Algoritmos e programação II

Prof. Viviane Bonadia dos Santos

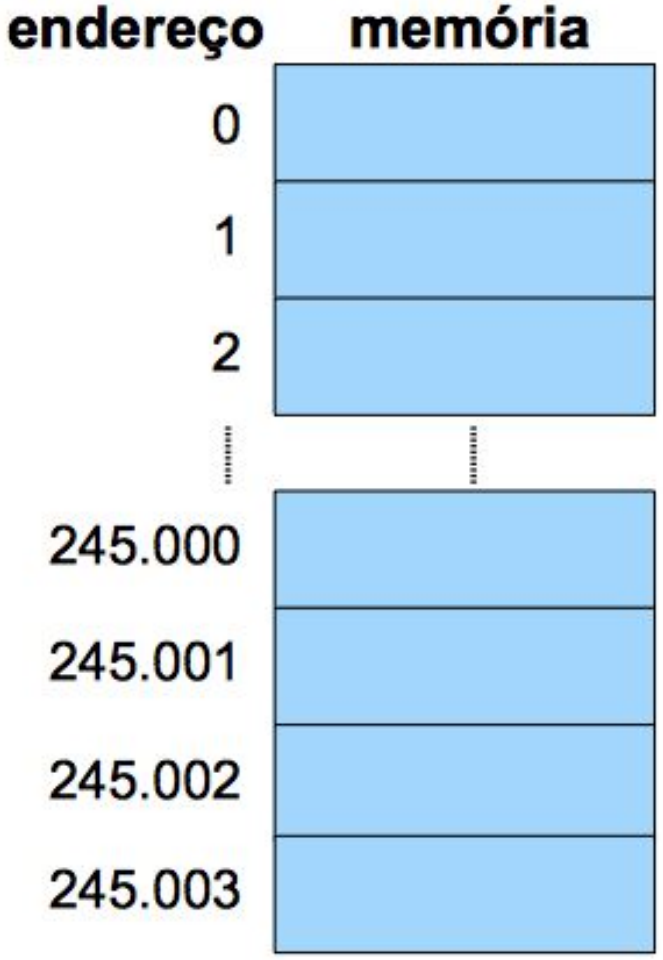
Relembrando...

O QUE SÃO PONTEIROS VARIÁVEIS??

O QUE SÃO PONTEIROS VARIÁVEIS??

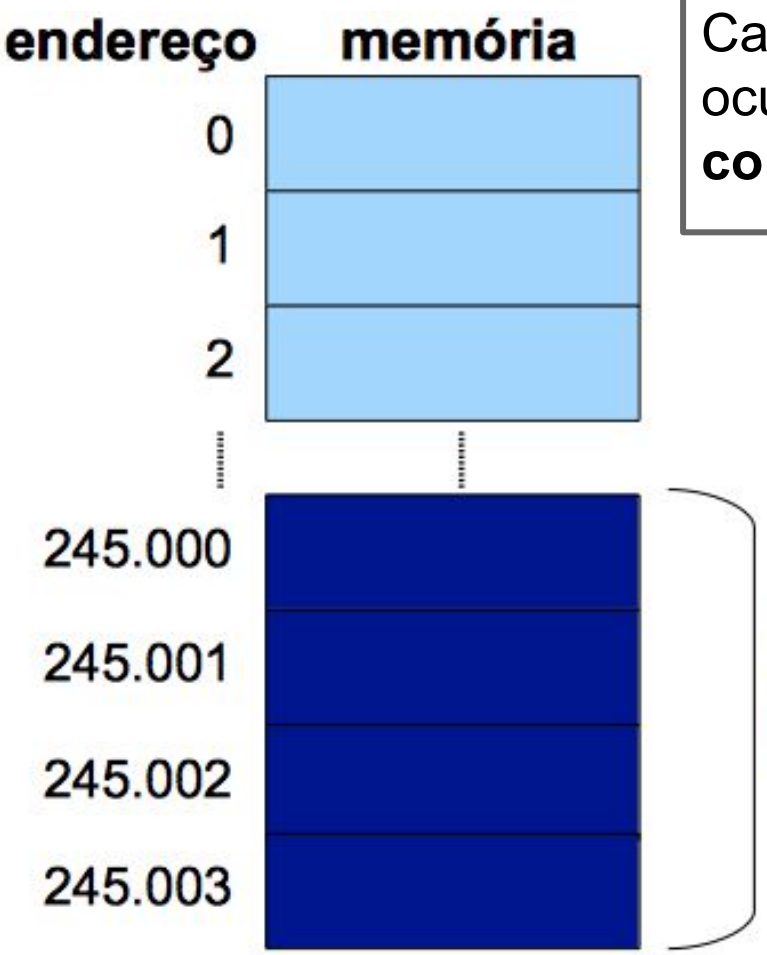
"Um ponteiro (ou apontador) é um tipo especial de variável que armazena o endereço (localização) de outra variável."

