Funções

Professor Leandro O. Freitas
leandro@politecnico.ufsm.br

Retomando...

Sistemas para Internet
UFSM

- Laços: Executam um bloco de código enquanto uma **condição é verdadeira**:

```
while (x > 10) {
    ...
}
```

  - Executa o conjunto de instruções do laço sempre que o valor de x for maior que 10;
  - Testa a condição antes de executar o bloco.

```
do {
    ...
} while (x > 10);
```

  - Executa o conjunto de instruções do laço sempre que o valor de x for maior que 10;
  - Testa a condição após executar o bloco;
  - Executa o bloco de código pelo menos uma vez.

```
for (c=0; c<10; c++) {
    ...
}
```

  - Executa o conjunto de instruções do laço sempre que o valor de c for menor que 10;
  - Define-se um valor de incremento no cabeçalho do laço;
  - Testa a condição antes de executar o laço.

Funções

Sistemas para Internet
UFSM

- Também podem ser conhecidas como:
  - Sub programas, sub algoritmos, métodos, procedimentos, sub rotinas

- O que são funções?
  - Blocos de código que contém início e fim, identificados por um nome, que podem ser utilizados (chamados) em qualquer parte da função principal (main) ou de outras funções;
  - Devem identificar um tipo de retorno;
  - **Devem ser criadas ANTES da função que a chama/invoca!**

Porque utilizar funções?

Sistemas para Internet
UFSM

- Reaproveitamento de código, evitando repetição;
- Facilitar a leitura e o entendimento do código-fonte;
- Evitar que os blocos do programa fiquem muito extensos;

- Utilizamos funções desde nosso primeiro programa em C:

```
int main()
{
     printf ("Estamos criando um programa de computador \n");
     return 0;
}
```

Análise da função main

Sistemas para Internet
UFSM

```
int main() {
    ...
    return 0;
}
```

- É a função que controla/ gerencia o programa. Tudo acontece dentro dela;
- Precisa ter um tipo de retorno:
  - Tipo de retorno é **int** pois deve retornar um valor inteiro;
  - **return 0** indica ao sistema operacional que o programa foi concluído com êxito. Outros valores indicam mensagens de erros específicos ao SO;
- Pode conter parâmetros de comandos utilizados para chamá-la. Estrutura:

