

Curso de Programação em Computadores V

Modulo 2 – Leitura, Escrita e IF

Aura & Erick
aconci@ic.uff.br, erickr@id.uff.br

Roteiro

Ciclo de um programa

IDE - ambiente de programação C (++, #)

- ▶ **Introdução**
 - ▶ Estrutura básica
 - ▶ Diretivas
- ▶ **Componentes**
 - ▶ Declaração
 - ▶ Atribuição
 - ▶ Entrada e Saída
 - ▶ Operadores
 - ▶ Funções Matemáticas

Ciclo de Vida de um Programa

- ▶ Especificação de requisitos (Entradas/Saídas/ o que o programa deve fazer)
- ▶ Algoritmo (solução)
- ▶ Testes
- ▶ Programação
- ▶ Testes
- ▶ Manutenção



Teste

- ▶ **Todo algoritmo deve ser testado**
 - ▶ Usar dados e resultados previamente calculados
 - ▶ Seguir precisamente as instruções do algoritmo
 - ▶ Verificar se o procedimento está correto
- ▶ Exemplo: Fazer teste de mesa (chines) para o algoritmo da média (nota máxima = 10)

P1	P2	P3	P4	Média

Introdução - Estrutura Básica

- ▶ diretivas para o pré-processador
- ▶ declaração de variáveis globais

main ()

{

declaração de variáveis locais da função **main**
comandos da função **main**

}

Meu Primeiro Programa em C

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

```
int main()
{
    printf("Alô! \n");
    system("pause");
}
```



Como Fazer o Computador Executá-lo?



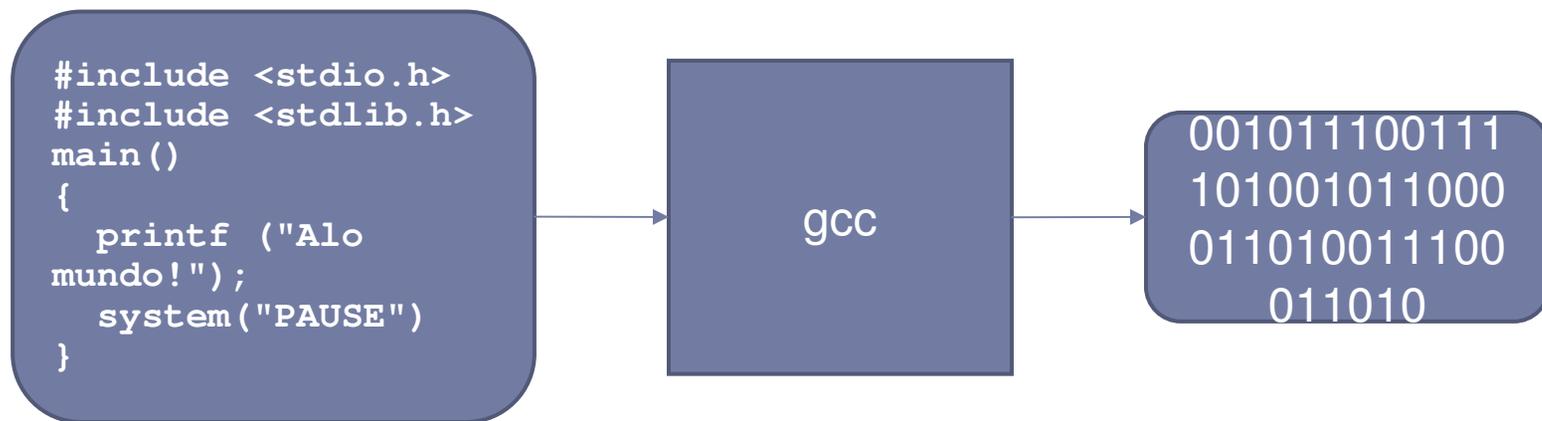
Por que Preciso Compilar?

- ▶ Computador só “entende” zeros e uns...



Por que Preciso Compilar?

