

# Introdução à Computação II – AULA 03

BCC Noturno - EMA896115B

Prof. Rafael Oliveira  
olivrap@gmail.com

Universidade Estadual Paulista  
“Júlio de Mesquita Filho”  
UNESP

Rio Claro 2014 (Sem 2)

# Revisão – Aula 02 – Hello World

```
#include <stdio.h>
int main()
{
    printf("Hello world\n");
}
```

## Revisão – Aula 02 Entrada/Saída

- `scanf(formato, &var1, &var2, ...);`
  - `scanf("%lf", &angulo);`
  - `scanf("%d %d %d", &dia, &mes, &ano);`
- `printf(formato, expr1, expr2, ...);`
  - `printf("O valor lido foi: %d\n", dia);`
  - `printf("O valor do seno de %lf eh %lf\n", x, sin(x));`

## Revisão – Aula 02

- Operadores aritméticos:  $((a + b) (c - d))/2$
- Operadores relacionais:  $a > b$
- Atribuição
  - $k = k + 1;$
  - $y = \sin(x) + \cos(x);$
  - $j = (\sin(x) == \cos(x)); // \text{ Faz sentido????}$

# Agenda – Aula 03

- Comandos de seleção em C (if)
  - Exercícios

## Comandos de seleção em C

- Execução sequencial:
  - Até agora, todos os comandos que vimos são executados em uma ordem pré-determinado
  - Um comando aparece escrito “em baixo do outro”
  - A ordem de execução dos comandos é aquela que aparece no texto do programa
  - Em alguns casos precisamos de estruturas mais elaboradas para expressar os nossos algoritmos

## Comandos de seleção em C

- Escrever um programa que leia dois números e diga qual deles é maior
  - a. Ler dois números em  $x$  e  $y$
  - b. Se  $x$  for maior escrever que o maior é  $x$
  - c. Caso contrário escrever que  $y$  é o maior

## Comandos if - exemplo

```
#include <stdio.h>
int main()
{
int x, y;
printf("Digite os dois numeros inteiros");
scanf("%d %d", &x, &y);
    if ( x > y )
        printf("O maior eh: %d", x);
    else
        printf("O maior eh: %d", y);
}
```

## Comandos if - exemplo

```
#include <stdio.h>
int main()
{
    int x, y, maior;
    printf("Digite os dois numeros inteiros");
    scanf("%d %d", &x, &y);
    if ( x > y )
        maior = x;
    else
        maior = y;
    printf("O maior eh: %d", maior);
}
```

# Estrutura do if

- If ( expressão )  
comando 1;  
else  
comando 2;
- if ( expressão )  
{  
    comando 1; comando2; ...  
}  
else  
{  
    comando 3; comando 4; ...  
}

# Comando if – exemplo (2)

```
int main()
{
    int x, y, maior, menor;
    printf("Digite os dois numeros inteiros");
    scanf("%d %d", &x, &y);

    if ( x > y )
    {
        maior = x;
        menor = y;
    }
    else
    {
        maior = y;
        menor = x;
    }
    printf("Maior: %d -- Menor: %d", maior, menor);
}
```

## Comandos if aninhados

- Dentro de um comando if pode aparecer qualquer comando, inclusive outro if
- Ler três números e dizer qual é o maior deles

# if aninhado – exemplo

```
int x, y, z, maior;
printf("Digite os 3 numeros inteiros");
scanf("%d %d %d", &x, &y, &z);
if ( x > y )
{
    if ( z > x )
        maior = z;
    else
        maior = x;
} else
{
    if ( z > y )
        maior = z;
    else
        maior = y;
} printf("O maior eh : % d\n", maior);
```

## Bom hábito - Indentação

- A forma como os comandos são organizados na forma de um texto é completamente livre
- Para que se possa entender o código de um programa, existem algumas regras que devem ser seguidas
- Elas não mudam o significado do programa mas servem para embelezá-lo e deixá-lo mais inteligível

## Bom hábito - Indentação

- Declaração de função deve ser feita na primeira coluna
- Chaves devem estar na primeira coluna, em linhas sozinhas
- Comandos não devem aparecer na mesma linha das chaves

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

```
int main()
```

```
{
```

```
    // Comandos aqui
```

```
}
```

## Bom hábito - Indentação

- Declaração de variáveis também deve aparecer na primeira coluna
- Deixe uma linha em branco após as declarações

