

## Funções de Entrada e Saída Formatada

➤ printf (continuação)

**Exercício:** Construa um programa em C que utilizando-se dos códigos % escreva na saída padrão a seguinte sequência no formato apresentado:

|                       |       |       |           |
|-----------------------|-------|-------|-----------|
| teste                 | 00027 | 28.37 | funcionou |
| -----                 |       |       |           |
| Número de<br>colunas: | 15    | 5     | 10        |
|                       |       |       | 20        |

## Funções de Entrada e Saída Formatada

➡ printf (continuação)

Exercício (resposta):

```
#include<stdio.h>
main ()
{
    printf("|%-15s|%05d|%10.2f|%20s|",
        "teste",27,28.37,"funcionou");
}
```

### Constantes de barra invertida

| Constante | Significado                |
|-----------|----------------------------|
| \n        | new line                   |
| \"        | aspas                      |
| \'        | apóstrofo                  |
| \0        | nulo (zero decimal)        |
| \\        | barra invertida            |
| \t        | tabulação horizontal (tab) |
| \b        | retorno do cursor          |

# **Funções de Entrada e Saída Formatada**

## **Parte III**



## Funções de Entrada e Saída Formatada

### ✦ **scanf ()**

- função para leitura de dados;
- formato geral:

***scanf (string\_de\_controle,  
lista\_de\_argumentos);***

- ✦ *string\_de\_controle* → descrição de todos os valores que serão lidos, com informações de seus tipos e ordem de leitura.
- ✦ *lista\_de\_argumentos* → lista com os identificadores das variáveis onde os valores lidos serão armazenados, em ordem compatível com a *string\_de\_controle*;

## Funções de Entrada e Saída Formatada

➤ scanf (continuação)

### Tabela simplificada de códigos de formato (%)

| Código | Formato                         |
|--------|---------------------------------|
| %c     | Um caractere (char)             |
| %d     | Um número inteiro decimal (int) |
| %f     | Ponto flutuante decimal         |
| %s     | String                          |

## Funções de Entrada e Saída Formatada

### ➡ scanf (continuação)

Observação: colocar antes de cada identificador da *lista de argumentos* o caractere '&'

Exemplo:

```
#include<stdio.h>
int main()
{
    char ch;
    scanf ("%c", &ch);
    . . .
}
```

### Exercício:

Construa um programa, na linguagem C, que solicite ao usuário o fornecimento de um valor real, através da entrada padrão, e o retorne na saída padrão com dois dígitos de precisão.

## Funções de Entrada e Saída Formatada

```
#include <stdio.h>
int main ()
{
    float valor;
    scanf("%f", &valor);
    printf("%.2f" , valor);
}
```

## Funções de Entrada e Saída Formatada

```
#include <stdio.h>
int main ()
{
    float valor;
    printf("Forneca um valor real: ");
    scanf("%f", &valor);
    printf("\nO valor fornecido foi: %.2f\n", valor);
}
```