- ▶ Computador só “entende” zeros e uns...



Integrated Development Environment (IDE) e Compiladores



codeblocks

- ▶ Esta disponível no Lab do IC:
- ▶ veja o vídeo gravado pelo Erick de como usá-lo em:
- ▶
- ▶ <https://www.youtube.com/watch?v=BYpf87nQ1Mw>
- ▶
- ▶ Só que a gente só vai precisar usar o laboratório caso tenhamos **algum aluno que não consiga instalar os compiladores em sua casa** ou em uma aula/prova prática (VS) lá.
- ▶ As provas P1 e P2 vão ser apenas perguntas sobre a linguagem, supondo que você esta fazendo os programas rodarem em sua casa nos exercícios.



Integrated Development Environment (IDE) e Compiladores

- ▶ *No Visual Studio, crie um novo projeto como no tutorial:*
 - ▶ https://youtu.be/VKds2loxc_U
 - ▶ <https://www.youtube.com/watch?v=u60ABTDYyNc>

- ▶ Ou, inicie o Dev-C++ pelo menu
- ▶ Crie um novo arquivo, com o comando *File, New Source File*

Edite o “Meu Primeiro Programa em C”

Visual Studio Code

Entrada (1,1 x) IADES - Inst x www2.ic.ufi x Google Tra x Nothing Bu x Spotify Play x Visual Studi x Downloads x Download x visual studi x most used x read value x Ambiente d x

Visual Studio Code Docs Updates Connect Gallery FAQ Search

VS Code Beta is available. Check out the [new features](#) and [update](#) it now.

Code focused development. Redefined.

Build and debug modern web and cloud applications. Code is free and available on your favorite platform - Linux, Mac OSX, and Windows.

[Download for Windows](#) [Download for Linux x64](#) [Download for OS X](#)

32bit version

By downloading and using Visual Studio Code, you agree to the [license terms](#) and [privacy statement](#) for Visual Studio Code. When this tool crashes, we automatically collect crash dumps so we can figure out what went wrong. If you don't want to send us crash dumps, go [here](#) to learn how to disable them.

Pesquisar na Web e no Windows

20:38 30/11/2015

→ Vantagem: Gratuito e multiplataforma

IDE e Compiladores

- ▶ Visual Studio != Visual Studio Code
 - ▶ O visual studio community é gratuito (apenas Windows):
 - ▶ <https://www.visualstudio.com/pt-br/products/visual-studio-community-vs.aspx>
 - ▶ Diversos compiladores e IDEs podem ser utilizados, o requerimento é compilar código na linguagem C/C#/C++.
 - ▶ Também é possível utilizar o Visual Studio em outros sistemas operacionais, utilizando máquinas virtuais.
-

O Visual Studio

- ▶ A ferramenta Visual Studio:

- ▶ <https://visualstudio.com/en-us/downloads/download-visual-studio-vs.aspx>

é um ambiente de desenvolvimento de programas em C , C# e C++ (editor, compilador, bibliotecas...)

- ▶ Pode ser baixado no site acima

- ▶ Vamos criar apenas programas para console, na linguagem C (C++ e C#). O vídeo explica isso:

- ▶ https://www.youtube.com/watch?v=VKds2lox_c_U&feature=youtu.be

Mas há diversas outras formas

- ▶ Da linguagem que você vai poder usar igualmente depois

quando já estiver mais familiarizado:

- ▶ https://www.youtube.com/watch?v=-CKSX5H_vK0



C

- ▶ é uma linguagem de programação compilada de propósito geral, estruturada, procedural, padronizada pela ISO, criada em 1972, por Dennis Ritchie, no AT&T Bell Labs, para desenvolver o sistema operacional Unix (que foi originalmente escrito em Assembly).
 - ▶ é uma das linguagens de programação mais populares e existem poucas arquiteturas para as quais não existem compiladores para C.
 - ▶ C tem influenciado muitas outras linguagens de programação, mais notavelmente C++, que originalmente começou como uma extensão para C.
-



C++



- ▶ (em lê-se "cê mais mais", ou *see plus plus*) é uma linguagem de programação orientada a objeto e de uso geral.
- ▶ A linguagem é considerada de médio nível, pois combina características de linguagens de alto e baixo níveis.
- ▶ Desde os anos 1990 é uma das linguagens comerciais mais populares, sendo bastante usada por seu grande desempenho.



