



Aula 04: Fluxogramas e condicionais

Introdução a Programação

Túlio Toffolo & Puca Huachi
<http://www.toffolo.com.br>

Aula Anterior

- Introdução ao C/C++
 - Variáveis
 - Operadores aritméticos
 - Comandos de entrada/saída

Aula de Hoje

- 1 Fluxogramas
- 2 Biblioteca `<math.h>`
- 3 Operadores relacionais
- 4 Operadores lógicos
- 5 Comandos de decisão
- 6 Exercícios
- 7 Próxima Aula

Aula de Hoje

- 1 Fluxogramas
- 2 Biblioteca `<math.h>`
- 3 Operadores relacionais
- 4 Operadores lógicos
- 5 Comandos de decisão
- 6 Exercícios
- 7 Próxima Aula

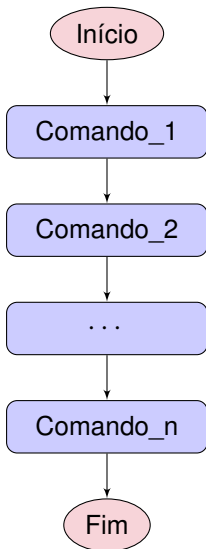
Estrutura básica de um programa em C/C++

```
1 <inclusão_de_bibliotecas>
2
3 int main(<declaração_dos_parâmetros>)
4 {
5     instrução_1;
6     instrução_2;
7     instrução_3;
8     ...
9     instrução_n;
10
11     return 0;
12 }
```

Fluxogramas

- Os fluxogramas são representações gráficas dos programas.
- São utilizados para nos ajudar a compreender um programa.
- Não estão associados a um linguagem específica.
- Apresentam a lógica do algoritmo e não as instruções da linguagem.
- Utilizam diferentes tipos de blocos para indicar os comandos (entradas, saídas, processamentos, decisões, etc) e setas para indicar a sequência de execução.

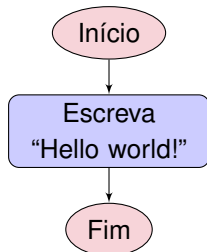
Fluxograma de um programa em C/C++



Estrutura básica de um programa em C/C++

- Exemplo: fluxuograma de um “Hello World”

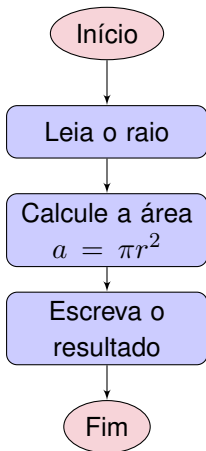
```
1  // Meu Primeiro Programa
2
3  #include <stdio.h>
4
5  int main()
6  {
7      // comentário explicativo
8      printf("Hello world!\n");
9      return 0;
10 }
```



Exemplo 1:

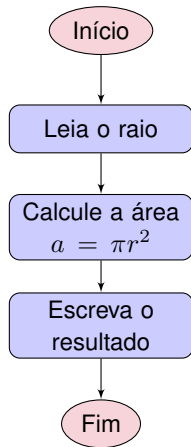
Faça um programa em C/C++, para calcular a área de um círculo. A área de um círculo é dada pela seguinte fórmula $a = \pi r^2$. O valor do raio r será digitado pelo usuário.

Fluxograma da solução



Solução do Exemplo 1:

```
1  /* Programa que calcula a área de um círculo
2  */
3
4  #include <stdio.h>
5
6  int main()
7  {
8      // declaração da constante Pi
9      const double PI = 3.141592;
10     double raio;
11
12     printf("Digite o raio do círculo: ");
13     scanf("%lf", &raio);
14
15     // calculando e imprimindo a área
16     double area = PI * raio * raio;
17     printf("\nÁrea do círculo: %lf\n", area);
18
19     return 0;
20 }
```



O Qualificador `const`

- A palavra-chave `const` assegura que a variável associada não será alterada em todo o programa.
- Esse qualificador é indicado para declarar valores constantes.
- Obrigatoriamente, as variáveis associadas ao qualificador `const` devem ser inicializadas.

