BCC201 – Introdução à Computação

Turmas 61, 62, 63, 64, 65 e 66

Universidade Federal
de Ouro Preto

Puca Huachi Vaz Penna

Departamento de Computação
Universidade Federal de Ouro Preto

http://www.decom.ufop.br/puca
puca@iceb.ufop.br

Aula 4
Estruturas Condicionais
2017/1

Aula Anterior

- Introdução ao C++
  - Declaração de variáveis
  - Operadores Aritméticos
  - Operadores Relacionais
  - Operadores Lógicos

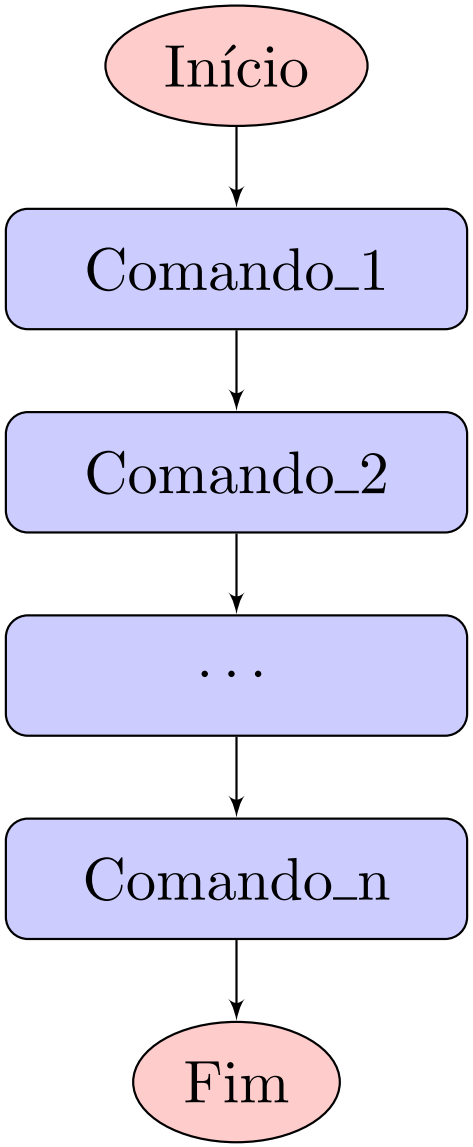
Estrutura básica de um programa em C++

```
1  <Inclusão_de_bibliotecas>
2
3  tipo main(<declaração_dos_parâmetros>)
4
5  {
6       instrução_1;
7       instrução_2;
8       instrução_3;
9          ...
10      instrução_n;
11
12      return 0;
13 }
```

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.

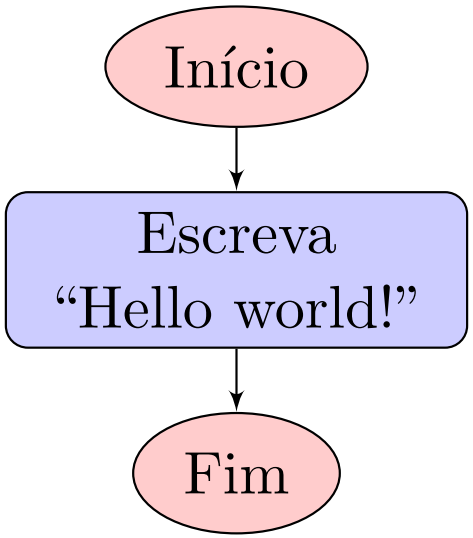
Fluxograma de um programa em C++



Estrutura básica de um programa em C++

```
1  // Meu Primeiro Programa
2  // Autor: Puca Huachi
3
4  #include <iostream>
5
6  using namespace std;
7
8  int main()
9  {
10     // comentário explicativo
11     cout << "Hello world!";
12
13     return 0;
14 }
```

