PUC Minas

Entrada e Saída

João Pedro Oliveira Batisteli

Fevereiro, 2025

Saída

Como mostrar o resultado do programa que você desenvolveu?

Saída

Como mostrar o resultado do programa que você desenvolveu?

A saída padrão é o local para onde o programa envia os dados que gera.

Saída

Como mostrar o resultado do programa que você desenvolveu?

A saída padrão é o local para onde o programa envia os dados que gera.

A saída padrão de um programa desenvolvido para Console é a tela.

Saída

Como mostrar o resultado do programa que você desenvolveu?

A saída padrão é o local para onde o programa envia os dados que gera.

A saída padrão de um programa desenvolvido para Console é a tela.

Como os dados armazenados na memória do computador são apenas 0´s e 1´s, o tipo indica como representar a informação. Mas a disposição dela na saída também pode ser formatada.

Saída

Como mostrar o resultado do programa que você desenvolveu?

A saída padrão é o local para onde o programa envia os dados que gera.

A saída padrão de um programa desenvolvido para Console é a tela.

Como os dados armazenados na memória do computador são apenas 0´s e 1´s, o tipo indica como representar a informação. Mas a disposição dela na saída também pode ser formatada.

É possível redirecionar a saída padrão para um arquivo ou para outro dispositivo, como uma impressora.

Saída em C

As funções de entrada e saída estão na biblioteca `stdio.h`.

Saída em C

As funções de entrada e saída estão na biblioteca `stdio.h`.

Saída: `printf`

Saída em C

As funções de entrada e saída estão na biblioteca `stdio.h`.

Saída: `printf`

Entrada: `scanf`

Sintaxe 1:

printf

Sintaxe 1:

`printf(...);`.

printf

Sintaxe 1:

`printf(...);`.

Escreve a informação entre parênteses a partir da posição atual do cursor no Console.

printf

Sintaxe 1:

`printf(...);`.

Escreve a informação entre parênteses a partir da posição atual do cursor no Console.

Sintaxe 2:

printf

Sintaxe 1:

`printf(...);`.

Escreve a informação entre parênteses a partir da posição atual do cursor no Console.

Sintaxe 2:

`printf("... %? ... %? ...", var1, var2);`.

printf

Sintaxe 1:

`printf(...);`.

Escreve a informação entre parênteses a partir da posição atual do cursor no Console.

Sintaxe 2:

`printf("... %? ... %? ...", var1, var2);`.

Nas interrogações os conteúdos das variáveis (`var1` e `var2`) na ordem indicada é integrado ao texto (conjunto de caracteres).

Saída Formatada

Especificador	Tipo
%d	int
%.xf	float (opcional x casas decimais)
%.xlf	double (opcional x casas decimais)
%s	char[]
%b	bool
%c	char

Saída Formatada

Códigos de formatação para printf()	Significado
%c	Caractere simples.
%d	Inteiro decimal com sinal.
%i	Inteiro decimal com sinal.
%e	Notação científica (e minúsculo).
%E	Notação científica (E maiúsculo).
%f	Ponto flutuante em decimal.
%g	Usa %e ou %f, o que for menor.
%G	Usa %E ou %f, o que for menor.
%o	Inteiro octal sem sinal.
%s	String de caracteres.
%u	Inteiro decimal sem sinal.
%x	Inteiro hexadecimal sem sinal (letras minúsculas).
%X	Inteiro hexadecimal sem sinal (letras maiúsculas).
%p	Ponteiro (endereço).
%n	Ponteiro inteiro.
%%	Imprime um caractere %.

Exemplos

Código:

```
1  float numero = 5.1520;
2  printf("Número real: %f \n", numero);
3  printf("Número real com 2 casas decimais: %.2f \n", numero);
4  printf("Número inteiro %d \n", (int) numero);
5  printf("Character: %c", (char)((int) numero+32));
```

Exemplos

Código:

```
1  float numero = 5.1520;
2  printf("Número real: %f \n", numero);
3  printf("Número real com 2 casas decimais: %.2f \n", numero);
4  printf("Número inteiro %d \n", (int) numero);
5  printf("Character: %c", (char)((int) numero+32));
```

Saída:

Exemplos

Código:

```
1  float numero = 5.1520;
2  printf("Número real: %f \n", numero);
3  printf("Número real com 2 casas decimais: %.2f \n", numero);
4  printf("Número inteiro %d \n", (int) numero);
5  printf("Character: %c", (char)((int) numero+32));
```

Saída:

Número real: 5.152000

Número real com 2 casas decimais: 5.15

Número inteiro 5

Character %

Exercício

1. Crie duas variáveis, `var1` e `var2`, para armazenar os valores 7.76835 e 45, respectivamente. Escolha o tipo de dado mais adequado para cada variável.

2. Exiba os valores armazenados em var1 e var2 no console. Utilize uma única instrução de impressão para exibir ambos os valores, separados por uma vírgula.