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

```
int main()
```

```
{
```

```
int i, j, k;
```

```
double x, y, z;
```

```
    // comandos aqui
```

```
}
```

## Bom hábito - Indentação

- Os comandos devem estar afastados algumas posições à direita das chaves
- Todos devem iniciar na mesma coluna

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

```
int main()
```

```
{
```

```
printf(".... ", ...);
```

```
...
```

```
scanf("....", ...);
```

```
}
```

# Bom hábito - Indentação

- Mesmo vale para if aninhado

```
int main()
{
    printf(".... ", ...);
    if ( a > b )
    {
        if ( a > c )
        {
            maior = ac
        }
        else
        {
            maior = c;
        }
    } else...
}
```

## If sem Else

- *A parte correspondente ao else pode não estar presente, ou seja, ela é opcional*
- *Corresponde a um comando “se acontecer tal condição execute tal comando”*
- *Se essa condição não acontecer, nada deve ser feito*
- *Exemplo: se o valor do ângulo for negativo, ajustá-lo para um valor positivo*

# If sem Else - Exemplo

```
#include <stdio.h>
#include <math.h>
int main()
{
    double angulo;
    int sinal, voltas;

    printf("Digite o valor do angulo ==> ");
    scanf("%lf", &angulo);

    // quantas voltas da no circulo
    voltas = (int) (angulo / (2 * M_PI));
    // valor alem da ultima volta
    angulo = angulo - (voltas * 2 * M_PI);
    sinal = 1;
    if ( angulo < 0 )
    { // -sin(x) ´e igual sin(-x)
        sinal = -1;
        angulo = -angulo;
    }
    printf("O valor do seno eh %lf\n", sinal *sin(angulo));
}
```

# Coersão de tipos

- Às vezes é necessário transformas um valor de um tipo para outros
- No exemplo anterior, de double para int
- Isso é feito colocando o tipo para o qual se deseja transformar, entre parênteses, antes do valor que se deseja transformar
  - $voltas = (int) (angulo / (2 * M\_PI));$
  - $ang\_rad = ( (float) ang\_grau ) * M\_PI / 180 )$

# Exercícios

01- Faça um programa que compute as soluções da equação  $Ax^2 + Bx + C = 0$ . Os valores de A, B e C devem ser fornecidos pelo usuário. Se o valor de A for 0, então o programa deve emitir uma mensagem de erro dizendo que a equação não é do segundo grau. Se o valor de “Delta” for negativo, o programa deve emitir uma outra mensagem de erro, dizendo que não existem raízes reais para a equação.

# Exercícios – 01 Solução

```
#include <stdio.h>
#include <math.h>
int main()
{
double delta, x1, x2;
double A, B, C;

printf("Resolva a equacao Ax**2 + Bx + C = 0\n");
printf("Entre com o valor de A ==> ");
scanf("%lf", &A);
printf("Entre com o valor de B ==> ");
scanf("%lf", &B);
printf("Entre com o valor de C ==> ");
scanf("%lf", &C);
```

# Exercícios – 01 Solução

```
{
    printf("Essa nao eh uma equacao do segundo grau\n");
}else
{
    delta = B * B - (4 * A * C);
    if ( delta < 0 )
    {
        printf("Essa equacao nao tem raiz real\n");
    }else
    {
        x1 = -B + sqrt(delta) / (2 * A) ;
        x2 = -B - sqrt(delta) / (2 * A) ;
        printf("Solucoes: %lf e %lf\n", x1, x2);
    }
}
```

# Para facilitar

- O comando return pode ser usado para terminar a execução da função main
- Isso faz com que o programa termine
- Cuidado, se estiver usando o “pause” para visualizar o resultado
- Ele deve ser usado em cada um dos comandos return
- O return deve ter um valor que é retornado para o S.O
  - 0 - ok
  - outro valor - erro

# Uso do Return

```
if ( A == 0 )
{
    printf("Essa nao eh uma equacao do segundo grau\n");
    return 0;
}
delta = B * B - (4 * A * C);
if ( delta < 0 )
{
    printf("Essa equacao nao tem raiz real\n");
    return 0;
}
x1 = -B + sqrt(delta) / (2 * A) ;
x2 = -B - sqrt(delta) / (2 * A) ;
printf("Solucoes: %lf e %lf\n", x1, x2);
```