C Sharp



- ▶ Alguns pensam que o nome C# viria duma sobreposição de quatro símbolos +, dando a impressão de +++, uma alusão à continuação do C++.
- ▶ Entretanto, o # de C# se refere ao sinal musical sustenido (#), pronunciado *sharp* em inglês, que aumenta em meio tom uma nota musical!

C#, C Sharp (em português lê-se "cê charp"), é uma linguagem de programação interpretada fortemente tipada, e, possuindo paradigmas de programação funcional, declarativa, orientada a objetos e genérica. Foi desenvolvida pela Microsoft .

A sua sintaxe orientada a objetos foi baseada no C++ mas inclui muitas influências de outras linguagens de programação, como Java .



Esse mesmo símbolo em C

- ▶ Sempre se referiu aos comandos do Pré processador ,
- ▶ Que sempre iniciam por #

- ▶ No módulo passado vimos o **# define** (para as constantes)



Introdução – Diretivas

- ▶ Diretivas para o pré processador - Bibliotecas
- ▶ Diretiva **#include** permite incluir uma biblioteca
- ▶ Bibliotecas contêm funções pré-definidas, utilizadas nos programas

Diretiva

`#include <stdio.h>`

`#include <stdlib.h>`

`#include <math.h>`

`#include <system.h>`

`#include <string.h>`

Conteúdo

Funções de entrada e saída

Funções padrão

Funções matemáticas

Funções do sistema

Funções de texto

Meu Segundo Programa em C

```
#include <stdio.h>
#include <stdlib.h>
main()
{
    printf ("Meu nome eh ");
    printf ("Beatriz \n");
    system("PAUSE");
}
```



Exercicio

- ▶ Compilem e executem o programa 1 e 2 anteriores deste módulo,
- ▶ Se der **errado** em algo use **as dicas que o Erick** passou no vídeo do *youtube*, por exemplo ou procure ajuda com ele por e-mail.



Dicas do **Erick**

- ▶ Termine todas as linhas com ;
- ▶ **Sempre salve** o programa antes de compilar
- ▶ Sempre **compile** o programa antes de executar
- ▶ Quando ocorrer um erro de compilação, **deixe o mouse sobre a mensagem de erro para destacar o comando errado no programa**
- ▶ Verifique também a linha anterior, que pode ser a responsável pelo erro, especialmente se faltar o ;
- ▶ **Use comentários explicando partes do seu programa, iniciados por //**

Exemplo de comentários

```
//diretivas para o pré-processador
#include <stdio.h>
#include <stdlib.h>
//declaração de variáveis globais

//programa principal
main()
{
    printf ("Alo mundo!");
    system("PAUSE");
}
```

Declarações de Variáveis

- ▶ Declaram as variáveis e seus tipos
- ▶ Os nomes das variáveis devem conter apenas **letras**, **dígitos** e o símbolo **_**
- ▶ Os principais tipos são: **int**, **float**, **double** e **char**
- ▶ Exemplos

```
int n;  
int quantidade_valores;  
float x, y, somaValores;  
char sexo;  
char nome[40];
```

C diferencia letras
maiúsculas de
minúsculas!

int, N → n é diferente de N!