Endereços de memória



- A **memória** de qualquer computador é uma **sequência de bytes.**
- Cada byte armazena um de 256 possíveis valores. Os bytes são **numerados sequencialmente.**
- O número de um byte é o seu **endereço** (address).

Endereços de memória

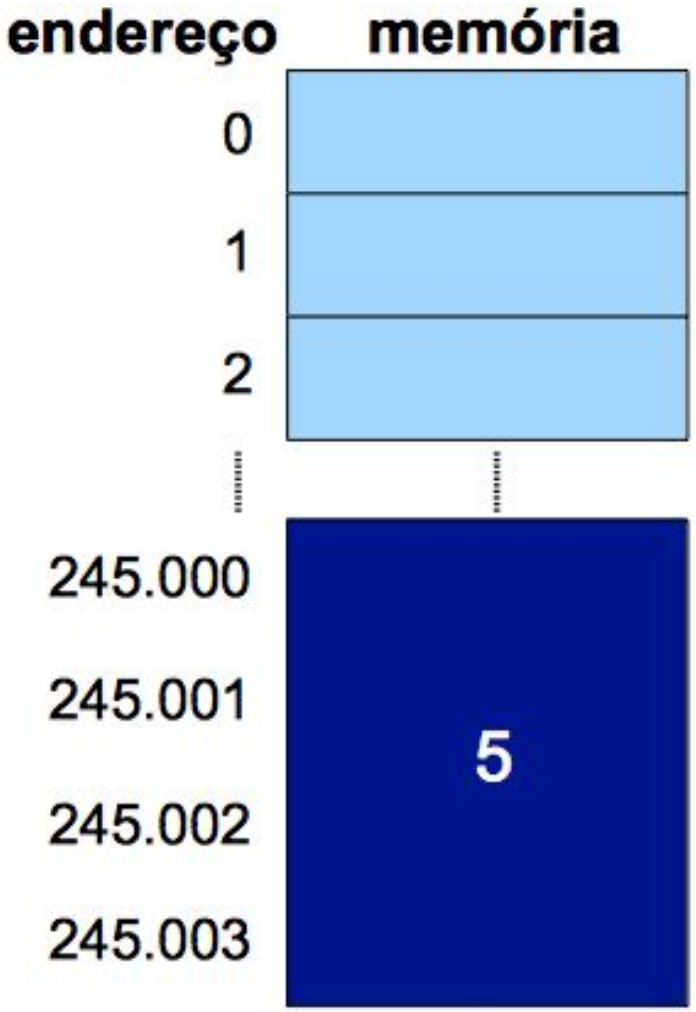


Cada objeto na memória do computador ocupa um certo número de bytes **consecutivos**.

```
int i;
```

4 bytes reservados para `i` que inicia no endereço 245.000

Endereços de memória




```
i = 5;
```

Cada objeto na memória do computador tem um endereço. Na maioria dos casos, **o endereço de um objeto é o endereço do seu primeiro byte**.

