Linguagem C: funções e ponteiros

Prof. Críston

Algoritmos e Programação

Funções

- Recurso das linguagens de programação que permite dar um nome para um conjunto de instruções
- Facilita a reutilização de algoritmos
  - com um pouco de organização não precisamos fazer novamente algoritmos que já implementamos no passado
  - podemos utilizar algoritmos implementados por outras pessoas

Funções em C

```
tipo_retorno nome_função (parâmetros)
{
    comando1;
    comando2;
    ...
}
```

- **Bloco**: conjunto de instruções entre chaves
- Comando return

Exemplo

```
double pi ()
{
    printf("Retorna o valor de pi\n");
    return 3.14;
}

main()
{
    double raio = 1.0;
    double area = 2 * pi() * raio;
}
```

Função sem retorno: tipo void

```
void imprime_menu ()
{
    printf("1- Inserir\n");
    printf("2- Remover\n");
    printf("Opcao? ");
}

main()
{
    imprime_menu();
    ...
    imprime_menu();
}
```

Modificando o comportamento das funções: passagem de parâmetros

```
double area (double raio)
{
    return 2 * 3.14 * raio;
}

main()
{
    printf("A area vale %f\n", area(1.0));
}
```

Escopo de variáveis

- Regras que determinam onde as variáveis podem ser acessadas no programa.
  - Variáveis locais
  - Parâmetros
  - Variáveis globais

Variáveis locais

- Existem apenas no bloco onde foram declaradas
- Ex.: variável raio existe apenas dentro da função area

```
double area ()
{
    double raio = 1.0;
    return 2 * 3.14 * raio;
}

main()
{
   // variável raio não pode ser acessada aqui
}
```

Parâmetros

- Podemos interpretar parâmetros como variáveis locais criadas no início da função e inicializadas com os valores recebidos na chamada da função.

```
double area (double raio)
{
    // é o mesmo que “double raio = 1.0;”
    return 2 * 3.14 * raio;
}

main()
{
    printf(“A area vale %f\n”, area(1.0));
}
```

Exercício

- Função fatorial que recebe um inteiro n e retorna o fatorial de n.
- Função base_In que recebe um inteiro k e retorna a soma dos k primeiros termos da série

  1 + 1/1! + 1/2! + 1/3! + …

  Utilize a função fatorial.

Variáveis globais (uso deve ser evitado)

- São declaradas fora das funções e podem ser acessadas por qualquer função
- Se uma função tem uma variável local com o mesmo nome de uma variável local, a variável local será utilizada

Variáveis globais (uso deve ser evitado)

```
int z,k;
func1 (...)
{
    int x,y;
    ...
}
func2 (...)
{
    int x,y,z;
    ...
    z=10;
    ...
}
main ()
{
    int count;
    z=7;
    func2(...);
    ...
}
```

Exercício – O que vai ser impresso na tela?

```
int num;
int func(int a, int b)
{
    a = (a+b)/2;
    num -= a+1;
    return a;
}
main()
{
    int first = 0, sec = 50;
    num = 10;
    printf("%d %d %d\n", num, first, sec);
    num += func(first, sec);
    printf("%d %d %d\n", num, first, sec);
}
```

Ponteiro

- Variável que armazenam o endereço de memória de outra variável
- Declaração: **`tipo *nome;`**
  - Ex: **`int *pt;`**

- Operador &: foneçe o end. de memória de uma variável
  - Ex.:
```
int count = 10;
int *pt;
pt = &count;
```

- Operador *: acessando/modificando variável apontada
  - Ex.:
```
int n = 2 + *pt; // n recebe valor 12
*pt = 5; // isto modifica também count
```

Exemplo

```
main()
{
    int n = 10;
    int *p;
    p = &n;
    printf(“Valor na memória: %d\n", *p);
    printf(“Endereço de memória %p\n", p);
    *p = 5;
    printf(“Valor de n: %d\n", n);
}
```

Passagem de parâmetro por valor x referência

- Por valor: valor é copiado para o parâmetro da função

```
double area (double raio)
{
    return 2 * 3.14 * raio;
}

main()
{
    double r = 1.0;
    printf("A area vale %f\n", area(r));
    // valor de r é copiado para o parâmetro raio
}
```

Passagem de parâmetro por valor x referência

- Por referência: o endereço de memória é copiado para o parâmetro da função (permite modificar a variável passada como parâmetro)

```
double area (double *raio)
{
    return 2 * 3.14 * (*raio);
    raio = 0.0;
}
main()
{
    double r = 1.0;
    printf("A area vale %f\n", area(&r));
    // r agora vale 0.0
}
```

Exemplo – O que vai ser impresso na tela?

```
void incrementa (int *a, int b)
{
   *a += b;
}
main ()
{
   int num;
   num=100;
   printf (“Valor de num %d\n”, num);
   incrementa (&num,50);
   printf (“Valor de num %d\n”, num);
}
```

Exemplos

- Exemplo: função scanf
- Função para trocar o valor de duas variáveis inteiras passadas como parâmetro.

Exercício

- Função que retorna 1 se o valor passado como parâmetro for primo, e 0 caso contrário.
- Função para imprimir todos os valores primos entre A e B recebidos como parâmetro. Utilizar a função anterior.
- Função que recebe por referência 4 variáveis ponto flutuante e ordena (crescente) os valores destas variáveis. Utilize a função que troca os valores de 2 variáveis.