   cout << setw(5) << "d[" << i << "]= "
        << setw(4) << d[i] << endl;
```

INICIALIZANDO ARRAYS

- Se o tamanho do array for omitido na declaração, o número de elementos do mesmo será considerado de acordo com a lista de inicialização. Abaixo um array com nome e que foi inicializado com 4 elementos.

```
int e[ ] = {4, 0, 9, 3};

 for(int i=0; i<4; i++)
   cout << setw(5) << "e[" << i << "]= "
        << setw(4) << e[i] << endl;
```

- A declaração abaixo causará erro de compilação, pois o array f foi declarado com menos elementos que na lista de inicialização.

```
int f[3] = {7, 4, 3, 2};
```

VARIÁVEL CONSTANTE

- Com a palavra reservada *const* declaramos variáveis constantes. Estas variáveis devem ser inicializadas na declaração e não podem ser modificadas. Um exemplo de utilização é a determinação do tamanho de arrays.
- O trecho de código abaixo declara uma constante N que recebe o valor 4. Esse será o tamanho do array g declarado em seguida. Perceba que podemos usar N para imprimir os elementos do array com auxílio do *for*.

```
const int N = 4;

int g[N] = {8, 0, 5, 1};

 for(int i=0; i<N; i++)
   cout << setw(5) << "g[" << i << "]= "
        << setw(4) << g[i] << endl;
```

ARRAY DE CARACTERES

- Podemos ter arrays com quaisquer tipos de dados. Um array comum é o array de caracteres. Quando este apresenta em seu final um caractere nulo '\0', o mesmo consistirá em uma *string*.

```
char h[ ] = "WR Kits"; //string com 8 caracteres (devemos considerar o caractere nulo '\0'

char i[ ] = {'W','R','K','i','t','s','\0'}; //outra forma de inicializar

char j[ ] = {'W','R','K','i','t','s'}; //não é string, é apenas um array de caracteres
```

ATUALIZANDO A CLASSE SalesScore

- Utilizando arrays vamos trabalhar a nossa classe SalesScore para efetuar vários cálculos com um conjunto de valores que representam as vendas de produtos hipotéticos.

PESQUISA LINEAR COM ARRAYS

- A seguir, um exemplo de software que realiza a pesquisa linear em um *array*.

CLASSIFICAÇÃO EM ARRAYS

- Confira o software que organiza um *array* de 10 elementos em ordem crescente.

EXERCÍCIO

50 min.

- Crie um programa em C++ que peça ao usuário para entrar com 5 valores inteiros.
- Armazene os valores em um array e após organize os dados em ordem decrescente.
- Depois, entra em loop infinito que solicita ao usuário para consultar uma posição desejada do vetor e mostrar qual conteúdo está armazenado na posição digitada.
- Crie uma classe chamada “Decrescente” que contenha o construtor, uma função para organizar o array passado para ela em ordem decrescente e outra para efetuar a busca pelos conteúdos do array no loop infinito.

REFERÊNCIAS

- DAMAS, Luís. Linguagem C. 10.ed. LTC
- DEITEL. C++: Como Programar. 5.ed. Pearson
- Linguagem C, o Curso Definitivo WR Kits

CANAL DE SUPORTE

- CLIQUE AQUI e poste suas dúvidas no espaço de comentários.