Exemplo

Algoritmo Soma

1. Leia A (inteiro)
2. Leia B (inteiro)
3. SOMA = A + B
4. Escreva SOMA

Programa em C

```
#include <stdio.h>
#include <stdlib.h>
main()
{
    int A, B, SOMA;
    scanf ("%d %d", &A, &B);
    SOMA = A + B;
    printf ("A soma eh %d",
    SOMA);
    system("PAUSE");
}
```

Case Sensitive = É Como se Chama

Isso do C

- ▶ **diferenciar letras maiúsculas de minúsculas!**
- ▶ **int A, B, SOMA**
- ▶ **Não é igual a:**
- ▶ **int a, b, Soma, sOMA,**
- ▶ **SomA, sOmA, SoMa, etc...**



Atribuição

- ▶ Atribui o valor da direita à variável da esquerda
- ▶ O valor atribuído pode ser uma **constante**, uma **variável** ou uma **expressão**
- ▶ Exemplos

`x = 4;` **--> vemos x recebe 4**

`y = x + 2;`

`y = y + 4;`

`valor = 2.5;`

`sexo = 'F';`

Entrada e Saída

▶ Função **scanf**

```
scanf ("formatos", &var1, &var2, ...)
```

Exemplos:

```
int i, j;  
float x;  
char c;  
char nome[40];  
scanf ("%d", &i);  
scanf ("%d %f", &j, &x);  
scanf ("%c", &c);  
scanf ("%s", &nome);
```

%d	inteiro
%f	float
%lf	double
%c	char
%s	string (palavra / texto)

Exemplo

Algoritmo Soma

1. Leia A (inteiro)
2. Leia B (inteiro)
3. SOMA = A + B
4. Escreva SOMA

Programa em C

```
#include <stdio.h>
#include <stdlib.h>
main()
{
    int A, B, SOMA;
    scanf ("%d %d", &A, &B);
    SOMA = A + B;
    printf ("A soma eh %d",
    SOMA);
    system("PAUSE");
}
```

Operadores Matemáticos

Operador	Exemplo	Comentário
+	$x + y$	Soma x e y
-	$x - y$	Subtrai y de x
*	$x * y$	Multiplica x por y
/	x / y	Divide x por y
%	$x \% y$	Calcula o resto da divisão de x por y
++	x++	Soma 1 ao valor de x
--	x--	Subtrai 1 do valor de x



Entrada e Saída

▶ Função **printf**

```
printf ("formatos", var1, var2, ...)
```

Exemplos:

```
int i, j;  
float x;  
char c;  
char nome[40];  
printf("%d", i);  
printf("%d, %f", j, x);  
printf("%c", c);  
printf("%s", nome);
```

%d	inteiro
%f	float
%lf	double
%c	char
%s	string (palavra / texto)

Exemplo números inteiros



Mãos à obra!

Algoritmo Soma

1. Leia A (inteiro)
2. Leia B (inteiro)
3. SOMA = A + B
4. Escreva SOMA

Programa em C

```
#include <stdio.h>
#include <stdlib.h>
main()
{
    int A, B, SOMA;
    scanf ("%d %d", &A, &B);
    SOMA = A + B;
    printf ("A soma eh %d",
SOMA);
    system("PAUSE");
}
```

Exemplos Equivalentes números reais

```
#include <stdio.h>
#include <stdlib.h>
main()
{
    double n1, n2, n3, media;
    scanf ("%lf",&n1);
    scanf ("%lf",&n2);
    scanf ("%lf",&n3);
    media=(n1+n2+n3)/3;
    printf ("%lf",media);
    system("PAUSE");
}
```

```
#include <stdio.h>
#include <stdlib.h>
main()
{
    double n1, n2, n3, media;
    scanf ("%lf %lf %lf", &n1, &n2, &n3);
    media=(n1+n2+n3)/3;
    printf ("%lf",media);
    system("PAUSE");
}
```