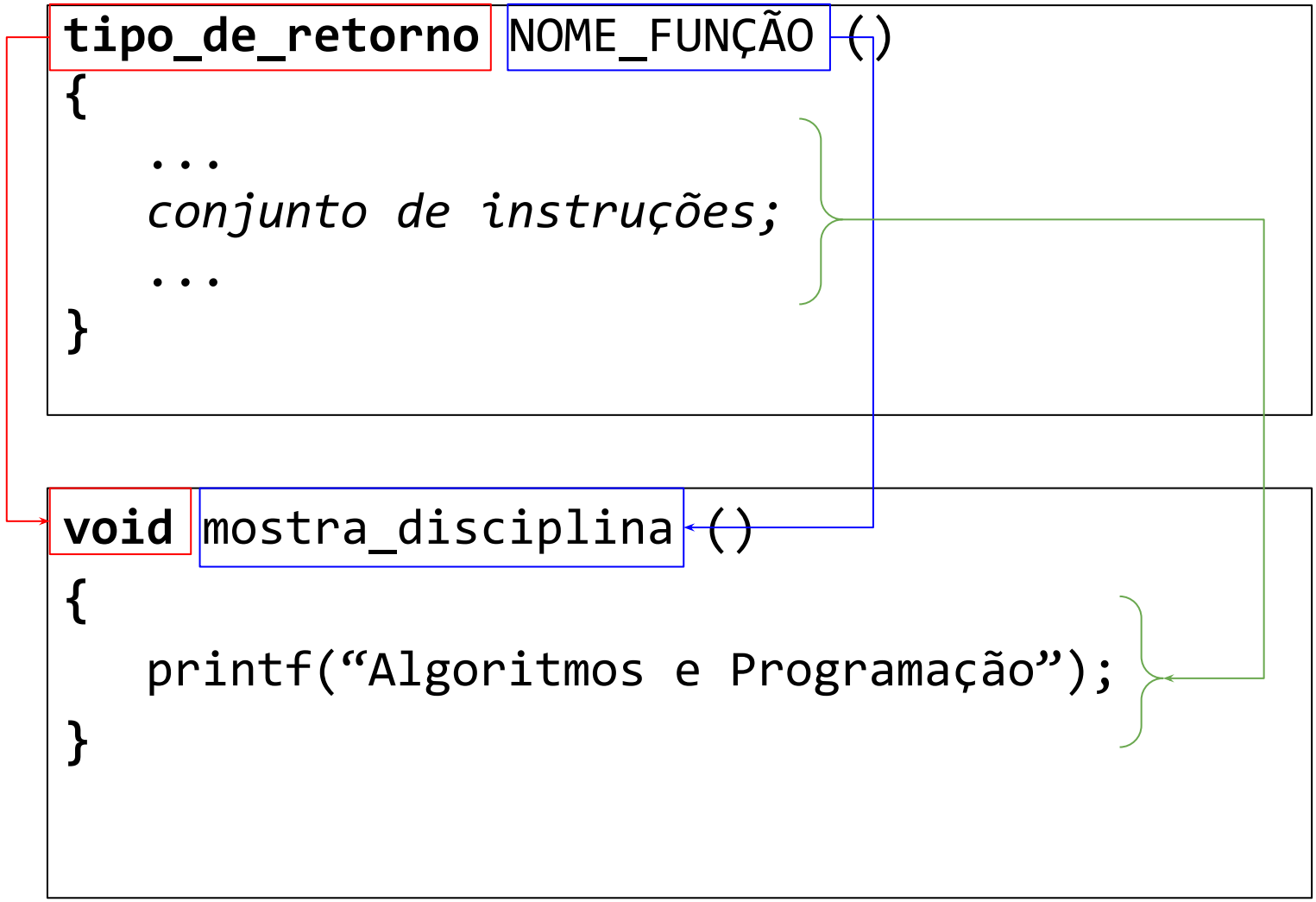
```
int main(int argc, char *argv[]) {
    ...
    return 0;
}
```

Instruções passadas na linha de comando

Total de argumentos passados na linha de comando

Sintaxe em C

Sistemas para Internet
UFSM



Exemplo de programa sem função

Sistemas para Internet
UFSM

```
14 int main ()
15 {
16     printf("Bem-vindo ao programa de soma:");
17
18     int a, b, r;
19     printf("Informe o valor de A: ");
20     scanf("%d", &a);
21     printf("Informe o valor de B: ");
22     scanf("%d", &b);
23     r = a + b;
24     printf("Resultado da soma: %d", r);
25
26     return 0;
27 }
```

```
void soma() {


}
```

Exemplo de programa com função

Sistemas para Internet
UFSM

```
3  void soma ()
4  {
5      int a, b, r;
6      printf("Informe o valor de A: ");
7      scanf("%d", &a);
8      printf("Informe o valor de B: ");
9      scanf("%d", &b);
10     r = a + b;
11     printf("Resultado da soma: %d", r);
12 }
13
14 int main ()  1º
15 {
16     printf("Bem-vindo ao programa de soma:");
17     soma();
18     return 0;  3º
19     }
```

2º

Escopo de variáveis

Tipos de variáveis

Sistemas para Internet
UFSM

- Variáveis **globais**
  - Uma variável é considerada GLOBAL quando é declarada no início do programa (fora de qualquer função), podendo ser utilizada por qualquer função e a qualquer momento durante a execução do programa.

- Variáveis **locais**
  - Uma variável é considerada LOCAL quando ela é declarada dentro de uma função. Ela só é válida dentro da função na qual foi declarada.

