Laboratório de Programação

Prof. Dr. Paulo Rogério de Almeida Ribeiro

Coordenação do Curso de Engenharia da Computação

Ponteiros

Endereço (&) e valor de variável

```c
#include <stdio.h>

int main()
{
int num1=10, num2=20;

printf(" Num 1 - valor: %d", num1);
printf("\n Num 1 - Endereco memoria: %ld", (long int) &num1);

printf("\n Num 2 - valor: %d", num2);
printf("\n Num 2 - Endereco memoria: %ld\n", (long int) &num2);

return 0;
}
```

Ponteiro

- Tipo de variável;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Tipo de variável;
- Armazena um endereço;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Tipo de variável;
- Armazena um endereço;
- Ponteiro p armazena o endereço de uma variável i;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Tipo de variável;
- Armazena um endereço;
- Ponteiro p armazena o endereço de uma variável i;
- p aponta para i;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Tipo de variável;
- Armazena um endereço;
- Ponteiro p armazena o endereço de uma variável i;
- p aponta para i;
- p é o endereço de i;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Tipo de variável;
- Armazena um endereço;
- Ponteiro p armazena o endereço de uma variável i;
- p aponta para i;
- p é o endereço de i;
- p é uma referência a variável i;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Tipo de variável;
- Armazena um endereço;
- Ponteiro p armazena o endereço de uma variável i;
- p aponta para i;
- p é o endereço de i;
- p é uma referência a variável i;
- *p é o valor da variável apontada por p;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Tipo de variável;
- Armazena um endereço;
- Ponteiro p armazena o endereço de uma variável i;
- p aponta para i;
- p é o endereço de i;
- p é uma referência a variável i;
- *p é o valor da variável apontada por p;
- Para inicializar um ponteiro usamos NULL.

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro p, armazenado no endereço 60001
Contém o endereço (89422) de uma variável inteira com valor -9999

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Declaração: tipo_dado *nome_ponteiro;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Declaração: tipo_dado *nome_ponteiro;
- Exemplo:
  - int *p;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Declaração: tipo_dado *nome_ponteiro;
- Exemplo:
  - int *p;
  - float *ponteiro;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Ponteiro

- Declaração: tipo_dado *nome_ponteiro;
- Exemplo:
  - int *p;
  - float *ponteiro;
  - char *pont;

Fonte: https://www.ime.usp.br/~pf/algoritmos/aulas/pont.html

Tamanho dos tipos: char, int, float e double
Função sizeof()

```c
#include <stdio.h>

int main()
{

printf("Tamanho em bytes do char %ld\n", sizeof(char));
printf("Tamanho em bytes do int %ld\n", sizeof(int));
printf("Tamanho em bytes do float %ld\n", sizeof(float));
printf("Tamanho em bytes do double %ld\n", sizeof(double));

return 0;
}

```

Operador endereço (&)

```c
#include <stdio.h>

int main()
{

int m;
int * p;
p = &m;

printf("Endereco da variavel m %ld e %ld \n", (long int) &m, (long int)  p);

return 0;
}

```

Operador conteúdo (*)

```c
#include <stdio.h>

int main()
{

int m;
int *p;

p = &m;
m = 3;

//Sem ponteiro
printf("Endereco da variavel m %ld e seu valor %d \n", (long int) &m, m);
//Com ponteiro
printf("Endereco da variavel m %ld e seu valor %d \n", (long int)  p, *p);

return 0;
}

```

Quando usar ponteiro?

Passagem por valor e por referência

Quando usar ponteiro?

Passagem por valor e por referência

```c
#include <stdio.h>

void AlteraValor(int num);

int main()
{
int m = 3;
printf("Valor de m %d \n", m);
AlteraValor(m);
printf("Valor de m %d \n", m);
return 0;
}

void AlteraValor(int num)
{
  num=2;
  printf("Valor de m na funcao %d \n", num);
}
```

Quando usar ponteiro?

Passagem por valor e por referência

Quando usar ponteiro?