Ponteiros armazenam endereços de variáveis. Ok.

Como podemos criar um ponteiro e armazenar nele o endereço de uma variável??

Ponteiros

- Assim como variáveis comuns, ponteiros devem ser declarados. A única diferença é que ponteiros são identificados por um *

Declarando um ponteiro:

```
int *p;
```

- Para usar p, **primeiro é preciso inicializá-lo**. Uma forma de fazer isso é associar o endereço de uma variável.

```
int i = 5;
int *p;
p = &i;
```

Ponteiros

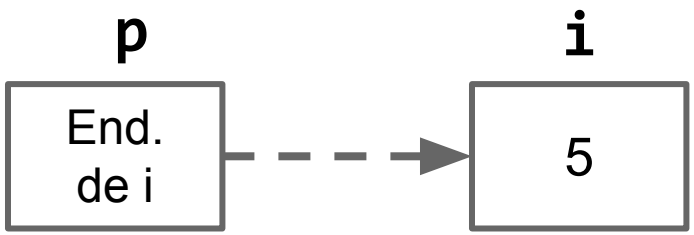
- Assim como
  ser declar
  ponteiro

Declarand
```
int *p;
```

- Para usar p,
  Uma forma de fa
  de uma variáve

```
int i = 5;
int *p;
p = &i;
```

& é o operador **endereço**. Se **i** é a variável, então **&i** é o endereço de memória de **i**.



O operador *

- Uma vez que o ponteiro aponta para um objeto, é possível usar o operador * para acessar o seu conteúdo!

```
printf("%d\n", *p); //mostra o valor de i
```

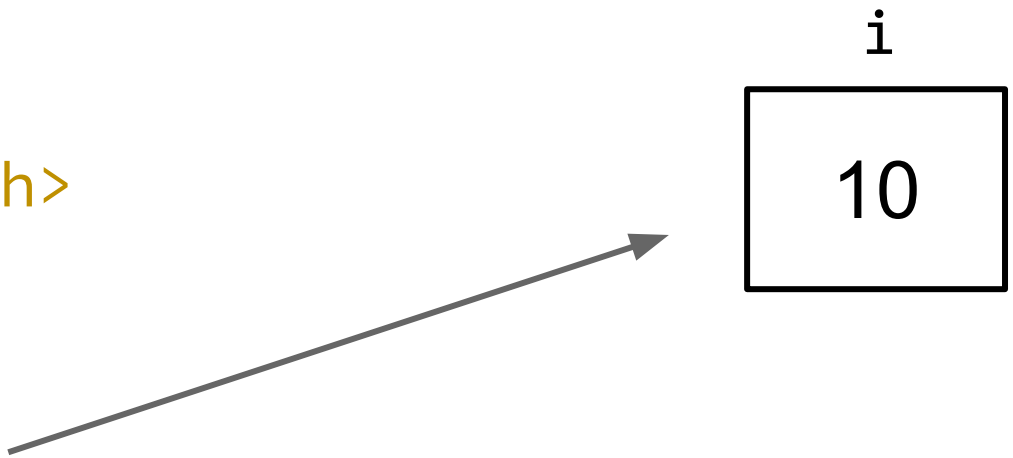
- Se **p** aponta para **i**, ***p** tem o **mesmo valor** de **i**. Alterar ***p** também altera o valor de **i**!!!

