Estrutura Linear

Informática - IFSULDEMINAS

Primeiro Semestre de 2013

Roteiro

A função *printf*

- A função *printf* é uma função da biblioteca *stdio.h*.
- Ela permite apresentar textos e conteúdos de variáveis na tela.

Exemplo

```
printf("Linguagem de Programação!");
```

Saída: Linguagem de Programação!

A função *printf*

- A função *printf* é uma função da biblioteca *stdio.h*.
- Ela permite apresentar textos e conteúdos de variáveis na tela.

Exemplo

```
printf("Linguagem de Programação!");
```

Saída: Linguagem de Programação!

A função *printf*

- A função *printf* é uma função da biblioteca *stdio.h*.
- Ela permite apresentar textos e conteúdos de variáveis na tela.

Exemplo

```
printf("Linguagem de Programação!");
```

Saída: Linguagem de Programação!

A função *printf*

- A função *printf* é uma função da biblioteca *stdio.h*.
- Ela permite apresentar textos e conteúdos de variáveis na tela.

Exemplo

```
printf("Linguagem de Programação!");
```

Saída: Linguagem de Programação!

Escrevendo variáveis na tela

- Além de textos podemos imprimir o conteúdo de variáveis.
- Para que o conteúdo de variáveis seja impresso na tela utilizamos símbolos especiais no texto para representar trechos que devem ser substituídos por variáveis.
- Após a string com os caracteres especiais passamos uma lista de variáveis ou constantes a serem substituídas, separadas por vírgula.
- Para cada caracter especial (representando uma variável), temos uma variável associada.

Forma Geral

```
printf (<string>, <lista_de_argumentos>);
```

Escrevendo variáveis na tela

- Além de textos podemos imprimir o conteúdo de variáveis.
- Para que o conteúdo de variáveis seja impresso na tela utilizamos símbolos especiais no texto para representar trechos que devem ser substituídos por variáveis.
- Após a string com os caracteres especiais passamos uma lista de variáveis ou constantes a serem substituídas, separadas por vírgula.
- Para cada caracter especial (representando uma variável), temos uma variável associada.

Forma Geral

```
printf (<string>, <lista_de_argumentos>);
```

Escrevendo variáveis na tela

- Além de textos podemos imprimir o conteúdo de variáveis.
- Para que o conteúdo de variáveis seja impresso na tela utilizamos símbolos especiais no texto para representar trechos que devem ser substituídos por variáveis.
- Após a string com os caracteres especiais passamos uma lista de variáveis ou constantes a serem substituídas, separadas por vírgula.
- Para cada caracter especial (representando uma variável), temos uma variável associada.

Forma Geral

```
printf (<string>, <lista_de_argumentos>);
```

Escrevendo variáveis na tela

- Além de textos podemos imprimir o conteúdo de variáveis.
- Para que o conteúdo de variáveis seja impresso na tela utilizamos símbolos especiais no texto para representar trechos que devem ser substituídos por variáveis.
- Após a string com os caracteres especiais passamos uma lista de variáveis ou constantes a serem substituídas, separadas por vírgula.
- Para cada caracter especial (representando uma variável), temos uma variável associada.

Forma Geral

```
printf (<string>, <lista_de_argumentos>);
```

Escrevendo variáveis na tela

- Além de textos podemos imprimir o conteúdo de variáveis.
- Para que o conteúdo de variáveis seja impresso na tela utilizamos símbolos especiais no texto para representar trechos que devem ser substituídos por variáveis.
- Após a string com os caracteres especiais passamos uma lista de variáveis ou constantes a serem substituídas, separadas por vírgula.
- Para cada caracter especial (representando uma variável), temos uma variável associada.

Forma Geral

printf (<string>, <lista_de_argumentos>);

Escrevendo variáveis na tela

Exemplo

```
printf("O aluno %s fez %d provas e sua média final foi
%f.", "Marcos", n, nota);
```

- Saída: O aluno Marcos fez 3 provas e sua média final foi 8.2.
- O símbolo %s foi substituído pela string Marcos, %d foi substituído pelo inteiro 3 armazenado na variável n, e %f foi substituído pelo valor 8.2 armazenado na variável nota.

Escrevendo variáveis na tela

Exemplo

```
printf("O aluno %s fez %d provas e sua média final foi
%f.", "Marcos", n, nota);
```

- Saída: O aluno Marcos fez 3 provas e sua média final foi 8.2.
- O símbolo %s foi substituído pela string Marcos, %d foi substituído pelo inteiro 3 armazenado na variável n, e %f foi substituído pelo valor 8.2 armazenado na variável nota.