Exercício

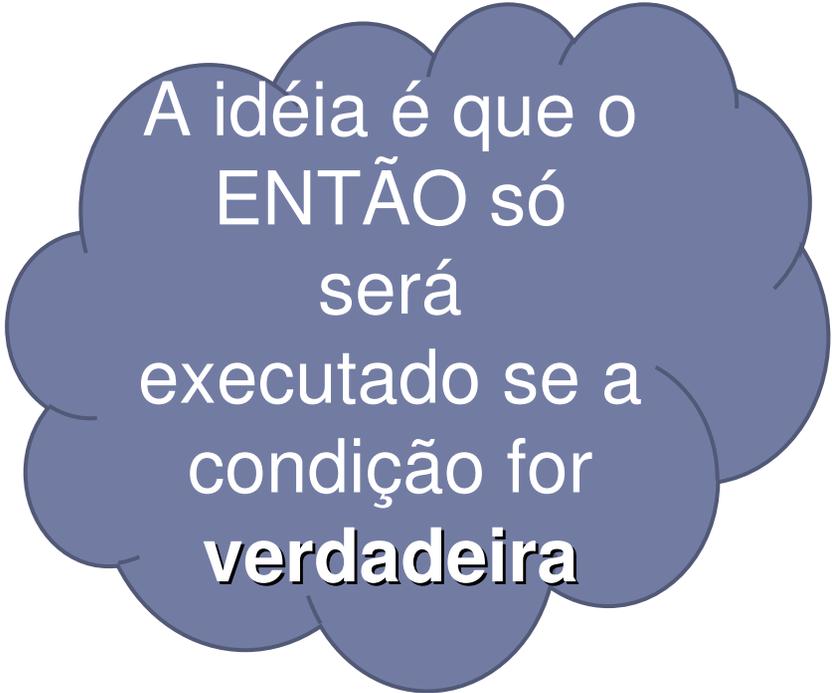
- ▶ Imprimir o valor em reais de uma compra feita em dólares
 - ▶ Dados de entrada?
 - ▶ Dados de saída?
- ▶ Façam primeiro o algoritmo (teste ele) e depois o programa em C correspondente!

Estruturas de Decisão

Problema: Ler 3 números e descobrir qual é o maior dos 3

Algoritmo:

1. Ler N1, N2, N3
2. MAIOR = N1
3. Se $N2 > \text{MAIOR}$
Então MAIOR = N2
1. Se $N3 > \text{MAIOR}$
Então MAIOR = N3
1. Escreva MAIOR



A idéia é que o
ENTÃO só
será
executado se a
condição for
verdadeira



Tipo de dado Booleano:

Em Computação, **booleano** é um tipo de dado que apenas possui dois valores, que podem ser considerados como 0 ou 1, falso ou verdadeiro.

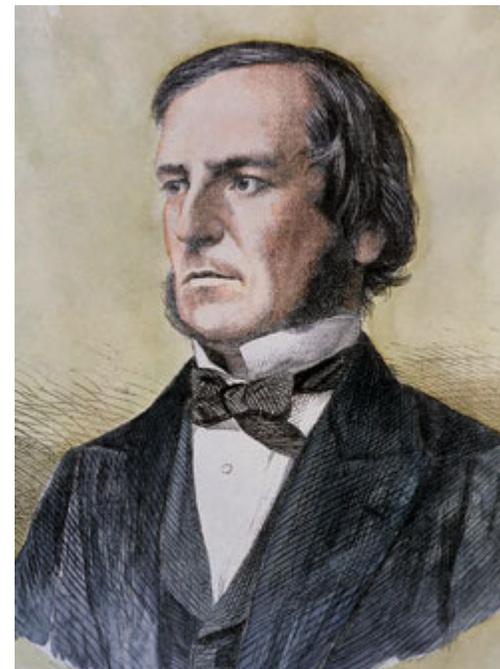
Esses dados são chamados **booleano** em homenagem a **George Boole**, que definiu um sistema de lógica algébrica pela primeira vez na metade do século XIX.

Ele é usado em operações lógicas como AND, OR, NOT , ou outras que correspondem a algumas das operações da **álgebra booleana**.



George Boole

(1815 – 1864) Filósofo britânico , foi criador da álgebra que usa as operações de **e**, **ou**, **ou exclusivo**, **não** , **verdadeiro** e **falso** , que é fundamental para o desenvolvimento da computação moderna.