O operador *

```c
#include<stdio.h>
void main(){
   int i = 10;
   int *p = NULL;
   p = &i;
   *p = 5;
}
```

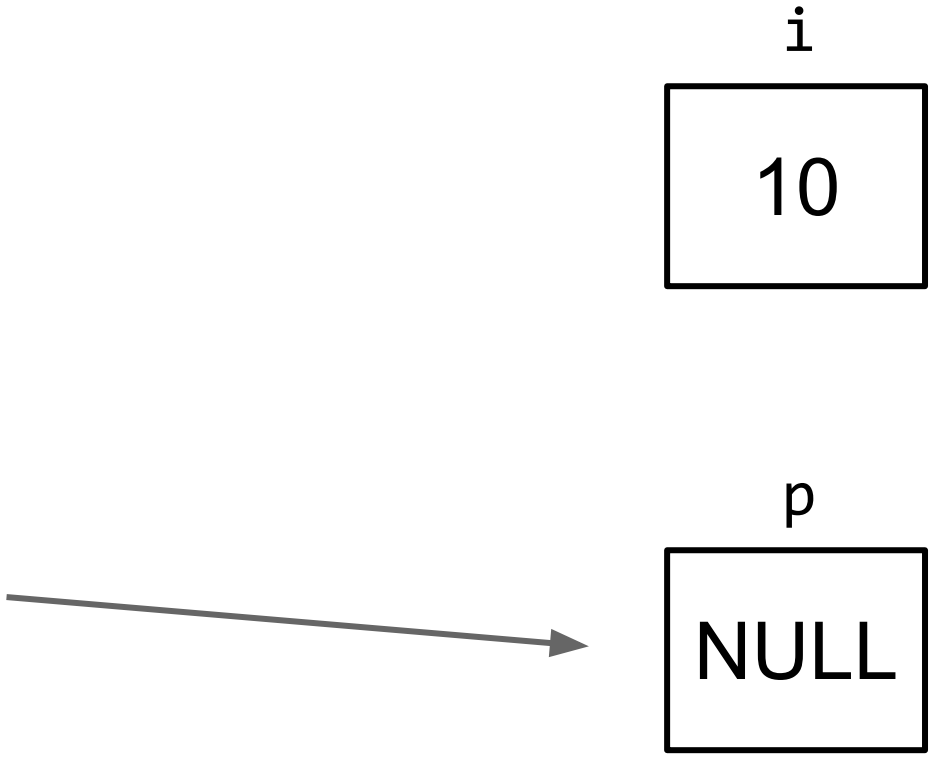
O operador *

```c
#include<stdio.h>

void main(){

    int i = 10;

    int *p = NULL;

    p = &i;

    *p = 5;

}
```



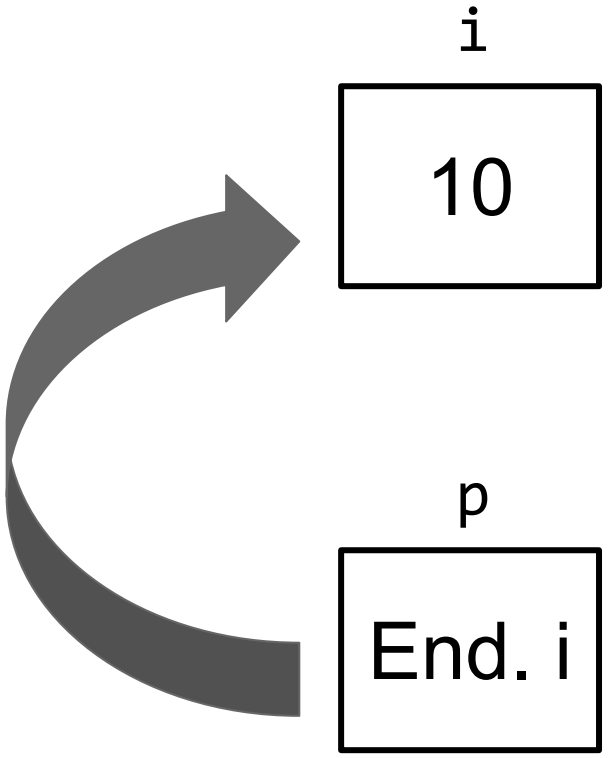
O operador *

```c
#include<stdio.h>

void main(){

    int i = 10;

    int *p = NULL;

    p = &i;

    *p = 5;

}
```