Escrevendo variáveis na tela

Exemplo

```
printf("O aluno %s fez %d provas e sua média final foi
%f.", "Marcos", n, nota);
```

- Saída: O aluno Marcos fez 3 provas e sua média final foi 8.2.
- O símbolo %s foi substituído pela string Marcos, %d foi substituído pelo inteiro 3 armazenado na variável n, e %f foi substituído pelo valor 8.2 armazenado na variável nota.

Símbolos especiais

- Os símbolos especiais estão associados aos tipos de dados.

Código	Função
%c	Lê um único caracter
%s	Lê uma série de caracteres
%d	Lê um número decimal
%f	Lê um número em ponto flutuante

printf e os símbolos especiais

%d — Escreve um inteiro na tela.

Exemplo

```
printf ("%d", 23);
```

- Saída: 23

Exemplo

```
int idade = 15;
printf ("O aluno tem %d anos.", idade);
```

- Saída: O aluno tem 15 anos.

printf e os símbolos especiais

%d — Escreve um inteiro na tela.

Exemplo

```
printf ("%d", 23);
```

- Saída: 23

Exemplo

```
int idade = 15;
printf ("O aluno tem %d anos.", idade);
```

- Saída: O aluno tem 15 anos.

printf e os símbolos especiais

%d — Escreve um inteiro na tela.

Exemplo

```
printf ("%d", 23);
```

- Saída: 23

Exemplo

```
int idade = 15;
printf ("O aluno tem %d anos.", idade);
```

- Saída: O aluno tem 15 anos.

printf e os símbolos especiais

%d — Escreve um inteiro na tela.

Exemplo

```
printf ("%d", 23);
```

- Saída: 23

Exemplo

```
int idade = 15;
printf ("O aluno tem %d anos.", idade);
```

- Saída: O aluno tem 15 anos.

printf e os símbolos especiais

%f — Escreve um ponto flutuante na tela.

Exemplo

```
printf ("%f", 56.5);
```

- Saída: 56.5
- Obs: Podemos utilizar o [%.< *decimais* >f] para determinar o número de casas decimais.

Exemplo

```
float pi = 3.14159;
printf ("O valor de PI é %.2f.", pi);
```

- Saída: O valor de PI é 3.14.

printf e os símbolos especiais

%f — Escreve um ponto flutuante na tela.

Exemplo

```
printf ("%f", 56.5);
```

- Saída: 56.5
- Obs: Podemos utilizar o [%.< *decimais* >f] para determinar o número de casas decimais.

Exemplo

```
float pi = 3.14159;
printf ("O valor de PI é %.2f.", pi);
```

- Saída: O valor de PI é 3.14.

printf e os símbolos especiais

%f — Escreve um ponto flutuante na tela.

Exemplo

```
printf ("%f", 56.5);
```

- Saída: 56.5
- Obs: Podemos utilizar o [%.< *decimais* >f] para determinar o número de casas decimais.

Exemplo

```
float pi = 3.14159;
printf ("O valor de PI é %.2f.", pi);
```

- Saída: O valor de PI é 3.14.

printf e os símbolos especiais

%f — Escreve um ponto flutuante na tela.

Exemplo

```
printf ("%f", 56.5);
```

- Saída: 56.5
- Obs: Podemos utilizar o [%.< *decimais* >f] para determinar o número de casas decimais.

Exemplo

```
float pi = 3.14159;
printf ("O valor de PI é %.2f.", pi);
```

- Saída: O valor de PI é 3.14.

printf e os símbolos especiais

%f — Escreve um ponto flutuante na tela.

Exemplo

```
printf ("%f", 56.5);
```

- Saída: 56.5
- Obs: Podemos utilizar o [%.< *decimais* >f] para determinar o número de casas decimais.

Exemplo

```
float pi = 3.14159;
printf ("O valor de PI é %.2f.", pi);
```

- Saída: O valor de PI é 3.14.

printf e os símbolos especiais

%c — Escreve uma letra.

Exemplo

```
printf ("%c", 'I');
```

- Saída: I.

Exemplo

```
char sexo = 'M';
printf ("O sexo é %c.", sexo);
```

- Saída: O sexo é M.

printf e os símbolos especiais

%c — Escreve uma letra.

Exemplo

```
printf ("%c", 'I');
```

- Saída: I.

Exemplo

```
char sexo = 'M';
printf ("O sexo é %c.", sexo);
```

- Saída: O sexo é M.

printf e os símbolos especiais

%c — Escreve uma letra.

Exemplo

```
printf ("%c", 'I');
```

- Saída: I.

Exemplo

```
char sexo = 'M';
printf ("O sexo é %c.", sexo);
```

- Saída: O sexo é M.

printf e os símbolos especiais

%c — Escreve uma letra.

Exemplo

```
printf ("%c", 'I');
```

- Saída: I.