Desvio Condicionais

- ▶ A instrução ou comando **if** (ou **se**) é bastante utilizado na programação.
- ▶ Ele avalia um valor lógico (T ou F) e, com base nisso, decide se entra ou não no bloco de código.

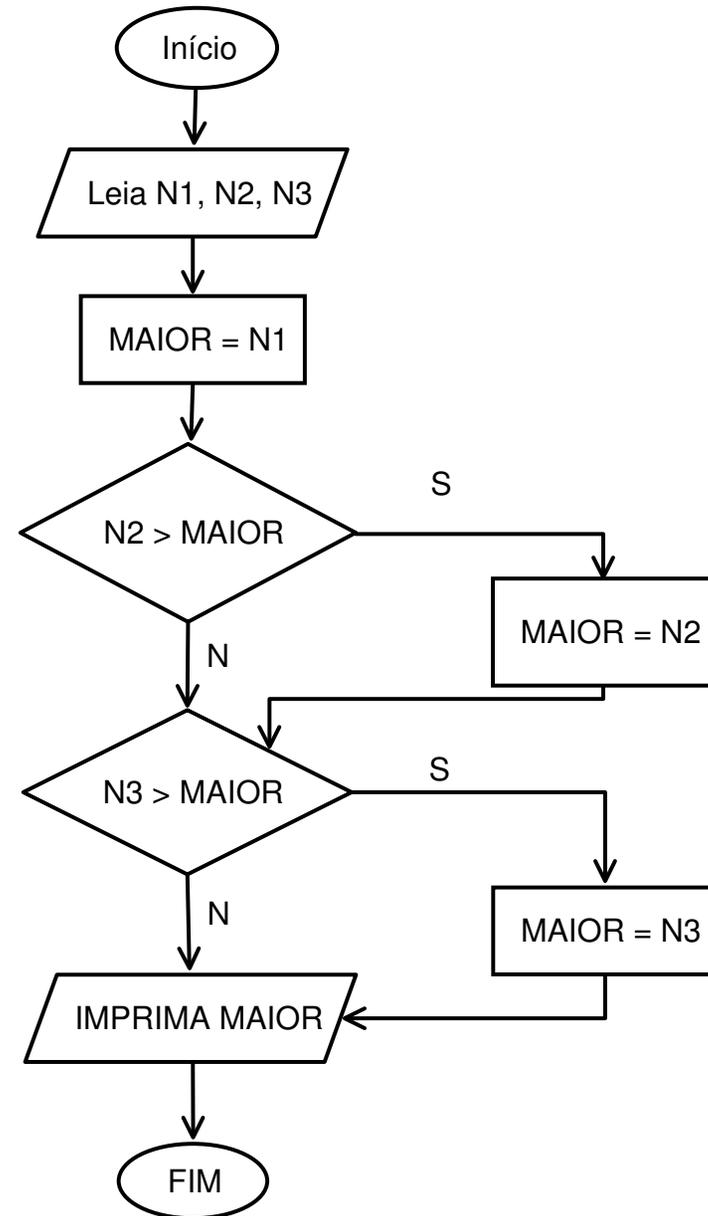
- ▶ Suponha que **a** seja dado Booleano;
- ▶ **if (a){**
 - ▶ *//se “a” for igual a **T=true**, os comandos dentro dos colchetes serão executados*
- ▶ **}**
- ▶ *//se “a” for **F=false**, o algoritmo passa direto para este ponto do programa*



Fluxograma

Algoritmo:

1. Ler N1, N2, N3
2. MAIOR = N1
3. Se $N2 > \text{MAIOR}$
Então MAIOR = N2
1. Se $N3 > \text{MAIOR}$
Então MAIOR = N3
1. Escreva MAIOR



Em C

if **EXPR. LÓGICA**
COMANDO;