Variáveis Globais

Sistemas para Internet
UFSM

- Exemplo em C

Função soma

Função main

```
1  #include<stdio.h>
2
3
4  int a, b, result;
5
6  void soma (){
7      result = a + b;
8      printf("\n Resultado da soma: %d + %d = %d", a, b, result);
9  }
10
11  int main (){
12     printf("\n Informe o valor de a: ");
13     scanf("%d", &a);
14     printf("\n Informe o valor de b: ");
15     scanf("%d", &b);
16
17     soma ();
18
19     return 0;
20 }
```

Variáveis Locais

Sistemas para Internet
UFSM

- Exemplo em C

soma

```
void soma ()
{
    int a, b, result;
    printf("\n Informe o valor de a: ");
    scanf("%d", &a);
    printf("\n Informe o valor de b: ");
    scanf("%d", &b);
    result = a + b;
    printf("\n Resultado da soma: %d + %d = %d", a, b, result);
}
```

main

```
int main ()
{
    soma ();
    printf ("%d %d", a, b);
    return 0;
}
```

→ **Não pode!!!**

Escopo de variáveis

Sistemas para Internet
UFSM

- Quando utilizar variáveis globais ou locais?
  - Depende da aplicação que está sendo criada.

- Variáveis globais :
  - Indicada quando precisa-se usar uma variável em diversas funções;
  - Ocupam memória do computador enquanto o programa está sendo executado.

- Variáveis locais:
  - Indicada quando precisa-se usar uma variável em poucas funções;
  - Ocupam memória do computador apenas enquanto sua função estiver sendo executada.

Exercício 1

Sistemas para Internet
UFSM

- Desenvolva um algoritmo que contenha uma função para verificar se um número é positivo ou negativo. Caso seja positivo, informe se é par ou ímpar.
  - Faça testes utilizando variáveis locais e globais para perceber a diferença.

Solução exercício 1:

Sistemas para Internet
UFSM

```
void verifica()
{
    int numero;
    printf("digite um numero inteiro: ");
    scanf("%d", &numero);
    if (numero > 0)
    {
        printf("Positivo \n");
        if (numero % 2 == 0)
        {
            printf("Par \n");
        }
        else
        {
            printf("Impar \n");
        }
    }
}

int main (){
   verifica();
   return 0;
}
```

Funções com parâmetros

Funções com parâmetros

Sistemas para Internet
UFSM

- O que é um parâmetro?
  - É um valor, ou conjunto de valores, necessário(s) para a execução de uma função.
    - Informações externas que são necessárias para a função.

- O que é passagem de parâmetros?
  - Processo de envio uma cópia de valores de variáveis de uma função para que sejam manipulados dentro de uma outra função.

Sintaxe em C

Sistemas para Internet
UFSM

```
void NOME_FUNÇÃO (parâmetros)
{
    ...
    conjunto de instruções;
    ...
}
```

```
void mostra_idade (int x)
{
    printf("idade %d", x);
}
```

Não seria possível imprimir o valor de " x ",
sem a utilização do parâmetro!

Funções com parâmetros: exemplo

Sistemas para Internet
UFSM

```
1  #include <stdio.h>
2
3  void soma (int x, int y){
4      int result;
5      result = x + y;
6      printf("Resultado da soma: %d\n", result);
7      x = 5;
8  }
9
10 int main()
11 {
12     int a, b;
13     printf("Digite o valor de a: ");
14     scanf("%d", &a);
15     printf("Digite o valor de b: ");
16     scanf("%d", &b);
17     soma(a, b);
18     printf("Valor de a: %d \n", a);
19
20     return 0;
21 }
```

→ Qual o valor de “ a ”?

Exercício 2

Sistemas para Internet
UFSM

- Escreva um programa que contenha uma função para apresentar o quadrado de todos os números inteiros maiores que zero e menores que K (recebido por parâmetro).
- Não esqueça do escopo de variáveis!