Exemplo

```
char sexo = 'M';
printf ("O sexo é %c.", sexo);
```

- Saída: O sexo é M.

printf e os símbolos especiais

%s — Escreve uma string

Exemplo

```
printf ("%s", "IFSULDEMINAS - Informática");
```

- Saída: IFSULDEMINAS - Informática.

Exemplo

```
char curso[30] = "Informática";
printf ("Faço %s no IFSULDEMINAS.");
```

- Saída: Faço Informática no IFSULDEMINAS.

printf e os símbolos especiais

%s — Escreve uma string

Exemplo

```
printf ("%s", "IFSULDEMINAS - Informática");
```

- Saída: IFSULDEMINAS - Informática.

Exemplo

```
char curso[30] = "Informática";
printf ("Faço %s no IFSULDEMINAS.");
```

- Saída: Faço Informática no IFSULDEMINAS.

printf e os símbolos especiais

%s — Escreve uma string

Exemplo

```
printf ("%s", "IFSULDEMINAS - Informática");
```

- Saída: IFSULDEMINAS - Informática.

Exemplo

```
char curso[30] = "Informática";
printf ("Faço %s no IFSULDEMINAS.");
```

- Saída: Faço Informática no IFSULDEMINAS.

printf e os símbolos especiais

%s — Escreve uma string

Exemplo

```
printf ("%s", "IFSULDEMINAS - Informática");
```

- Saída: IFSULDEMINAS - Informática.

Exemplo

```
char curso[30] = "Informática";
printf ("Faço %s no IFSULDEMINAS.");
```

- Saída: Faço Informática no IFSULDEMINAS.

Constantes de Barra Invertida

Para facilitar, a linguagem C utiliza alguns comandos de barra invertida. A tabela a seguir mostra alguns códigos de barra invertida.

Código	Função
\t	Tabulação Horizontal
\"	Aspas
\'	Apóstofro
\n	Nova linha

Exemplo

```
printf ("\t Texto usando constantes\n de barra
invertida.");
```

Saída:

Texto usando constantes
de barra invertida.

Constantes de Barra Invertida

Para facilitar, a linguagem C utiliza alguns comandos de barra invertida. A tabela a seguir mostra alguns códigos de barra invertida.

Código	Função
\t	Tabulação Horizontal
\"	Aspas
\'	Apóstofro
\n	Nova linha

Exemplo

```
printf ("\t Texto usando constantes\n de barra
invertida.");
```

Saída:

Texto usando constantes
de barra invertida.

A função *scanf*

- A função *scanf* é uma função da biblioteca *stdio.h*.
- Ela permite a interação com o usuário por meio da leitura de dados do teclado.

Forma Geral

```
printf (<string>, <lista_de_argumentos>);
```

- Parâmetros:
  - A string, indica os tipos de variáveis que serão lidas por meio dos símbolos especiais.
  - A lista de argumentos é a lista de variáveis.

A função *scanf*

- A função *scanf* é uma função da biblioteca *stdio.h*.
- Ela permite a interação com o usuário por meio da leitura de dados do teclado.

Forma Geral

```
printf (<string>, <lista_de_argumentos>);
```

- Parâmetros:
  - A string, indica os tipos de variáveis que serão lidas por meio dos símbolos especiais.
  - A lista de argumentos é a lista de variáveis.

A função *scanf*

- A função *scanf* é uma função da biblioteca *stdio.h*.
- Ela permite a interação com o usuário por meio da leitura de dados do teclado.

Forma Geral

printf (<string>, <lista_de_argumentos>);

- Parâmetros:
  - A string, indica os tipos de variáveis que serão lidas por meio dos símbolos especiais.
  - A lista de argumentos é a lista de variáveis.

A função *scanf*

- A função *scanf* é uma função da biblioteca *stdio.h*.
- Ela permite a interação com o usuário por meio da leitura de dados do teclado.

Forma Geral

printf (<string>, <lista_de_argumentos>);

- Parâmetros:
  - A string, indica os tipos de variáveis que serão lidas por meio dos símbolos especiais.
  - A lista de argumentos é a lista de variáveis.

A função *scanf*

- A função *scanf* é uma função da biblioteca *stdio.h*.
- Ela permite a interação com o usuário por meio da leitura de dados do teclado.

Forma Geral

printf (<string>, <lista_de_argumentos>);

- Parâmetros:
  - A string, indica os tipos de variáveis que serão lidas por meio dos símbolos especiais.
  - A lista de argumentos é a lista de variáveis.

Lendo dados do usuário

O programa abaixo é composto de quatro passos:

1. Cria uma variável `n`.
2. Escreve na tela: `Digite um número`.
3. Lê o valor do número digitado.
4. Imprime o valor do número digitado.