Exemplo:

```
If N2 > MAIOR  
    MAIOR = N2;
```

```
#include <stdio.h>  
void main ()  
{  
    int N1, N2, N3, MAIOR;  
    printf ("Digite tres inteiros: ");  
    scanf ("%d %d %d", &N1, &N2, &N3);  
    MAIOR = N1;  
    if (N2 > MAIOR)  
        MAIOR = N2;  
    if (N3 > MAIOR)  
        MAIOR = N3;  
    printf ("Maior dos tres: %d", MAIOR);  
    system("PAUSE");  
}
```



Comando Composto

- ▶ E se eu precisar executar mais de um comando no IF?
- ▶ Usar comando composto

Ler A, B

Ler OP

Se $OP == 1$

Então $R = A + B$

 Escrever (“SOMA”, R)

Se $OP == 2$

Então $R = A - B$

 Escrever (“SUBTR.”, R)

Escrever A, B



Comando Composto

Ler A, B

Ler OP

Se $OP == 1$

Então $R = A + B$

 Escrever (“soma”, R)

Se $OP == 2$

Então $R = A - B$

 Escrever (“subtracao.”, R)

Escrever A, B

```
#include <stdio.h>
void main ()
{
    int A, B, R, OP;
    printf ("Digite dois inteiros: ");
    scanf ("%d %d", &A, &B);
    printf ("Digite 1 para SOMAR ou 2 para
        SUBTRAIR)
    scanf ("%d", OP);
    if OP ==1 {
        R = A + B;
        printf ("soma", R);
    }
    if OP ==2 {
        R = A - B;
        printf ("subtracao", R);
    }
    system("PAUSE");
}
```



Operadores Relacionais

Operador	Exemplo	Comentário
==	$x == y$	O conteúdo de x é igual ao de y
!=	$x != y$	O conteúdo de x é diferente do de y
<=	$x <= y$	O conteúdo de x é menor ou igual ao de y
>=	$x >= y$	O conteúdo de x é maior ou igual ao de y
<	$x < y$	O conteúdo de x é menor que o de y
>	$x > y$	O conteúdo de x é maior que o de y

- ▶ As expressões relacionais em C retornam
 - ▶ 1 se verdadeiro
 - ▶ 0 se falso



Operadores Lógicos

- ▶ **&& (E lógico)**: retorna verdadeiro se ambos os operandos são verdadeiros e falso nos demais casos.
Exemplo: `if(a>2 && b<3)`.
- ▶ **|| (OU lógico)**: retorna verdadeiro se um ou ambos os operandos são verdadeiros e falso se ambos são falsos.
Exemplo: `if(a>1 || b<2)`.
- ▶ **! (NÃO lógico)**: usada com apenas um operando.
Retorna verdadeiro se o operando é falso e vice-versa.
Exemplo: `if(!var)`.

Operadores Lógicos

A	B	A && B
T	T	T
T	F	F
F	T	F
F	F	F

A	B	A B
T	T	T
T	F	T
F	T	T
F	F	F

A	!A
T	F
F	T



Exercícios

- ▶ Fazer exercícios
 - ▶ Comentados ao longo do modulo 1 e 2
- ▶ Entregar o exercício abaixo codificado em C,
- ▶ Ele será o trabalho 2
 - ▶ Peça 4 números inteiros de entrada, N1, N2, N3 e N4, escreva na tela qual é a maior soma possível de dois desses 4 números.
 - ▶ ***E.g.:*** Se $N1 = 2$, $N2 = 4$, $N3 = 1$ e $N4 = 10$, a maior soma de dois elementos é 14, ou seja, $N2 + N4$.

Nossa nota T

- ▶ Esta relacionada a fazer vários exercícios e entregar um para a aula seguinte!
- ▶ A agora é o melhor tempo já que vocês não tem muitas cadeiras cobrando e não acumulam a matéria.
- ▶ Assim a data de entrega será 22/12 (terça feira).
- ▶ Mandem ele (código e executável renomeando .exe para .trab2) por e-mail para o Erick.
- ▶ Se tiverem dúvidas entre em contato com ele imediatamente.
- ▶ Na subject da e-mail - incluir PROG V - TRAB 2