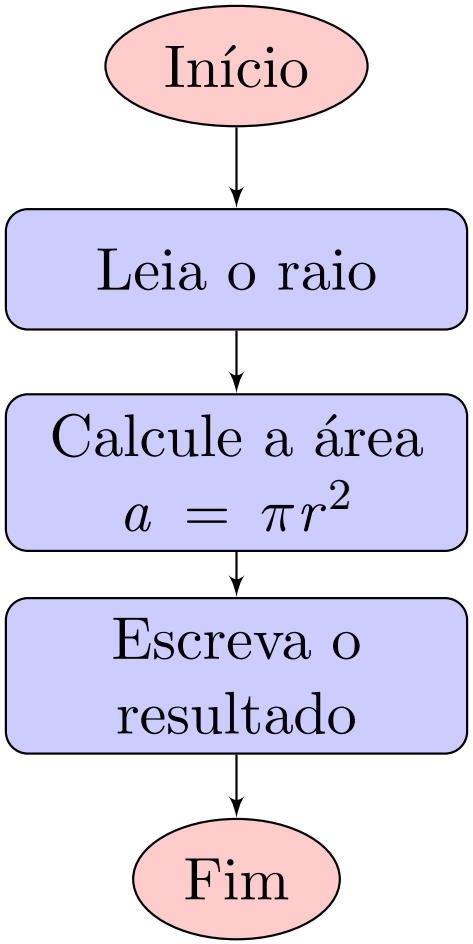
Fluxograma um programa em C++



Exemplo 1:

Faça um programa em 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  // Calcula a área de um círculo
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7      // declaração da constante Pi
8      const double Pi = 3.141592;
9      double raio;
10
11     cout << "DIGITE O VALOR DO RAIO DO CIRCULO: ";
12     cin >> raio;
13
14     double area = Pi * raio * raio;
15     cout << "\nAREA DO CIRCULO: " << area << endl;
16
17     return 0;
18 }
```

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?

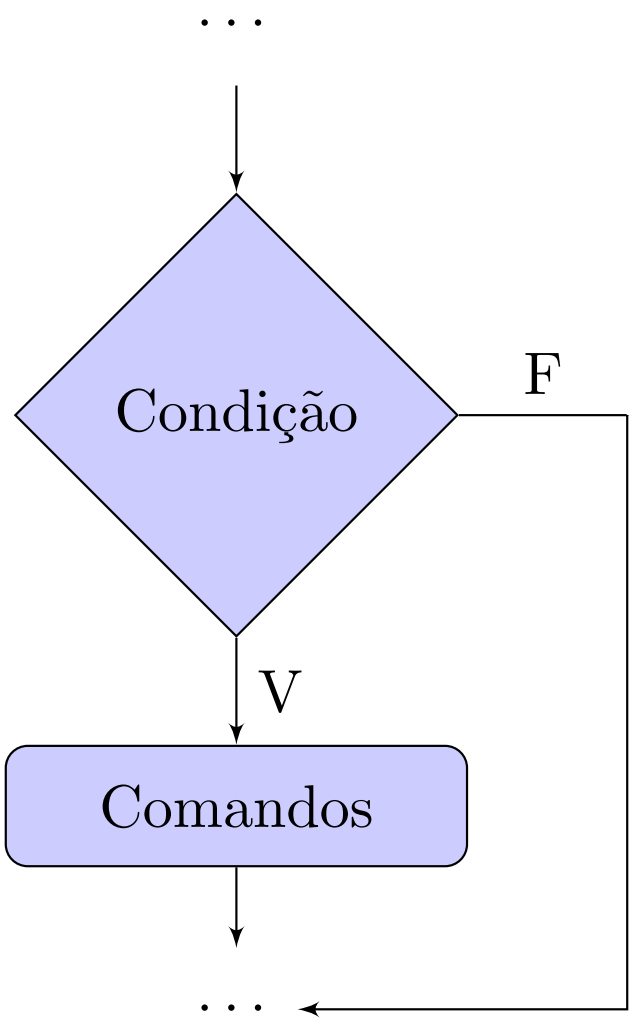
Tomada de decisão

- 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. Se a expressão de teste for verdadeira, a instrução será executada; do contrário, nada será feito.
  - 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`

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