Passagem por valor e por referência

```c
#include <stdio.h>

void AlteraValor(int *num);

int main()
{
int m = 3;
int *ponteiro;
ponteiro=&m;
printf("Valor de m %d \n", m);
AlteraValor(ponteiro);
printf("Valor de m %d \n", m);
return 0;
}

void AlteraValor(int *num)
{
  *num=2;
  printf("Valor de m na funcao %d \n", *num);
}
```

Lembre-se:

int *p

p = &m; => **p aponta para o endereço de m**

m = 3;

p; => **imprime o endereço de m**

***p;** => **imprime o conteúdo de m (3 nesse caso)**

Relação ponteiro e arranjo

- Relação entre ponteiros e arranjos (vetores, matrizes e strings);

Relação ponteiro e arranjo

- Relação entre ponteiros e arranjos (vetores, matrizes e strings);
- O nome de um arranjo = ponteiro para o primeiro elemento do arranjo;

Relação ponteiro e arranjo

- Relação entre ponteiros e arranjos (vetores, matrizes e strings);
- O nome de um arranjo = ponteiro para o primeiro elemento do arranjo;

- **Exemplo:**

```
char c [10];
char * ch;
ch=c; // ch aponta para o primeiro elemento, ou seja, &c[0]
```

Aritmética de Ponteiros

- Duas operações aritméticas: adição e subtração;

Aritmética de Ponteiros

- Duas operações aritméticas: adição e subtração;
- Incremento: próxima posição;

Aritmética de Ponteiros

- Duas operações aritméticas: adição e subtração;
- Incremento: próxima posição;
- Decremento: posição anterior;

Aritmética de Ponteiros

- Duas operações aritméticas: adição e subtração;
- Incremento: próxima posição;
- Decremento: posição anterior;
- Incremento/decremento: mudança de endereço;

Aritmética de Ponteiros

- Duas operações aritméticas: adição e subtração;
- Incremento: próxima posição;
- Decremento: posição anterior;
- Incremento/decremento: mudança de endereço;
- Exemplos:
  - ptr + 1: endereço de onde ptr apontava + sizeof(tipo);

Aritmética de Ponteiros

- Duas operações aritméticas: adição e subtração;
- Incremento: próxima posição;
- Decremento: posição anterior;
- Incremento/decremento: mudança de endereço;
- Exemplos:
  - ptr + 1: endereço de onde ptr apontava + sizeof(tipo);
  - ptr + 2: endereço de onde ptr apontava deslocado em dois;

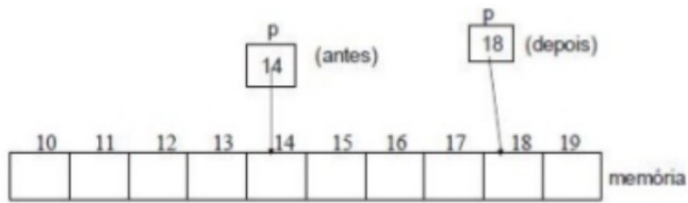
Aritmética de Ponteiros

- Duas operações aritméticas: adição e subtração;
- Incremento: próxima posição;
- Decremento: posição anterior;
- Incremento/decremento: mudança de endereço;
- Exemplos:
  - ptr + 1: endereço de onde ptr apontava + sizeof(tipo);
  - ptr + 2: endereço de onde ptr apontava deslocado em dois;
  - ptr – 3: endereço de onde ptr apontava, deslocado de três (trás).

Exemplo aritmética de ponteiros: float (4 bytes)

- Exemplo incremento:

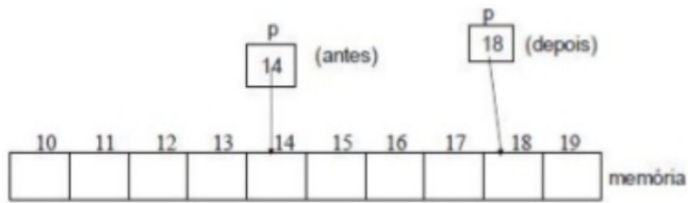
```
float *p
p++ ou ++p (= somar 1)
```



Exemplo aritmética de ponteiros: float (4 bytes)

