



# Introdução à Programação

*Eduardo Silva Lira*

*XLVIII Programa de Verão do IME-USP*

*São Paulo - SP, Jan 2019*



INSTITUTO DE MATEMÁTICA E ESTATÍSTICA  
UNIVERSIDADE DE SÃO PAULO



# Revisão da aula anterior

- Tipos de dados em C
- Entrada

# Modificadores

- **unsigned**
  - Sem sinal
- **signed**
  - Com sinal
- **long**
- **short**

# Constantes

- Constantes:
  - Valores, guardados em um espaço de memória, que **não** devem ser **alterados durante a execução de um programa**.
  - Diferencie pelo padrão de nomenclatura:
  - UPPERCASE
  - UPPER\_CASE\_WITH\_UNDERSCORES
  - Exemplo (não é código fonte):
    - MAX\_LINHA = 1000
    - MIN\_IDADE = 18

# Constantes

- Constantes:
  - Valores, guardados em um espaço de memória, que **não** devem ser **alterados durante a execução de um programa**.
  - Utilize **#define** ou **const**

# Constantes

- Exemplo:

```
#include<stdio.h>

int main(){

    #define PI 3.1415
    int const MAIORIDADE = 18;

    printf("Agora posso utilizar constantes!\n%f\n%d\n\n", PI, MAIORIDADE);

    return 0;

}
```

# Dicas!

## Exemplo: macros para renomear comandos!

```
/*Definindo macros!*/
#include<stdio.h>

/*Definir macros para leitura e exibição*/
#define exiba printf
#define leia  scanf

/*Funcao principal*/
int main(){

    double preco;

    exiba("Digite o preco do produto: ");
    leia("%lf", &preco);

    exiba("Preco informado: R$ %.2f\n", preco);

    return 0; /*Execucao ocorreu normalmente*/
}
```

# Dicas para desenvolvedores

Como desenvolvedor de soluções, sempre pense:

O programa que estou desenvolvendo será utilizado por outras pessoas. Como posso deixá-lo da melhor maneira em relação a **usabilidade**?

Outros desenvolvedores poderão efetuar **manutenções** em seu código. Como posso deixar o código fonte da maneira mais clara possível?



# Dicas para desenvolvedores

Informações interessantes sobre formatadores da função printf:

<http://planeta-globo.com/2012/03/04/aprendendo-c-formatacao-do-printf/>

# Operadores Aritméticos

● Adição:  $+$

● Subtração:  $-$

● Multiplicação:  $*$

● Divisão:  $/$

# Operadores Aritméticos

● Adição: +

● Subtração: -

● Multiplicação: \*

● Divisão: /

- Inteira e ponto flutuante

4 / 3

5.5 / 2

# Operadores Aritméticos - Atenção

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

```
int main () {  
    double resultado;  
  
    resultado = 5 / 2;  
  
    printf("%f\n", resultado);  
  
    return 0;  
}
```

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

```
int main () {  
    double resultado;  
  
    resultado = 5.2 / 2;  
  
    printf("%f\n", resultado);  
  
    return 0;  
}
```

# Operadores Aritméticos - Atenção

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

```
int main () {  
    int valor1, valor2, resultado;  
  
    valor1 = 5;  
    valor2 = 2;  
  
    resultado = valor1 / valor2;  
  
    printf("%d\n", resultado);  
  
    return 0;  
}
```

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

```
int main () {  
    int valor1, valor2;  
    double resultado;  
  
    valor1 = 5;  
    valor2 = 2;  
  
    resultado = valor1 / valor2;  
  
    printf("%f\n", resultado);  
  
    return 0;  
}
```

# Operadores Aritméticos - Atenção

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

```
int main () {  
    double valor1, resultado;  
    int valor2;  
  
    valor1 = 5.0;  
    valor2 = 2;  
  
    resultado = valor1 / valor2;  
  
    printf("%f\n", resultado);  
  
    return 0;  
}
```

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

```
int main () {  
    double valor1, valor2;  
    int resultado;  
  
    valor1 = 11.0;  
    valor2 = 2.0;  
  
    resultado = valor1 / valor2;  
  
    printf("%d\n", resultado);  
  
    return 0;  
}
```

# Operadores Aritméticos

- Resto da divisão inteira: %
  - $13 \% 2$
  - $218 \% 10$
- Potenciação: sem moleza!
  - Por enquanto façam  $7*7*7*7*7*7$

# Type casting

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

```
int main () {
```

```
    int estoque, qtdeLojas;
```

```
    double media;
```

```
    scanf("%d%d", &estoque, &qtdeLojas);
```

```
    media = estoque / qtdeLojas;
```

```
    printf("%.4f\n", media);
```

```
    return 0;
```

```
}
```



# Type casting

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

```
int main () {
```

```
    int estoque, qtdeLojas;
```

```
    double media;
```

```
    scanf("%d%d", &estoque, &qtdeLojas);
```

```
    media = (double) estoque / qtdeLojas;
```

```
    printf("%.4f\n", media);
```

```
    return 0;
```

```
}
```

# Type casting

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

```
int main () {
```

```
    int estoque, qtdeLojas;
```

```
    double media;
```

```
    scanf("%d%d", &estoque, &qtdeLojas);
```

```
    media = (double) estoque / qtdeLojas;
```

```
    printf("%.4f\n", media);
```

```
    return 0;
```

```
}
```

# Type casting

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

```
int main () {
```

```
    double valor;
```

```
    scanf("%lf", &valor);
```