# Uso do Return

```
if ( A == 0 )
{
    printf("Essa não é uma equação do segundo grau\n");
    system("pause");
    return 0;
}
delta = B * B - (4 * A * C);
if ( delta < 0 )
{
    printf("Essa equação não tem raiz real\n");
    system("pause");
    return 0;
}
x1 = -B + sqrt(delta) / (2 * A) ;
x2 = -B - sqrt(delta) / (2 * A) ;
printf("Soluções: %lf e %lf\n", x1, x2);
```

# Operadores Lógicos

- São feitos para “combinar” expressões lógicas
- **Operador && (AND)**
  - Resultado 1 somente se os dois operandos são diferentes de 0
  - Caso contrário, resultado é 0
  - `if ( a > b && b > c ) maior = a;`
- **Operador || (OR)**
  - Resultado 0 somente se os dois operandos são 0
  - Caso contrário, resultado é 1
  - `if ( mes < 1 || mes > 12 )`  
`printf("Mes invalido\n");`
- **Operador ! (NOT)**
  - O resultado é 1 se operando é 0.
  - Caso contrário é 0
  - `if ( ! ( mes < 1 || mes > 12 ) )`
  - `printf("Mes eh valido\n");`

# Exercícios

02- Faça um programa que leia 2 notas de um aluno, verifique se as notas são válidas e exiba na tela Se o aluno foi aprovado, reprovado, ou se ficou de REC. Uma nota válida deve ser obrigatoriamente um valor entre 0.0 e 10.0. Caso a nota não possua um valor válido, este fato deve ser informado ao usuário e o programa termina. Escreva duas versões desse programa. Uma usando o operador `&&` para verificar se uma nota é válida ou não e outra usando o operador `||`.

# Exercícios – solução 01

```
int main()
{
double nota1, nota2, media;
    printf("Digite o valor da nota 1 ==> ");
    scanf("%lf", &nota1);
    if ( nota1 < 0.0 || nota1 > 10.0 )
    {
        printf("Nota nao eh valida\n");
        return 0;
    }
    printf("Digite o valor da nota 2 ==> ");
    scanf("%lf", &nota2);
    if ( nota2 < 0.0 || nota2 > 10.0 )
    {
        printf("Nota nao eh valida\n");
        return 0;
    }
}
```

## Exercícios – solução 02 – cont ...

```
media = (nota1 + nota2) / 2.0;
if ( media >= 5.0 )
{
    printf("Aluno aprovado\n");
}
else
{
    if ( media >= 3 )
        printf("Aluno de REC\n");
    else
        printf("Aluno reprovado\n");
}
return 0;
}
```

# Exercícios – solução 02 – outra

Outra solução ...

```
int main()
{
double nota1, nota2, media;
    printf("Digite o valor da nota 1 ==> ");
    scanf("%lf", &nota1);
    if ( ! (nota1 >= 0.0 && nota1 <= 10.0 ) )
    {
        printf("Nota nao eh valida\n");
        return 0;
    }
    printf("Digite o valor da nota 2 ==> ");
    scanf("%lf", &nota2);
    if ( ! (nota2 >= 0.0 && nota2 <= 10.0 ) )
    {
        printf("Nota nao eh valida\n");
        return 0;
    }
}
```

## Exercícios

03- Uma empresa decide dar um aumento aos seus funcionários de acordo com uma tabela que considera o salário atual e o tempo de serviço de cada funcionário. Os funcionários com menor salário terão um aumento proporcionalmente maior do que os funcionários com um salário maior, e conforme o tempo de serviço na empresa, cada funcionário irá receber um bônus adicional de salário. (continua ...)

# Exercícios

03- Faça um programa que leia: (1) o valor do salário atual do funcionário; (2) o tempo de serviço deste funcionário na empresa (nro. de anos de trabalho na empresa). Use as tabelas abaixo para calcular o salário reajustado deste funcionário e imprima o valor do salário final reajustado, ou uma mensagem caso o funcionário não tenha direito a nenhum aumento.

Sal atual	Reajuste	Tempo	Bonus
Até 500,00	25%	menos 1 ano	Nada
Até 1000,00	20%	1 a 3 anos	100,00
Até 1500,00	15%	4 a 6 anos	200,00
Até 2000,00	10%	7 a 10 anos	300,00
Mais de 2000,00	Nada	Mais de 10 anos	500,00