```
1  // Calcula a área de um círculo
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7      const double Pi = 3.141592; // Constante Pi
8      double raio;
9
10     cout << "DIGITE O VALOR DO RAIO DO CIRCULO: ";
11     cin >> raio;
12
13      if (raio >= 0) // Testa se o raio é positivo
14      {
15          double area = Pi * raio * raio;
16          cout << "\nAREA DO CIRCULO: " << area;
17          cout << endl;
18      }
19      return 0;
20 }
```

Exercício 2

- Codifique um programa que leia um número inteiro positivo. A seguir o programa imprime uma mensagem para o usuário dizendo se o número digitado é par. Se o número não for par, o programa não deve fazer nada.

```cpp
// Verifica se um número é par
#include <iostream>
using namespace std;

int main()
{
    int numero; //variável para armazenar o número

    cout << "Digite um numero inteiro: ";
    cin >> numero;

     // Testa se o número é par
     if ((numero % 2) == 0)
     {
         cout << "\nO número " << numero
              << " é par." << endl;
     }
     return 0;
}
```

Exemplo 3

Faça um programa em 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.

Verifique se o raio é positivo antes de efetuar cálculo, caso contrário imprima uma mensagem de erro ao usuário.

```cpp
// Calcula a área de um círculo
#include <iostream>
using namespace std;
int main()
{
  const double Pi = 3.141592; // Constante Pi
  double raio;

  cout << "DIGITE O VALOR DO RAIO DO CIRCULO: ";
  cin >> raio;
  if (raio >= 0) // Testa se o raio é positivo
  {
    double area = Pi * raio * raio;
    cout << "\nAREA DO CIRCULO: " << area << endl;
  }
  if (raio < 0)
    cout << "O raio deve ser positivo!" << endl;
  return 0;
}
```

Exercício 4

- Codifique um programa que leia um número inteiro positivo. A seguir o programa imprime uma mensagem para o usuário dizendo se o número digitado é **par ou impar**.

```
1  // Programa que verifica se um no. é par ou impar
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7     int n; //variável para armazenar o número
8     cout << "Digite um numero inteiro: ";  cin >> n;
9     // Testa se o número é par
10    if ((numero % 2) == 0)
11    {
12       cout << "\nO número " << n << " é par.\n";
13    }
14    //Testa se o número é impar
15    if ((numero % 2) != 0)
16    {
17       cout << "\nO número " << n << " é impar\n.";
18    }
19    return 0;
20 }
```

Exemplo de execução

- Execução 1:

```
Digite um numero inteiro:  5
O número 5 é ímpar
```

- Execução 2:

```
Digite um numero inteiro:  8
O número 8 é par
```

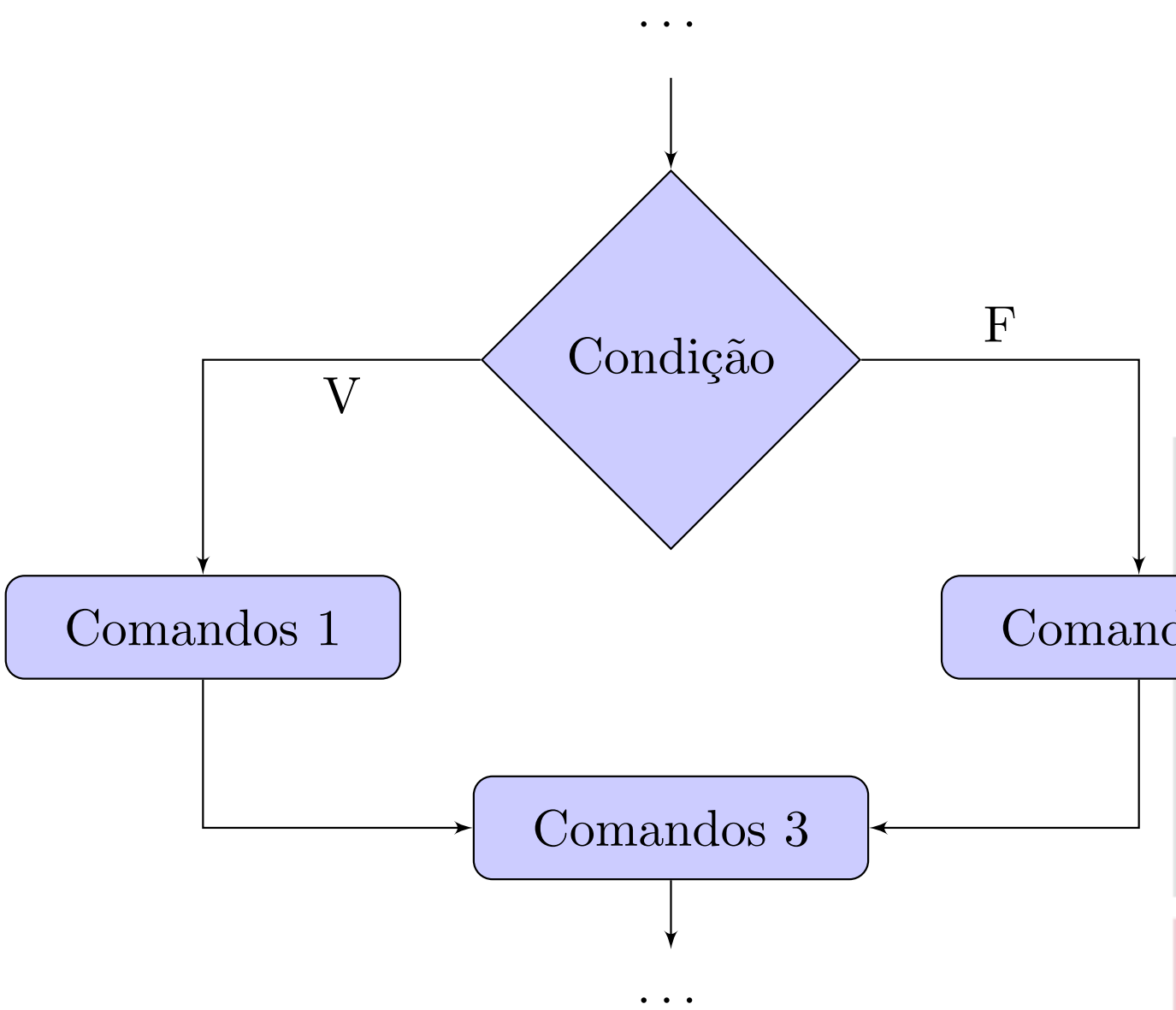
O Comando `if`

- No Exercício 3 e 4, o programa pode tomar duas decisões distintas.
- No entanto, são necessários dois comandos `if`.
- Podemos simplificar o programa com o uso do comando `if-else`.

O Comando `if-else`

- O comando `if` permite executarmos um ou mais comandos, se a expressão relacional resultar em verdadeiro. Se desejarmos que algo seja executado se a expressão relacional resultar em **f**also, então devemos utilizar o comando `if-else`.
- Dessa forma, podemos executar um ou mais comandos se o teste for verdadeiro; ou executar um ou mais comandos distintos se o teste for falso.
- O comando `if-else` consiste no comando `if` seguido de uma instrução (ou um bloco de instruções), seguido da palavra-chave `else`, seguido de uma instrução (ou um bloco de instruções).

Tomada de decisão



Sintaxe do comando `if-else`

```
1  if ( <expressão_de_teste> )
2      instrução_única_V;
3  else
4      instrução_única_F;
```

ou

```
1  if ( <expressão_de_teste> )
2  {
3      instrução_V1;
4          ...
5      instrução_Vn;
6  }
7  else
8  {
9      instrução_F1;
10         ...
11     instrução_Fn;
12 }
```

Exercício 5

- Codifique um programa que leia um número inteiro positivo. A seguir o programa imprime uma mensagem para o usuário dizendo se o número digitado é **par ou impar**.