3. Exiba a parte inteira do valor armazenado em var2 no console.

4. Exiba o caractere referente ao valor inteiro armazenado na variável `var2` no console.

Caracteres Especiais

Dentro da *string* (entre os aspas do comando `printf`), pode-se incluir após uma '\' caracteres que assumem funções especiais:

Caracteres Especiais

Dentro da *string* (entre os aspas do comando `printf`), pode-se incluir após uma '\' caracteres que assumem funções especiais:

- \n - Nova linha

Caracteres Especiais

Dentro da *string* (entre os aspas do comando `printf`), pode-se incluir após uma '\' caracteres que assumem funções especiais:

- $\backslash n$ - Nova linha
- $\backslash t$ - Tabulação

Caracteres Especiais

Dentro da *string* (entre os aspas do comando `printf`), pode-se incluir após uma '\' caracteres que assumem funções especiais:

- \n - Nova linha
- \t - Tabulação
- \\ - Exibe a \

Caracteres Especiais

Dentro da *string* (entre os aspas do comando `printf`), pode-se incluir após uma '\' caracteres que assumem funções especiais:

- \n - Nova linha
- \t - Tabulação
- \\ - Exibe a \
- \" - Exibe a "

Exemplo de Caracteres Especiais

Código:

```
printf("Exemplo \n \" de \t \t caracteres \\ especiais
\\");
```

Exemplo de Caracteres Especiais

Código:

```
printf("Exemplo \n \" de \t \t caracteres \\ especiais
\\");
```

Saída:

Exemplo de Caracteres Especiais

Código:

```
printf("Exemplo \n \" de \t \t caracteres \\ especiais
\\");
```

Saída:

Exemplo
"de caracteres \especiais\

Exercícios

1) Escreva o comando `printf` que gera a seguinte saída:

Este "é um
	exemplo \de	saída\

Exercícios

1) Escreva o comando `printf` que gera a seguinte saída:

```
Este "é um
        exemplo \de      saída\
```

2) Escreva o comando `printf` que gera a seguinte saída:

```
Este
\"é
outro
exemplo \de      saída\
```

String

O termo string serve para identificar uma sequência de caracteres.

String

O termo string serve para identificar uma sequência de caracteres.

Na prática, as strings são usadas para representar textos.

String

O termo string serve para identificar uma sequência de caracteres.

Na prática, as strings são usadas para representar textos.

Uma string deve começar e terminar em uma mesma linha, ou seja, o "abre aspas" e o "fecha aspas" devem estar na mesma linha.

String

O termo string serve para identificar uma sequência de caracteres.

Na prática, as strings são usadas para representar textos.

Uma string deve começar e terminar em uma mesma linha, ou seja, o "abre aspas" e o "fecha aspas" devem estar na mesma linha.

Para textos maiores, pode-se utilizar mais de uma string em sequência, delimitada por aspas.

Exemplos de string maiores

Código:

```
printf("Exemplo de texto"
"muito extenso\n");
```

Exemplos de string maiores

Código:

```
printf("Exemplo de texto"
"muito extenso\n");
```

Saída:

Exemplos de string maiores

Código:

```
printf("Exemplo de texto"
"muito extenso\n");
```

Saída:

Exemplo de texto muito extenso

Exercícios

Entrada

Para um programa desenvolvido para console a saída padrão é a tela.

Entrada

Para um programa desenvolvido para console a saída padrão é a tela.

Entrada padrão de um programa desenvolvido para console $\longrightarrow$?

Entrada

Para um programa desenvolvido para console a saída padrão é a tela.

Entrada padrão de um programa desenvolvido para console $\longrightarrow$ teclado.

Entrada

Para um programa desenvolvido para console a saída padrão é a tela.

Entrada padrão de um programa desenvolvido para console $\longrightarrow$ teclado.

A função `scanf` recebe do teclado uma lista de números, caracteres, strings, etc...

scanf

O comando scanf possui o seguinte formato geral:

scanf

O comando scanf possui o seguinte formato geral:

```
scanf("%?  ...  %?", &var1, ..., &varn);
```

scanf

O comando scanf possui o seguinte formato geral:

```
scanf("%?  ...  %?", &var1, ..., &varn);
```

O %? e o %? são referentes aos códigos de formatação (igual ao utilizado no scanf).

scanf

O comando scanf possui o seguinte formato geral:

```
scanf("%?  ...  %?", &var1, ..., &varn);
```

O %? e o %? são referentes aos códigos de formatação (igual ao utilizado no scanf).

Eles se referem ao tipo da variável que irá armazenar o dado lido (pode ocorrer uma conversão implícita durante a leitura).

Exemplo - Recebendo um inteiro

```
#include <stdio.h>
int main(void){
   int a;
   printf("Digite um número inteiro \n");
   scanf("%d", a);
   printf("O número digitado foi:  %d", &a);
   return 0;
}
```

Exemplo - um pouco mais complexo

```
#include <stdio.h>
int main(void){
   int a, b;
   double val;
   float val2;
   char c;
   printf("Hello World \n");
   .
   .
   .
   return 0;
}
```

Exemplo - um pouco mais complexo

```
#include <stdio.h>
int main(void){
   int a, b;
   double val;
   float val2;
   char c;
   printf("Hello World \n");
   .
   .
   .
   return 0;
}
```

Exercício: Receba os valores de `a`, `b`, `val1`, `val2` e `c` usando `scanf`. Depois imprima todos os valores usando `printf`.

Dúvidas?