

www.facom.ufu.br/~renan/classes/ipc



Introdução à Programação de Computadores
GSI002 - 2012/2
Profs. Walter de Oliveira / Renan Cattelan

Prática 10

Funções & ponteiros

Ponteiros

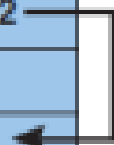
- ❑ Variável: espaço reservado de memória usado para guardar um valor

```
int count = 10;
```

- ❑ Ponteiro: espaço reservado de memória usado para guardar o endereço de memória de uma outra variável

```
int* p = &count;
```

Memória		
#	var	conteúdo
119		
120	int *p	#122
121		
122	int count	10
123		



Lista 9

18. Escreva um programa que contenha duas variáveis inteiras. Compare seus endereços e exiba o maior endereço.

```
#include<stdio.h>
```

```
int main(void)
```

```
{
```

```
    int x, y;
```

```
    x = 10;
```

```
    y = 20;
```

```
    if (&x > &y)
```

```
        printf("Endereco maior eh de x: %d.\n", &x);
```

```
    else
```

```
        printf("Endereco maior eh de y: %d.\n", &y);
```

```
    system("pause");
```

```
    return 0;
```

```
}
```

Ponteiros

- Já vimos ponteiros antes, na passagem de parâmetros por referência
 - Referência = endereço na memória
 - Referência = ponteiro

Lista 9

12) Implemente uma função que calcule a área da superfície e o volume de uma esfera de raio R . Essa função deve obedecer ao protótipo:

```
void calc_esfera(float R, float *area, float *volume)
```

A área da superfície e o volume são dados, respectivamente, por:

$$A = 4 * p * R^2$$

$$V = 4/3 * p * R^3$$

```
#include<stdio.h>

void calc_esfera(float R, float *area, float *volume) {
    (*area) = 4 * 3.1415 * R * R;
    (*volume) = (4/3) * 3.1415 * R * R * R;
}

int main(void)
{
    float r, a, v;
    printf("Digite o raio: ");
    scanf("%f", &r);

    calc_esfera(r, &a, &v);
    printf("\nA area eh: %.2f\n", a);
    printf("\nO volume eh: %.2f\n", v);

    system("pause");
    return 0;
}
```


Ponteiros & Arrays

- ❑ Arrays são agrupamentos de dados do mesmo tipo na memória
 - ❑ Reserva sequencial na memória
- ❑ Como resultado dessa operação, o computador nos devolve um ponteiro que aponta para o começo dessa seqüência de bytes na memória
 - ❑ `int vet[5], *p;`
 - ❑ `p = vet;`

Memória		
#	var	conteúdo
123	int *p	#125
124		
125	int vet[5]	1
126		2
127		3
128		4
129		5



Lista 9

13. Implemente uma função que receba como parâmetro um vetor de números reais (VET) de tamanho N e retorne quantos números negativos há nesse vetor. Essa função deve obedecer ao protótipo:

```
int negativos(float *vet, int N);
```

```
#include<stdio.h>
```

```
int negativos(float *vet, int N) {  
    int i;  
    int count = 0;  
    for (i=0; i<N; i++) {  
        if ( *(vet+i) < 0 )  
            count++;  
    }  
    return count;  
}
```

```
int main(void)  
{  
    float vet[] = {1.0, -2.19, 10.0, -5.9, 0};  
    printf("\nNro de negativos no array: %d\n", negativos(vet, 5));  
  
    system("pause");  
    return 0;  
}
```