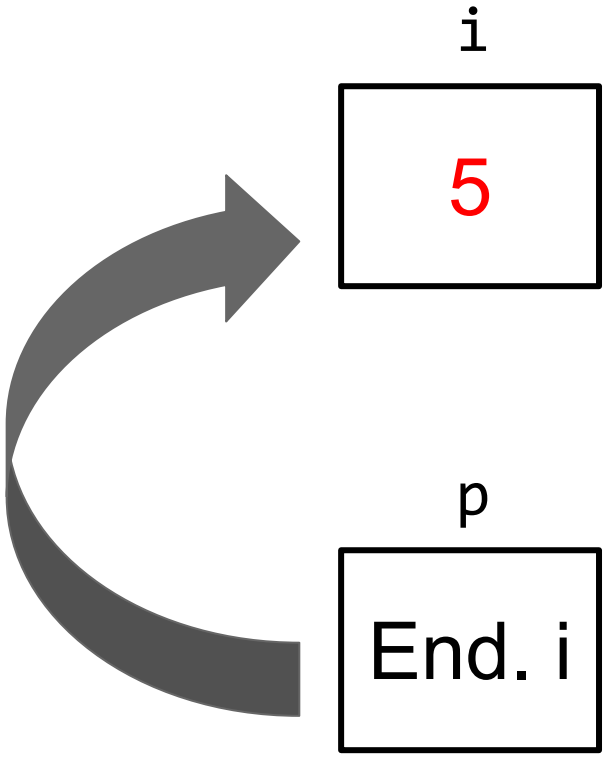
O operador *

```c
#include<stdio.h>
void main(){
    int i = 10;
    int *p = NULL;
    p = &i;
    *p = 5;
}
```



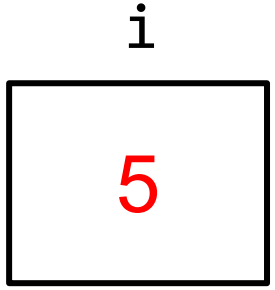
O operador *

```c
#include<stdio.h>
void main(){
    int i = 10;
    int *p = NULL;
    p = &i;
    *p = 5;
}
```



O operador *

```
#in
voi
    int
    p = &i;
    *p = 5;
}
```



Alterar ***p** também altera o valor de i!!!

p

Por que isso é muito legal??

Nossa vida sem ponteiros =/

```c
#include<stdio.h>
void troca(int a, int b){
   int aux = a;
   a = b;
   b = aux;
}

void main(){
   int num1 = 10;
   int num2 = 3;
   troca(num1, num2);
}
```

Nossa vida SEM ponteiros =/

```c
#include<stdio.h>
void troca(int a, int b){
   int aux = a;
   a = b;
   b = aux;
}

void main(){
   int num1 = 10;
   int num2 = 3;
   troca(num1, num2);
}
```

Antes de executar o método **troca**, **num1** armazena o número **10** e **num2** armazena o número **3**.

Depois de executar o método **troca**, **num1** armazena o número **10** e **num2** armazena o número **3**. **NADA mudou!**

Nossa vida com ponteiros :)

```c
#include<stdio.h>
void troca(int *a, int *b){
   int aux = *a;
   *a = *b;
   *b = aux;
}

void main(){
   int num1 = 10;
   int num2 = 3;
   troca(&num1, &num2);
}
```

Nossa vida com ponteiros :)

```c
#include<stdio.h>
void troca(int *a, int *b){
   int aux = *a;
   *a = *b;
   *b = aux;
}

void main(){
   int num1 = 10;
   int num2 = 3;
   troca(&num1, &num2);
}
```

Antes de executar o método **troca**, **num1** armazena o número **10** e **num2** armazena o número **3**.

Depois de executar o método **troca**, **num1** armazena o número **3** e **num2** armazena o número **10**. **O método de fato altera os valores armazenados pelas variáveis!!!**