### Exercício:

Construa um programa, na linguagem C, que receba três notas e seus respectivos pesos, através da entrada padrão, calcule a média ponderada dessas notas e exiba o resultado na saída padrão.

```
#include <stdio.h>
int main ()
{
    float n1, n2, n3, mediaPonderada;
    int p1, p2, p3;
    printf("Digite a primeira nota: ");
    scanf("%f", &n1);
    printf("\nDigite o peso da primeira nota: ");
    scanf("%d", &p1);
    printf("Digite a segunda nota: ");
    scanf("%f", &n2);
    printf("\nDigite o peso da segunda nota: ");
    scanf("%d", &p2);
    printf("Digite a terceira nota: ");
    scanf("%f", &n3);
    printf("\nDigite o peso da terceira nota: ");
    scanf("%d", &p3);
    mediaPonderada = (n1*p1+n2*p2+n3*p3)/(p1+p2+p3);
    printf("\nA media ponderada eh %.2f", mediaPonderada);
}
```



```
#include <stdio.h>
int main ()
{
    float n1, n2, n3;
    int p1, p2, p3;
    printf("Digite a primeira nota: ");
    scanf("%f", &n1);
    printf("\nDigite o peso da primeira nota: ");
    scanf("%d", &p1);
    printf("Digite a segunda nota: ");
    scanf("%f", &n2);
    printf("\nDigite o peso da segunda nota: ");
    scanf("%d", &p2);
    printf("Digite a terceira nota: ");
    scanf("%f", &n3);
    printf("\nDigite o peso da terceira nota: ");
    scanf("%d", &p3);
    printf("\nA media ponderada eh %.2f",
        (n1*p1+n2*p2+n3*p3)/(p1+p2+p3));
}
```

```
#include <stdio.h>
int main () {
    float nota, notasPonderadas=0.0;
    int peso, pesos=0;
    printf("Digite a primeira nota: ");
    scanf("%f", &nota);
    printf("\nDigite o peso da primeira nota: ");
    scanf("%d", &peso);
    notasPonderadas += nota*peso;
    pesos += peso;
    printf("Digite a segunda nota: ");
    scanf("%f", &nota);
    printf("\nDigite o peso da segunda nota: ");
    scanf("%d", &peso);
    notasPonderadas += nota*peso;
    pesos += peso;
    printf("Digite a terceira nota: ");
    scanf("%f", &nota);
    printf("\nDigite o peso da terceira nota: ");
    scanf("%d", &peso);
    notasPonderadas += nota*peso;
    pesos += peso;
    printf("\nA media ponderada eh %.2f", notasPonderadas/pesos);
}
```



# Modificadores de Tipos

# Modificadores de Tipos



Palavras reservadas:

- ➡ **signed (com sinal)**
- ➡ **unsigned (sem sinal)**
- ➡ **long (expande faixa de valores)**
- ➡ **short (contrai faixa de valores)**

Exemplo:    unsigned char letra;  
              long int numero1, numero2;

# Tipos de dados definidos no padrão ANSI

| Tipo               | Tamanho aproximado em bits | Faixa mínima                   |
|--------------------|----------------------------|--------------------------------|
| char               | 8                          | -127 a 127                     |
| unsigned char      | 8                          | 0 a 255                        |
| signed char        | 8                          | -127 a 127                     |
| int                | 16                         | -32.767 a 32.767               |
| unsigned int       | 16                         | 0 a 65.535                     |
| signed int         | 16                         | O mesmo que int                |
| short int          | 16                         | O mesmo que int                |
| unsigned short int | 16                         | 0 a 65.535                     |
| signed short int   | 16                         | O mesmo que short int          |
| long int           | 32                         | -2.147.483.647 a 2.147.483.647 |
| signed long int    | 32                         | O mesmo que long int           |
| unsigned long int  | 32                         | 0 a 4.294.967.295              |
| float              | 32                         | Seis dígitos de precisão       |
| double             | 64                         | Dez dígitos de precisão        |
| long double        | 80                         | Dez dígitos de precisão        |

Fonte: livro "C Completo e Total"



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

```
int main()
```

```
{
```

```
    char c;
```

```
    int i;
```

```
    short int si;
```

```
    unsigned int ui;
```

```
    long int li;
```

```
    float f;
```

```
    double d;
```

```
    printf("char %d \n", sizeof(c));
```

```
    printf("int %d \n", sizeof(i));
```

```
    printf("short int %d \n", sizeof(si));
```

```
    printf("unsigned int %d \n", sizeof(ui));
```

```
    printf("long int %d \n", sizeof(li));
```

```
    printf("float %d \n", sizeof(f));
```

```
    printf("double %d \n", sizeof(d));
```

```
    printf("double %d", sizeof(double));
```

```
}
```

Função que recebe a palavra reservada referente a um tipo primitivo (com ou sem modificador de tipo) ou um identificador de uma variável e retorna o número de bytes reservado na memória para o armazenamento da mesma.



# Conversão de Tipos

➤ **Implícita** → dos tipos menores para os tipos maiores

Exemplos:

char → int

int → long int

float → double

```
#include <stdio.h>
int main ()
{
    int a;
    char b;
    b='B';
    printf ("nº de bytes da variavel b = %d\n", sizeof (b));
    printf ("conteudo da variavel b = %d\n", b);
    a=b;
    printf ("nº de bytes da variavel a = %d\n", sizeof (a));
    printf ("conteudo da variavel a = %d\n", a);
}
```



# Conversão de Tipos

➤ **Explícita** → de tipos maiores para tipos menores, utilizando *cast* (modelador).