- Exemplo incremento:

```
float *p
p++ ou ++p (= somar 1)
```

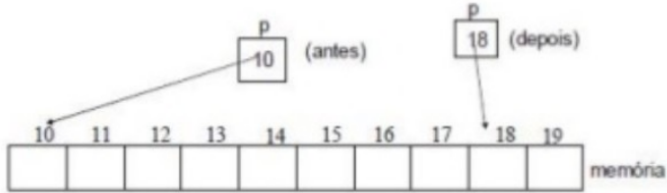


Endereço de onde ptr apontava + sizeof(tipo)

Exemplo aritmética de ponteiros: float (4 bytes)

- Exemplo adição:

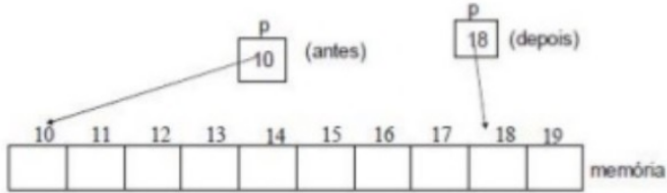
```
float *p
p=p+2
```



Exemplo aritmética de ponteiros: float (4 bytes)

- Exemplo adição:

```
float *p
p=p+2
```

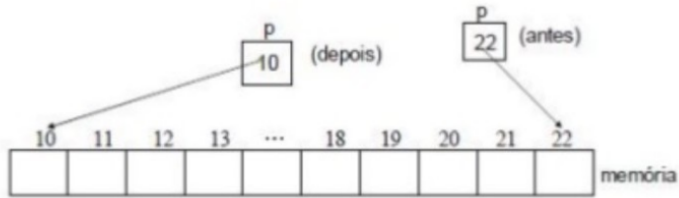


Endereço de onde ptr apontava + sizeof(tipo)

Exemplo aritmética de ponteiros: float (4 bytes)

- Exemplo subtração:

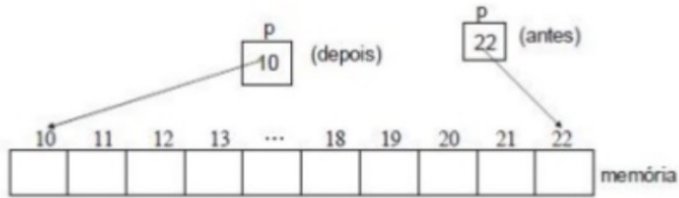
```
float *p
p=p - 3
```



Exemplo aritmética de ponteiros: float (4 bytes)

- Exemplo subtração:

```
float *p
p=p - 3
```



Endereço de onde ptr apontava - sizeof(tipo)

Exemplo aritmética de ponteiros

- Exemplo:

```
char nome[20], *pstr;
int val[10], *ptr;

pstr = nome;    // Equivalente a pstr = &nome[0]
ptr = val;    // Equivalente a ptr = &val[0]
pstr = nome + 4; // Equivalente a pstr = &nome[4]
ptr = val + 5; // Equivalente a ptr = &val[5]
```

Exemplo aritmética de ponteiros com arranjos

```c
#include <stdio.h>

const int N=5;

int main()
{
int *ponteiro=NULL;
int vetor[N], i;

//Ponteiro aponta para primeira posicao do vetor
ponteiro=vetor;
//ponteiro=&vetor[0];

for (i=0; i<N; i++)
{
  scanf("%d", ponteiro+i);
  //scanf("%d", &vetor[i]);
  //scanf("%d", ponteiro+i);
}

printf("Mostrar valores:\n");

for (i=0; i<N; i++)
{
  printf("%d\n", vetor[i]);
  //printf("%d\n", *(ponteiro+i));
  //printf("%d\n", *(ponteiro+i));
}

return 0;
}
```

Exercícios

1) Escreva um programa que receba uma string de caracteres via teclado e percorra essa string usando aritmética de ponteiros.

Exercícios

1) Escreva um programa que receba uma string de caracteres via teclado e percorra essa string usando aritmética de ponteiros.

2) Implemente uma função troca() que recebe dois números e troca-os. Mostre os valores antes da chamada a função, bem como depois.