Dúvida?

- Não existe área negativa.
- Portanto, o programa não pode calcular a área se o valor do raio for negativo.
- Como saber se o valor do raio digitado é positivo?
 - *Responderemos no final da aula...*

Aula de Hoje

- 1 Fluxogramas
- 2 Biblioteca `<math.h>`
- 3 Operadores relacionais
- 4 Operadores lógicos
- 5 Comandos de decisão
- 6 Exercícios
- 7 Próxima Aula

- Como calcular πr^2 ?

```
double area = PI * raio * raio;
```

- As linguagens C/C++ não possuem um operador para potência, mas possui uma biblioteca com diversas funções matemáticas, para usá-la devemos incluir a biblioteca `math.h`
- A função para potência é a `pow()`, sintaxe:

```
double pow(double base, double expoente);
```

- Exemplo:

```
#include <math.h>
...
double area = PI * pow(raio, 2);
...
```

Biblioteca Matemática – Parte I

Algumas funções matemáticas disponíveis na biblioteca `math.h`.

Para usá-las é necessário: `#include <math.h>`

Função	Descrição	Exemplo
<code>double ceil(x)</code>	arredonda x para cima	<code>ceil(9.1) → 10.0</code>
<code>double floor(x)</code>	arredonda x para baixo	<code>floor(9.8) → 9.0</code>
<code>double round(x)</code>	arredonda x	<code>round(9.5) → 10.0</code> <code>round(9.4) → 9.0</code>
<code>double trunc(x)</code>	retorna a parte inteira de x	<code>trunc(9.8) → 9.0</code>

Biblioteca Matemática – Parte I

Exemplo: Dada a tabela abaixo com os os valores de x , escreva os valores retornados pelas funções.

x	<code>round(x)</code>	<code>floor(x)</code>	<code>ceil(x)</code>	<code>trunc(x)</code>
2.3	2.0	2.0	3.0	2.0
3.8	4.0	3.0	4.0	3.0
5.5	6.0	5.0	6.0	5.0
-2.3	-2.0	-3.0	-2.0	-2.0
-3.8	-4.0	-4.0	-3.0	-3.0
-5.5	-6.0	-6.0	-5.0	-5.0

Biblioteca Matemática – Parte II

Funções para potências:

Função	Descrição	Exemplo
<code>double pow(x, y)</code>	x elevado a y: x^y	<code>pow(3, 2)</code> → 9.0
<code>double sqrt(x)</code>	raiz quadrada de x: $\sqrt{x}$	<code>sqrt(25)</code> → 5.0
<code>double cbrt(x)</code>	raiz cúbica de x: $\sqrt[3]{x}$	<code>cbrt(27)</code> → 3.0

Biblioteca Matemática – Parte III

Funções trigonométricas:

Função	Descrição	Exemplo
<code>double cos(x)*</code>	retorna o cosseno x	<code>cos(1.047) → 0.5</code>
<code>double sin(x)*</code>	retorna o seno x	<code>sin(1.571) → 1.0</code>
<code>double tan(x)*</code>	retorna a tangente x	<code>tan(0.785) → 1.0</code>
<code>double acos(x)**</code>	retorna o arco cosseno	<code>acos(0.5) → 1.047</code>
<code>double asin(x)**</code>	retorna o arco seno	<code>asin(1.0) → 1.571</code>
<code>double atan(x)**</code>	retorna o arco tangente	<code>atan(1.0) → 0.785</code>

*: valores em radianos

** : valores de x entre $[-1, 1]$

Biblioteca Matemática – Parte IV

Funções Exponencias e Logarítmicas:

Função	Descrição	Exemplo
<code>double exp(x)</code>	retorna exponencial de x: e^x	<code>exp(5)</code> → 148.4
<code>double log(x)</code>	logaritmo natural de x: $\ln(x)$	<code>log(5.5)</code> → 1.7
<code>double log10(x)</code>	logaritmo de x: $\log(x)$	<code>log10(1000)</code> → 3.0

Aula de Hoje

- 1 Fluxogramas
- 2 Biblioteca `<math.h>`
- 3 Operadores relacionais**
- 4 Operadores lógicos
- 5 Comandos de decisão
- 6 Exercícios
- 7 Próxima Aula

Tomada de decisões

- Permite a um programa realizar uma ação alternativa, a partir de um resultado **verdadeiro** ou **falso** produzido por uma condição.
- As **condições** são formadas utilizando-se os operadores de igualdade e os operadores relacionais.
- Ambos operadores de igualdade têm o mesmo nível de precedência, o qual é inferior ao dos operadores relacionais, e associam-se da esquerda para a direita.