Exemplos:

int → char

double → float

```
#include <stdio.h>
int main ()
{
    int a;
    char b;
    a=120; /*120(DEC) == ...0000000001111000(BIN)*/
    printf ("nº de bytes da variavel a = %ld\n", sizeof (a));
    printf ("conteudo da variavel a = %d\n", a);
    b=(char)a; /*Sem perda de dados. Pois, 01111000(BIN) == 120(DEC)*/
    printf ("nº de bytes da variavel b = %ld\n", sizeof (b));
    printf ("conteudo da variavel b = %d\n", b);
}
```

```
#include <stdio.h>
int main ()
{
    int a;
    char b;
    a=365; /*365(DEC) == 101101101(BIN)*/
    printf ("nº de bytes da variavel a = %ld\n", sizeof (a));
    printf ("conteudo da variavel a = %d\n", a);
    b=(char)a; /*Com perda de dados. Pois, 01101101(BIN) == 109(DEC)*/
    printf ("nº de bytes da variavel b = %ld\n", sizeof (b));
    printf ("conteudo da variavel b = %d\n", b);
}
```

## Exercício

Construa um programa, na linguagem C, que receba através da entrada padrão um valor inteiro pertencente ao intervalo  $[0,255]$  e a posição do bit que se deseja saber o valor, o programa deve escrever na saída padrão o valor do bit solicitado.

```
#include <stdio.h>
int main ()
{
    int desloca, valor_byte, aux=1;
    printf("\nDigite um numero pertencente ao intervalo [0,255]: ");
    scanf("%d", &valor_byte);
    printf("\nDigite o bit a testar(0 a 7): ");
    scanf("%d", &desloca);
    aux = aux << desloca;
    valor_byte = valor_byte & aux;
    valor_byte = valor_byte >> desloca;
    printf("\nO valor do bit eh %d\n",valor_byte);
}
```

```
#include <stdio.h>
int main ()
{
    unsigned char desloca, valor_byte, aux=1;
    printf("\nDigite um numero pertencente ao intervalo [0,255]: ");
    scanf("%hhu", &valor_byte);
    printf("\nDigite o bit a testar(0 a 7): ");
    scanf("%hhu", &desloca);
    aux = aux << desloca;
    valor_byte = valor_byte & aux;
    valor_byte = valor_byte >> desloca;
    printf("\nO valor do bit eh %hhu\n", valor_byte);
}
```



# **Funções Matemáticas**

## **Operador Condicional**

# Funções Matemáticas

➤ `abs ()` /\*#include <math.h>\*/

➤ Calcula o valor absoluto de um inteiro.

➤ Sintaxe:

Tipo de retorno

int abs (int);

Parâmetros

Exemplo:

...

int i;

i = -8;

printf ("%d", abs(i));

/\*escreve na tela 8\*/



# Funções Matemáticas

➤ *fabs ()* /\*#include <math.h>\*/



➤ Calcula o valor absoluto de um valor real.

➤ Sintaxe:

```
float fabs (float);
```

Exemplo: ...

```
float f;  
f = -3.7;  
printf ("%f", fabs(f));  
/*escreve na tela 3.70000*/
```

# Funções Matemáticas

➤ *pow ()* /\*#include <math.h>\*/



➤ Potenciação  $x^y$ .

➤ Sintaxe:

double pow (double x, double y);

Exemplo:

```
...  
printf ("%0.2lf", pow(2, 3));  
/* escreve na tela 8.00 */  
printf ("%0.2f", (float)pow(2, 3));  
/* escreve na tela 8.00 */
```

# Funções Matemáticas

➤ `sqrt ()` /\*#include <math.h>\*/



➤ Raiz quadrada.

➤ Sintaxe:

```
double sqrt (double);
```

Exemplo: ...

```
printf ( "%.3lf", sqrt(9)); /* escreve na tela 3.000 */
```

```
printf ( "%d", (int)sqrt(9)); /* escreve na tela 3 */
```

# Funções Matemáticas

➤ `log ()` /\*#include <math.h>\*/



➤ Logaritmo natural.

