ACH2001 - Introdução à Programação

Vetores (*arrays*)

Prof. Flávio Luiz Coutinho
flcoutinho@usp.br

Motivação

Escrever um programa que leia uma sequência de 3 valores inteiros da entrada padrão, e imprima sequência em ordem reversa à ordem de entrada.

Motivação

Escrever um programa que leia uma sequência de 3 valores inteiros da entrada padrão, e imprima sequência em ordem reversa à ordem de entrada.

Código

Motivação

Escrever um programa que leia uma sequência de 3 valores inteiros da entrada padrão, e imprima sequência em ordem reversa à ordem de entrada.

Código

Como fazer se desejamos ler 4 valores?

Motivação

Escrever um programa que leia uma sequência de 3 valores inteiros da entrada padrão, e imprima sequência em ordem reversa à ordem de entrada.

Código

Como fazer se desejamos ler 4 valores?

E para 5 valores?

Motivação

Escrever um programa que leia uma sequência de 3 valores inteiros da entrada padrão, e imprima sequência em ordem reversa à ordem de entrada.

Código

Como fazer se desejamos ler 4 valores?

E para 5 valores?

E para 10, 100, ou 1000 valores?

Motivação

Escrever um programa que leia uma sequência de 3 valores inteiros da entrada padrão, e imprima sequência em ordem reversa à ordem de entrada.

Código

Como fazer se desejamos ler 4 valores?

E para 5 valores?

E para 10, 100, ou 1000 valores?

E se a quantidade especificar apenas o máximo de valores que podem ser lidos?

Vetores ou *arrays*

Um vetor, ou *array*, é um tipo de variável capaz de armazenar um conjunto de valores de um mesmo tipo.

Vetores ou *arrays*

Um vetor, ou *array*, é um tipo de variável capaz de armazenar um conjunto de valores de um mesmo tipo.

```
int a[5]; // declara um array de 5 posições do tipo int
```

Vetores ou *arrays*

Um vetor, ou *array*, é um tipo de variável capaz de armazenar um conjunto de valores de um mesmo tipo.

```
int a[5]; // declara um array de 5 posições do tipo int
```

É capaz de armazenar 5 valores inteiros.

Vetores ou *arrays*

Um vetor, ou *array*, é um tipo de variável capaz de armazenar um conjunto de valores de um mesmo tipo.

```
int a[5]; // declara um array de 5 posições do tipo int
```

É capaz de armazenar 5 valores inteiros.

Cada um destes 5 valores é acessado através de um índice.

Vetores ou *arrays*

Um vetor, ou *array*, é um tipo de variável capaz de armazenar um conjunto de valores de um mesmo tipo.

```
int a[5]; // declara um array de 5 posições do tipo int
```

É capaz de armazenar 5 valores inteiros.

Cada um destes 5 valores é acessado através de um índice.

No caso de 5 posições, os índices vão de 0 a 4.

Vetores ou *arrays*

```
int a[5]; // declara um array de 5 posições do tipo int


a[0]: valor na primeira posição

a[1]: valor na segunda posição

a[2]: valor na terceira posição

a[3]: valor na quarta posição

a[4]: valor na quinta posição
```

Vetores ou *arrays*

Oferece uma forma muito mais prática de lidar com um grande volume de dados do mesmo tipo.

Vetores ou *arrays*

Oferece uma forma muito mais prática de lidar com um grande volume de dados do mesmo tipo.

Voltando ao nosso problema usado como motivação:

Vetores ou *arrays*

Oferece uma forma muito mais prática de lidar com um grande volume de dados do mesmo tipo.

Voltando ao nosso problema usado como motivação: até é possível solucioná-lo sem o uso usar vetores.

Vetores ou *arrays*

Oferece uma forma muito mais prática de lidar com um grande volume de dados do mesmo tipo.

Voltando ao nosso problema usado como motivação: até é possível solucioná-lo sem o uso usar vetores. Mas quanto maior a quantidade de valores na sequência, maior e mais complicado fica o programa.

Vetores ou *arrays*

Oferece uma forma muito mais prática de lidar com um grande volume de dados do mesmo tipo.

Voltando ao nosso problema usado como motivação: até é possível solucioná-lo sem o uso usar vetores. Mas quanto maior a quantidade de valores na sequência, maior e mais complicado fica o programa.

Código usando vetores.

Vetores ou *arrays*

```
int a;          // espaço de memória suficiente para guardar um valor inteiro
a = 10;         // guarda o valor 10 no espaço de memória da variável  a
x = a + 1;      // usa o valor armazenado no espaço de memória associado à
                // variável a em uma expressão aritmética.

int a[5];       // espaço suficiente para guardar 5 valores inteiros.
                // variável a contém endereço do início deste espaço .
a = 10;         // "tentativa" de redefinição do endereço guardado em  a.
a[0] = 10;      // atribuição do valor 10 logo no início do espaço.
x = a[0] + 1; // uso do valor armazenado no início do espaço de memória.
```

Vetores ou *arrays*

```
a[0]:  valor armazenado no endereço (a + 0 * sizeof(int))

a[1]:  valor armazenado no endereço (a + 1 * sizeof(int))

...

a[4]:  valor armazenado no endereço (a + 4 * sizeof(int))
```

```
a:     valor em a representa um endereço de memória

a[i]:  a[i] é o valor no endereço (a + i * sizeof(int))
```