Operadores de igualdade e relacionais

Operador algébrico de igualdade ou relacional padrão	Operador de igualdade ou relacional em C++	Exemplo de condição em C++	Significado da condição em C++
<i>Operadores relacionais</i>			
>	>	<code>x > y</code>	x é maior que y
<	<	<code>x < y</code>	x é menor que y
≥	>=	<code>x >= y</code>	x é maior que ou igual a y
≤	<=	<code>x <= y</code>	x é menor que ou igual a y
<i>Operadores de igualdade</i>			
=	==	<code>x == y</code>	x é igual a y
≠	!=	<code>x != y</code>	x não é igual a y

Erro Comum em Programação

- Confundir o operador de igualdade ==
- Com o operador de atribuição =

`total = x + 10;`  **avalia a expressão (direita)**
e atribui o resultado
à variável (esquerda)

`total == x + 10;` **verifica se a expressão da**
direita é IGUAL a expressão
da esquerda (vice-versa).

Aula de Hoje

- 1 Fluxogramas
- 2 Biblioteca `<math.h>`
- 3 Operadores relacionais
- 4 Operadores lógicos**
- 5 Comandos de decisão
- 6 Exercícios
- 7 Próxima Aula

Expressões booleanas

- No século 18, *George Boole*, matemático e filósofo britânico, desenvolveu um sistema algébrico lógico, que passou a ser conhecido como **Álgebra de Boole**.
 - Base para a lógica dos computadores digitais modernos.
 - Expressões lógicas (**expressões booleanas**) possuem o valor `true` ou `false`.
- Em C os inteiros também são usados como valores booleanos: qualquer valor não nulo (**1**) representa `true` e **0** representa `false`.

Expressões booleanas compostas

- na matemática, podemos restringir uma temperatura a um intervalo fechado, $0 \leq temp \leq 100$
- em C não podemos representar essa expressão por:
 $0 \leq temp \leq 100$
- embora ela seja uma expressão C válida!
- por exemplo, suponha que $temp = 150$; (não está no intervalo definido, logo esperamos que o resultado da expressão seja falso).

Expressões booleanas compostas

- os operadores relacionais são associativos a esquerda. A expressão será processada da seguinte forma:

$$\underbrace{0 \leq 150}_{(a)} \leq 100$$

- a expressão (a) resulta em **true**, que é representado em C pelo inteiro 1 (ou outro inteiro diferente de 0).
- Assim, na segunda etapa da avaliação, a expressão resulta em:

$$1 \leq 100$$

- que também é verdadeira e resulta em **true**.
- Entretanto, a expressão original deveria resultar falso, como na matemática.

Expressões booleanas compostas

- Para solucionar esse problema, reescrevemos a desigualdade como:

`(temp >= 0) && (temp <= 100)`

- onde `&&` é um operador lógico.
- Utilizamos os operadores lógicos para combinar expressões booleanas formando, assim, expressões booleanas compostas.

Operadores lógicos

Operador	Expressão	Nome	Descrição
!	!p	NÃO (negação)	!p é falso, se p é verd.; !p é verd., se p é falso.
&&	p && q	E (conjunção)	p && q é verdadeiro, se ambos, p e q são verd.; e falso, caso contrário.
	p q	OU (disjunção)	p q é verdadeiro, se p , q ou ambos é verd.; e falso, caso contrário.