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

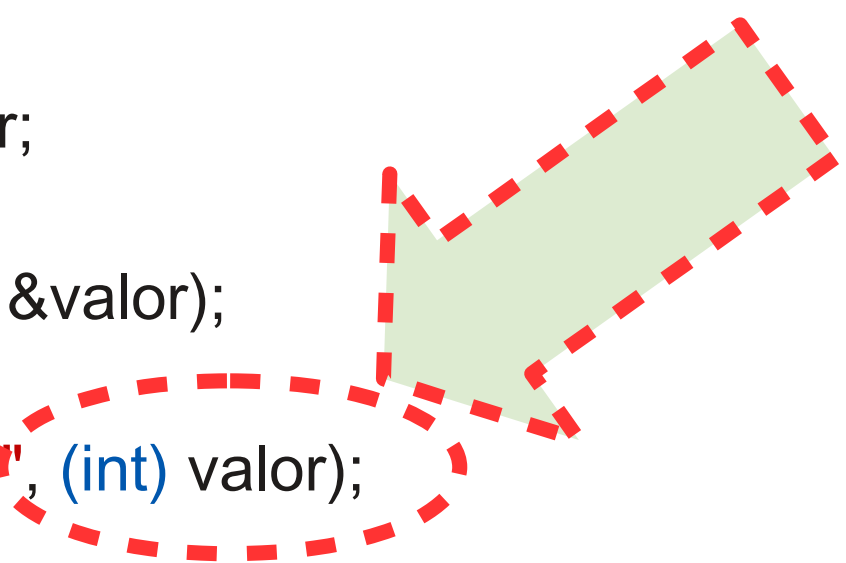
```
    return 0;
```

```
}
```

# Type casting

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

```
int main () {  
    double valor;  
  
    scanf("%lf", &valor);  
  
    printf("%d\n", (int) valor);  
  
    return 0;  
}
```



# Operadores Aritméticos

- Variantes com operador de **atribuição**
  - **`+=`**, **`-=`**, **`*=`**, **`/=`**, **`%=`**

Exemplos:

Atribuição normal	É equivalente a
<code>qtde += 1</code>	<code>qtde = qtde + 1</code>
<code>total -= desconto</code>	<code>total = total - desconto</code>
<code>res *= 3</code>	<code>res = res * 3</code>
<code>valor %= 2</code>	<code>valor = valor % 2</code>

# Dica! Mudança de **Config. Regionais**

- Possibilitar acentuação sem problemas
- Data, hora, padrão numérico, etc.
  - Seguirá o padrão definido pela função **setlocale**
  - É preciso incluir a biblioteca **locale.h**

# Dica! Mudança de Config. Regionais

```
1  /*Calcular a area do circulo - utilizar constantes!*/
2  #include<stdio.h>
3  #include<locale.h>
4
5  int main(){
6      const double PI = 3.14159;
7      double raio, area;
8
9      printf("Local do sistema: %s\n", setlocale(LC_ALL, ""));
10
11     printf("\nCALCULAR A ÁREA DO CIRCULO\n\n");
12     printf("Informe o raio: ");
13     scanf("%lf", &raio);
14
15     area = PI * raio * raio;
16
17     printf("A area é: %.2f\n", area);
18
19     return 0;
20 }
```

# Dica! Mudança de Config. Regionais

```
1  /*Calcular a area do circulo - utilizar constantes!*/
2  #include<stdio.h>
3  #include<locale.h>
4
5  int main(){
6      const double PI = 3.14159;
7      double raio, area;
8
9      printf("Local do sistema: %s\n", setlocale(LC_ALL, ""));
10
11     printf("\nCALCULAR A ÁREA DO CIRCULO\n\n");
12     printf("Informe o raio: ");
13     scanf("%lf", &raio);
14
15     area = PI * raio * raio;
16
17     printf("A area é: %.2f\n", area);
18
19     return 0;
20 }
```



# Dica! Mudança de **Config. Regionais**

- Caso o padrão do seu sistema não seja o Português do Brasil, tente:

```
setlocale(LC_ALL, "pt_BR")
```

ou

```
setlocale(LC_ALL, "Portuguese")
```

**Dúvidas?**

Leia um valor de ponto flutuante com duas casas decimais. Este valor representa um valor monetário. A seguir, calcule o menor número de notas e moedas possíveis no qual o valor pode ser decomposto. As notas consideradas são de 100, 50, 20, 10, 5, 2. As moedas possíveis são de 1, 0.50, 0.25, 0.10, 0.05 e 0.01. A seguir mostre a relação de notas necessárias.

## Entrada

O arquivo de entrada contém um valor de ponto flutuante **N** ( $0 \leq N \leq 1000000.00$ ).

## Saída

Imprima a quantidade mínima de notas e moedas necessárias para trocar o valor inicial, conforme exemplo fornecido.

Obs: Utilize ponto (.) para separar a parte decimal.

Exemplo de Entrada	Exemplo de Saída
576.73	NOTAS: 5 nota(s) de R\$ 100.00 1 nota(s) de R\$ 50.00 1 nota(s) de R\$ 20.00 0 nota(s) de R\$ 10.00 1 nota(s) de R\$ 5.00 0 nota(s) de R\$ 2.00 MOEDAS: 1 moeda(s) de R\$ 1.00 1 moeda(s) de R\$ 0.50 0 moeda(s) de R\$ 0.25 2 moeda(s) de R\$ 0.10 0 moeda(s) de R\$ 0.05 3 moeda(s) de R\$ 0.01