INTRODUÇÃO A PONTEIROS

Professor: Francisco Dantas Nobre Neto
E-mail: dantas.nobre@ifpb.edu.br

Agenda

- Introdução
- Operações com ponteiros
  - Declaração de ponteiros;
  - Operadores de ponteiros;
  - Aritmética de ponteiros.
- Utilizando ponteiros com funções
- Bugs comuns com ponteiros

Introdução

- Variável do tipo “int” guarda um inteiro;
- Variável do tipo “float” guarda um número real;
- Ponteiro guarda um endereço de memória;
- É por meio de ponteiros que é possível armazenar e manipular endereços de memória:
  - Para cada tipo de variável (int, float, etc), existe um ponteiro associado.
- Com ponteiros, é possível definir o tamanho de um vetor em tempo de execução.

Introdução

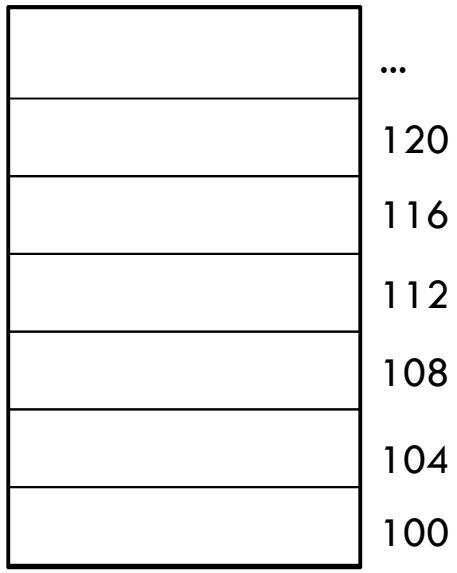
- Para se declarar um ponteiro:
  - *tipo *variavel* ou *tipo* variavel*.
- Operadores de ponteiros:
  - Operador unário * (lê-se “conteúdo apontado por”);
  - Operador unário & (lê-se “endereço de”).
- Aritmética de ponteiros:
  - Adição, subtração e atribuição.

Declaração de ponteiros

- Vamos acompanhar a execução das declarações abaixo.

```
/* variável inteiro */
int x;


/* variável ponteiro para inteiro */
int *p;
```



Pilha de execução

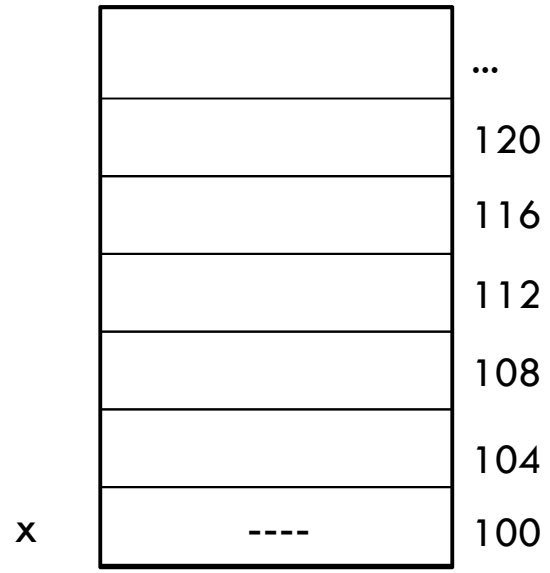
Declaração de ponteiros

- Declaração da variável “x”.

```
/* variável inteiro */
int x;

/* variável ponteiro para inteiro */
int *p;

...
```



Pilha de execução

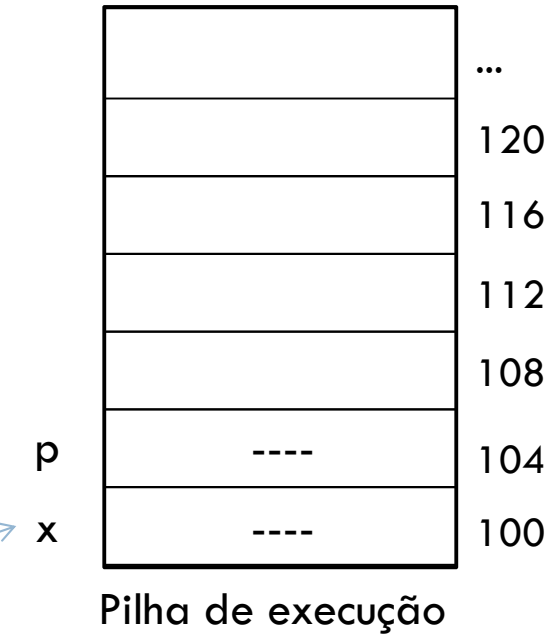
Declaração ponteiros

- Declaração do ponteiro do tipo inteiro “p”.

```
/* variável inteiro */
int x;


/* variável ponteiro para inteiro */
int *p;

...
```



Ambas variáveis possuem valores “lixo”, pois não foram inicializadas!!!