➤ Sintaxe:

`double log (double);`

Exemplo:

`...`

`printf ("%0.1f", log(30));`

`/* escreve na tela 3.4 */`

# Funções Matemáticas

➤ `log10 () /*#include <math.h>*/`



➤ Logaritmo na base 10.

➤ Sintaxe:

```
double log10 (double);
```

Exemplo:

```
...
```

```
printf (".3lf", log10(30));
```

```
/* escreve na tela 1.477 */
```



# Funções Matemáticas

➡ `cos () /*#include <math.h>*/`



➡ Cosseno.

➡ Sintaxe:

`double cos (double x); /* x em radianos */`

Exemplo: ...

`printf (".2lf", cos(3.141592/4));`

`/* escreve na tela 0.71 */`

# Funções Matemáticas

➤ `sin ()` `/*#include <math.h>*/`



➤ Seno.

➤ Sintaxe:

`double sin (double x); /* x em radianos */`

Exemplo: ...

`printf (".2lf", sin(3.141592/3));`

`/* escreve na tela 0.87 */`



# Funções Matemáticas

➤ *tan ()* /\*#include <math.h>\*/



➤ Tangente.

➤ Sintaxe:

```
double tan (double x); /* x em radianos */
```

Exemplo: ...

```
printf (".2lf", tan(3.141592/5));
```

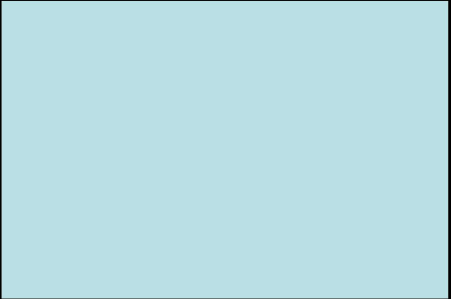
```
/* escreve na tela 0.73 */
```





## Exercício

Construa um programa, na linguagem C, que receba através da entrada padrão um valor real para  $x$  e avalie a expressão  $\sqrt{(3+x)^3 * 5}$ , apresentando o resultado da avaliação na saída padrão.



```
#include <stdio.h>
#include <math.h>
int main ()
{
    float x;
    printf("\nDigite um valor real para x: ");
    scanf("%f", &x);
    printf("\nO valor resultante da avaliacao da expressao eh: ");
    printf("%lf", sqrt(pow(3+x,3)*5));
}
```

# Operador Condicional

<expressão\_lógica>?<valo1>:<valor2>

ou

<expressão\_lógica>?<expressão1>:<expressão2>

# Operador Condicional

Exemplo: O programa abaixo recebe dois inteiros através da entrada padrão e retorna na tela o inteiro com maior valor dentre os dois fornecidos.

```
#include <stdio.h>
int main()
{
    int a,b;
    printf ("Entre com um valor inteiro: ");
    scanf ("%d",&a);
    printf ("\nEntre com outro valor inteiro: ");
    scanf ("%d",&b);
    printf ("\nO maior valor fornecido eh: %d",a>b?a:b);
}
```

## Exercício

Construa um programa, na linguagem C, que receba dois valores reais maiores que zero, através da entrada padrão, e determine qual o percentual do maior valor o menor valor representa. Apresente o resultado de seu cômputo na saída padrão.

```
#include <stdio.h>

int main ()
{
    float valorUm, valorDois, maior, menor;
    printf("Digite o primeiro valor real maior que zero: ");
    scanf("%f", &valorUm);
    printf("Digite o segundo valor real maior que zero: ");
    scanf("%f", &valorDois);
    maior = valorUm > valorDois ? valorUm : valorDois;
    menor = valorUm < valorDois ? valorUm : valorDois;
    printf("\n0 menor valor representa %.2f%% do maior valor fornecido.\n", menor*100/maior);
}
```

```
#include <stdio.h>
int main ()
{
    float valorUm, valorDois;
    printf("Digite o primeiro valor real maior que zero: ");
    scanf("%f", &valorUm);
    printf("Digite o segundo valor real maior que zero: ");
    scanf("%f", &valorDois);
    printf("\nO menor valor representa %.2f%% do maior valor fornecido.\n",
        valorUm > valorDois ? valorDois * 100 / valorUm : valorUm * 100 / valorDois);
}
```