Exercício 2

Sistemas para Internet
UFSM

- Escreva um programa que contenha uma função para apresentar o quadrado de todos os números inteiros maiores que zero e menores que K (recebido por parâmetro).
- Não esqueça do escopo de variáveis!!
- Solução:

```
void potencia(int k)
{
    int i;
    for (i = 1; i < k; i++)
    {
        printf("%d ", i * i);
    }
}

int main ()
{
    int numero;
    scanf("%d", &numero);
    potencia(numero);
    return 0;
}
```

Funções com retorno

Funções com retorno

Sistemas para Internet
UFSM

- O comando de retorno tem dois objetivos:
  1. Realizar a saída imediata de uma função;
  2. Retornar um valor para o código que chamou a função;

- Retorno 1º caso:

```
void soma (int x, int y){
    int result = x + y;
    if (result > 10){
        return;
    }
    printf("\n Resultado da soma eh menor que 10.");
}
```

Se a condição do *if* for verdadeira, essa linha não será executada.

Retorno: 2º caso

Sistemas para Internet
UFSM

```
tipo_de_retorno nome_funcao (parâmetros)
{
   <instruções diversas>
   return valor_de_retorno;
}
```

- ***tipo_de_retorno***:
  - Tipo de dado que será devolvido para função chamadora (int, float, char, etc.);

- ***return:***
  - Instrução que permite devolver algum valor para a função chamadora. Deve ser do mesmo tipo do ***tipo_de_retorno***.

Retorno: 2º caso

Sistemas para Internet
UFSM

```
tipo_de_retorno nome_funcao (parâmetros)
{
    <variáveis>
    <instruções>
    return valor_de_retorno;
}
```

```
int soma ()
{
    int x = 5, y = 10, s;
    s = x + y;
    return s;
}
```

Funções com retorno

- 2º caso

```
int soma (int x, int y){
    int r;
    r = x + y;
    return r;
}

int main (){
    int a, b,result;
    printf("\n Informe o valor de a: ");
    scanf("%d", &a);
    printf("\n Informe o valor de b: ");
    scanf("%d", &b);
    result = soma (a, b);
    printf("\n Resultado: %d", result);
    return 0;
}
```

Funções com retorno precisam de uma variável para guardar o valor que será devolvido.

Neste exemplo a variável ***result*** está sendo usada para guardar o valor que retorna da função ***soma.***

Exercício 3

Sistemas para Internet
UFSM

- Escreva um programa que contenha uma função que receba como parâmetro as 4 notas de um aluno e calcule a média. A função deve retornar a média para o programa principal para que seja apresentada na tela.

Exercício 3

Sistemas para Internet
UFSM

- Escreva um programa que contenha uma função que receba como parâmetro as 4 notas de um aluno e calcule a média. A função deve retornar a média para o programa principal para que seja apresentada na tela. Solução:

```
float potencia(float n1, float n2, float n3, float n4)
{
    float mf;
    mf = (n1 + n2 + n3 + n4)/4;
    return mf;
}

int main ()
{
    float n1, n2, n3, n4, media;
    scanf("%f %f %f %f", &n1, &n2, &n3, &n4);
    media = potencia(n1, n2, n3, n4);
    printf("Media final %.2f", media);
    return 0;
}
```

Protótipos

Protótipos

Sistemas para Internet
UFSM

- Declaração de funções
  - O protótipo de uma função é uma **declaração** da mesma, antes da função MAIN. Isso permite informar ao compilador que a função existe e está definida em outro ponto do programa.

- Criação de um protótipo
  - A sintaxe de um protótipo de uma função é igual a primeira linha dessa função, onde são definidos o tipo de retorno, nome e parâmetros (se houver). Porém, ao invés de abrir um bloco de código com chaves ***{ }***, a linha termina com ponto-e-vírgula ***;***. Assim, a função poderá desenvolvida após a MAIN.

- Vantagem: Organização do código fonte.

Protótipos: Exemplos

- Linguagem C:

```c
#include <stdio.h>

void soma (int x, int y);

int main() {
    int a, b;
    printf("Digite o valor de a: ");
    scanf("%d", &a);
    printf("Digite o valor de b: ");
    scanf("%d", &b);
    soma(a, b);
    printf("Valor de a: %d \n", a);

    return 0;
}

void soma (int x, int y) {
    int result;
    result = x + y;
    printf("Resultado da soma: %d\n", result);
    x = 5;
}
```

Exercícios

Sistemas para Internet
UFSM

- Altere os exemplos vistos durante a aula para utilização de protótipo de funções.