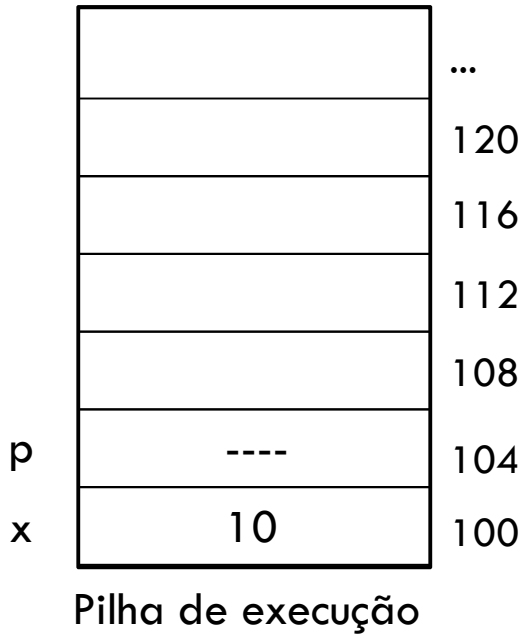
Atribuição de ponteiros

- Atribuição de valor à variável “x”.

```
/* x recebe o valor 10 */
x = 10;


/* p recebe o endereço de x */
p = &x;


/* conteúdo de p recebe o valor 30 */
*p = 30;
```



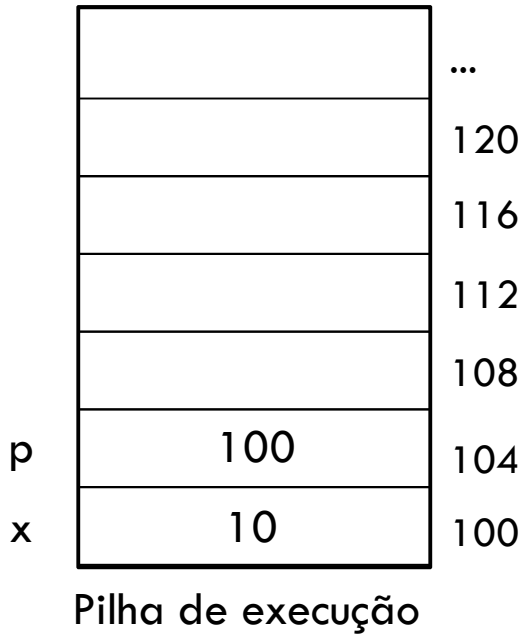
Atribuição de ponteiros

- Atribuindo endereço da variável “x” à variável “p”.

```
/* x recebe o valor 10 */
x = 10;


/* p recebe o endereço de x */
p = &x;


/* conteúdo de p recebe o valor 30 */
*p = 30;
```



Pilha de execução

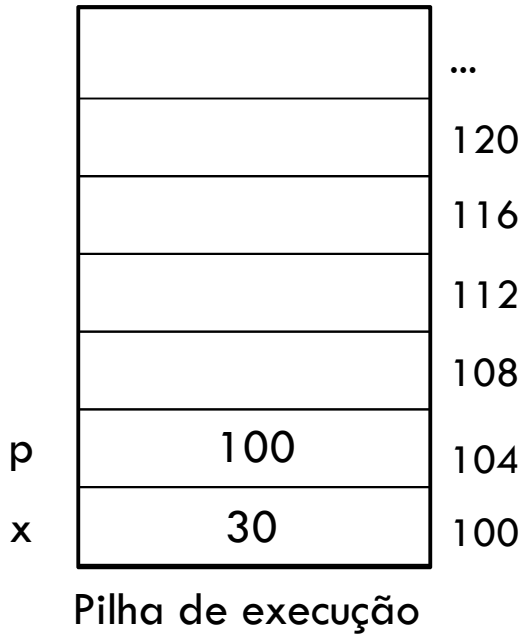
Atribuição de ponteiros

- Alterando o conteúdo **apontado** por "p".

```
/* x recebe o valor 10 */
x = 10;


/* p recebe o endereço de x */
p = &x;


/* conteúdo de p recebe o valor 30 */
*p = 30;
```



Aritmética de ponteiros

- Informe o que ocorre em cada caso:
  - *p = *p + 50 ?
  - *p = x – 10 ?
  - p = &x + 30 ?
  - p = 50 + 30 ?
  - p = x ?

Utilizando ponteiro em funções

- Ponteiros permitem alterar variáveis por acesso indireto (pelo endereço de memória);
- As funções não podem alterar diretamente um valor de uma variável;
- Resultado:
  - Utiliza-se ponteiros para alterar valores passados em uma função.

Utilizando ponteiros em funções

- Como solicitar que duas variáveis enviadas a uma função tenha seus valores permutados???
- É possível fazer isso sem ponteiros???
- Devemos usar qual definição de função? Por quê?

Utilizando ponteiros em função

- A função abaixo não irá funcionar:

```
void troca(int x, int y){
    int temp = x;
    x = y;
    y = temp;
}
```

Não é possível alterar o valor das variáveis diretamente na função!!!

- A função abaixo funcionará:

```
void troca(int *x, int *y){
    int temp = *x;
    *x = *y;
    *y = temp;
}
```

Os valores serão alterados indiretamente, por meio de ponteiros!!!

Bugs comuns em ponteiros

- Ponteiro não inicializado:

```
int *p;
*p = 10;
```

- Referências de ponteiros inválidas:

```
int *p, *q;
p = q;
```

- Referências de ponteiros:

```
int *p;
p = 0;
*p = 12;
```

Exercícios

- Explique a diferença entre os códigos abaixo:
  - p++;
  - (*p)++;
  - *(p++).
- Faça uma função que receba um vetor e dois números como parâmetro:
  - *void permuta_col(int *vetor, int col1, int col2);*
  - A função deverá permutar as duas colunas informadas como parâmetro;
  - Exemplo: [1, 4, 3, 5] ←→ [3, 4, 1, 5]
    - As colunas 1 e 3 foram alteradas no vetor.

Exercícios

- Qual o valor de y no final do programa?

```
int main()
{
    int y, *p, x;
    y = 0;
    p = &y;
    x = *p;
    x = 4;
    (*p)++;
    x--;
    (*p) += x;
    printf ("y = %d\n", y);
    return(0);
}
```

Bibliografia

- Como tudo funciona – Programação em C
  - http://informatica.hsw.uol.com.br/programacao-em-c.htm

- Waldemar Celes; Renato Cerqueira; José Lucas Rangel. **Introdução a Estruturas de Dados.** 2004.