Exemplo

```
#include <stdio.h>
int main(){
  float n;
  printf("Digite um número: ");
  scanf("%f",&n);
  printf("O valor digitado foi %f\n",n);
}
```

Lendo dados do usuário

Leitura de várias variáveis

```
#include <stdio.h>
main(){
   int n;
   float v;
   char c;
   printf("Digite um inteiro, um real e um caracter: ");
   scanf("%d %f %c",&n, &v, &c);
   printf("O dados digitados foram:
           %d %f %c\n", n, v, c);
}
```

Observações

- Os símbolos especiais utilizados na escrita são os mesmos utilizados na leitura de dados.
- Para a leitura de dados do tipo *int*, *float* e *char* suas respectivas variáveis devem ser precedidas com &.
- A leitura de strings utilizando a função *scanf* lê apenas palavras.

Observações

- Os símbolos especiais utilizados na escrita são os mesmos utilizados na leitura de dados.
- Para a leitura de dados do tipo *int*, *float* e *char* suas respectivas variáveis devem ser precedidas com &.
- A leitura de strings utilizando a função *scanf* lê apenas palavras.

Observações

- Os símbolos especiais utilizados na escrita são os mesmos utilizados na leitura de dados.
- Para a leitura de dados do tipo *int*, *float* e *char* suas respectivas variáveis devem ser precedidas com &.
- A leitura de strings utilizando a função *scanf* lê apenas palavras.

A função gets

- Realiza a leitura de um texto ou frase a partir do teclado.

Forma Geral

```
gets (<variavel>);
```

- Parâmetros:
  - Uma variável.
- Aguarda que o usuário digite um valor e atribui o valor digitado à variável.

Exemplo

```
char nome[30];
printf ("Entre com seu nome completo:");
gets(nome);
```

A função gets

- Realiza a leitura de um texto ou frase a partir do teclado.

Forma Geral

gets (<variavel>);

- Parâmetros:
  - Uma variável.
- Aguarda que o usuário digite um valor e atribui o valor digitado à variável.

Exemplo

```
char nome[30];
printf ("Entre com seu nome completo:");
gets(nome);
```

A função gets

- Realiza a leitura de um texto ou frase a partir do teclado.

Forma Geral

gets (<variavel>);

- Parâmetros:
  - Uma variável.
- Aguarda que o usuário digite um valor e atribui o valor digitado à variável.

Exemplo

```
char nome[30];
printf ("Entre com seu nome completo:");
gets(nome);
```

A função gets

- Realiza a leitura de um texto ou frase a partir do teclado.

Forma Geral

gets (<variavel>);

- Parâmetros:
  - Uma variável.
- Aguarda que o usuário digite um valor e atribui o valor digitado à variável.

Exemplo

```
char nome[30];
printf ("Entre com seu nome completo:");
gets(nome);
```

A função gets

- Realiza a leitura de um texto ou frase a partir do teclado.

Forma Geral

gets (<variavel>);

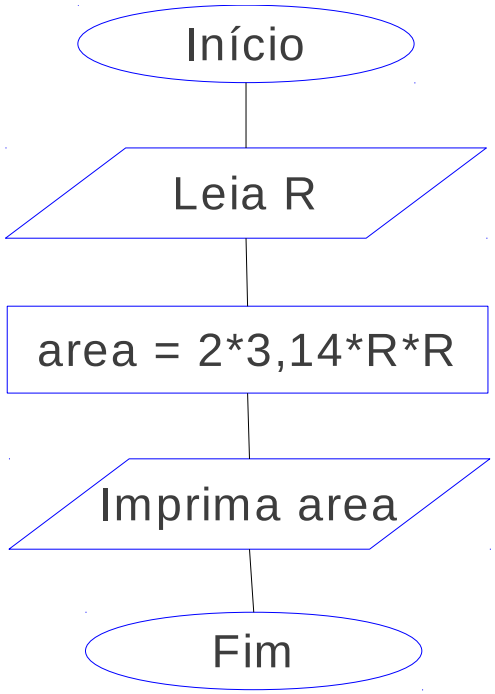
- Parâmetros:
  - Uma variável.
- Aguarda que o usuário digite um valor e atribui o valor digitado à variável.

Exemplo

```
char nome[30];
printf ("Entre com seu nome completo:");
gets(nome);
```

Estrutura Linear

- A estrutura linear representa uma sequência básica de computação: entrada, processamento e saída.
- Exemplo: Escreva um programa que calcule a área de uma circunferência. Área $= 2\Pi R^2$



Estrutura Linear

Área da circunferência

```
#include<stdio.h>

main() {
    float R, area;

    printf("Qual o valor do raio?");
    scanf("%f", &R);

    area = 2*3.14*R*R;

    printf("Area da circunferencia: %f", area);
}
```