Operadores lógicos – Tabela verdade

p	$!p$
true	false
false	true

p	q	$p \ \&\& \ q$	$p \ \ q$
true	true	true	true
true	false	false	true
false	true	false	true
false	false	false	false

Operadores lógicos – Precedência

Operator	Priority	Associativity
!, ~	highest	Right
/, *, %		Left
+, -		Left
<, >, <=, >=		Left
==, !=		Left
&		Left
^		Left
	lowest	Left
&&		Left
		Left
=, +=, *=, ...		Right

Aula de Hoje

- 1 Fluxogramas
- 2 Biblioteca `<math.h>`
- 3 Operadores relacionais
- 4 Operadores lógicos
- 5 Comandos de decisão**
- 6 Exercícios
- 7 Próxima Aula

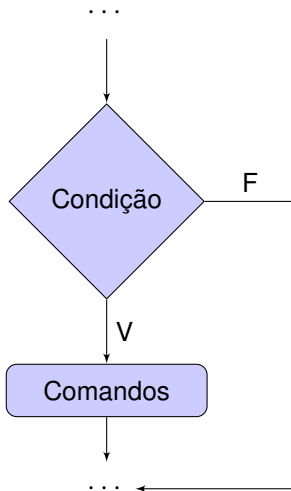
Tomada de decisão

- Permite a um programa realizar uma ação alternativa, a partir de um resultado **verdadeiro** ou **falso** produzido por uma condição.
- As condições são formadas utilizando-se os operadores de igualdade e os operadores relacionais.
- Ambos operadores de igualdade têm o mesmo nível de precedência, o qual é inferior ao dos operadores relacionais, e associam-se da esquerda para a direita.

Tomada de decisão

- Comando **if**
 - consiste de uma palavra-chave **if** seguida de uma expressão de teste entre parênteses. A instrução será executada apenas se a expressão de teste for **verdadeira**.
 - O corpo de um comando **if** pode conter **uma única instrução terminada por ponto-e-vírgula** ou **várias instruções entre chaves**.

Tomada de decisão



Sintaxe do comando `if`

```
1  if ( <expressão_de_teste> )  
2      instrução_única;
```

OU

```
1  if ( <expressão_de_teste> )  
2  {  
3      instrução1;  
4      instrução2;  
5      instrução3;  
6      ...  
7  }
```

Exemplo usando o comando `if`

Exemplo 1 (resolvido anteriormente):

Faça um programa em C/C++, para calcular a área de um círculo. A área de um círculo é dada pela seguinte fórmula $a = \pi r^2$. O valor do raio r será digitado pelo usuário.

- Altere o programa anterior para calcular a área somente se o valor do raio for positivo.

```
1  /* Programa que calcula a área de um círculo
2  */
3
4  #include <stdio.h>
5
6  int main()
7  {
8      // declaração da constante Pi
9      const double PI = 3.141592;
10     double raio;
11
12     printf("Digite o raio do círculo: ");
13     scanf("%lf", &raio);
14
15     // calculando e imprimindo a área do círculo
16     if (raio >= 0) {
17         double area = PI * raio * raio;
18         printf("\nÁrea do círculo: %lf\n", area);
19     }
20
21     return 0;
22 }
```

Aula de Hoje

- 1 Fluxogramas
- 2 Biblioteca `<math.h>`
- 3 Operadores relacionais
- 4 Operadores lógicos
- 5 Comandos de decisão
- 6 Exercícios**
- 7 Próxima Aula

Exercícios

Atenção: **entregar o exercício na próxima aula.**

Exercício 1: Maior número

Faça um programa que leia dois números inteiros e verifique qual deles é maior.

- Imprima uma mensagem informando qual deles é o maior.
- Exemplo de saída caso o usuário digite 10 e 20:

```
1 20 é o maior número
```

Aula de Hoje

- 1 Fluxogramas
- 2 Biblioteca `<math.h>`
- 3 Operadores relacionais
- 4 Operadores lógicos
- 5 Comandos de decisão
- 6 Exercícios
- 7 Próxima Aula**

Próxima Aula

- Resolução (e discussão) de exercícios pelos alunos (sorteio!!!)
- Comando de decisão
 - `if - else`



Perguntas?