```cpp
// Verifica se um número é par
#include <iostream>
using namespace std;

int main()
{
    int numero; //variável para armazenar o número

    cout << "Digite um numero inteiro: ";
    cin >> numero;

     // Testa se o número é par
     if ((numero % 2) == 0)
         cout << "\nO número " << numero
              << " é par." << endl;
     else
         cout << "\nO número " << numero
              << " é ímpar." << endl;
     return 0;
}
```

Exercício 6

- Faça um programa que leia dois números inteiros e verifique qual deles é maior.
- Imprima uma mensagem informando qual deles é o maior.

Exercício 7

- Uma loja deseja mandar uma correspondência a um dos seus clientes anunciando um bônus especial. Escreva um algoritmo que leia o valor das compras desse cliente no ano passado e calcule um bônus de 10%, se o valor das compras for menor que R$ 50.000,00, e de 15%, caso contrário. O algoritmo deve imprimir o valor do bônus cedido ao cliente.

Exercício 8

- Escreva um programa para encontrar as raízes de uma equação do segundo grau:

$$ax^2 + bx + c = 0$$

- Os coeficientes da equação são reais. O programa faz a alocação de 3 posições de memória para esses coeficientes, inicializando-os com o valor zero.
- O programa efetua a leitura dos coeficientes através do teclado, e a seguir, calcula o valor das raízes existentes. Caso não existam raízes reais, o programa deve informar este fato ao usuário.

```
1  // Calcula as raízes de uma equação do 2o grau
2  #include <iostream>
3  #include <cmath>
4  using namespace std;
5  int main()
6  {
7     double a = 0, b = 0, c =0;
8     double delta, x1, x2;
9
10    cout << "\nCoeficiente a: "; cin >> a;
11    cout << "\nCoeficiente b: "; cin >> b;
12    cout << "\nCoeficiente c: "; cin >> c;
13    delta = (b * b) - 4 * a * c;
14
15    if ( delta < 0)
16       cout << "\nNão existem raízes reais " << endl;
17    else {
18       x1 = (-b + sqrt(delta)) / (2 * a);
19       x2 = (-b - sqrt(delta)) / (2 * a);
20       cout << "\nX1 =  " << x1 << endl;
21       cout << "X2 = " << x2 << endl;
22    }
23     return 0;
24 }
```

Exercício 9

- Como melhorar o programa anterior para tratar as seguintes situações:
  - Não existem raízes reais ($\Delta < 0$);
  - Existem raízes reais idênticas ($\Delta = 0$);
  - Existem raízes reais distintas ($\Delta > 0$);

```
1  // Calcula as raízes de uma equação do 2o grau
2  #include <iostream>
3  #include <cmath>
4  using namespace std;
5  int main()
6  {
7     double a = 0, b = 0, c =0;
8     double delta, x1, x2;
9     cout << "\nCoeficiente a: "; cin >> a;
10    cout << "\nCoeficiente b: "; cin >> b;
11    cout << "\nCoeficiente c: "; cin >> c;
12    delta = (b * b) - 4 * a * c;
13    if ( delta < 0 )
14       cout << "\nNão existem raízes reais " << endl;
15    else
16       if ( delta == 0 ) {
17          x1 = -b / (2 * a);
18          cout << "\nX1 =  X2 = " << x1 << endl;
19       } else {
20          x1 = (-b + sqrt(delta)) / (2 * a);
21          x2 = (-b - sqrt(delta)) / (2 * a);
22          cout << "\nX1 =  " << x1 << endl;
23          cout << "X2 = " << x2 << endl;
24       }
25     return 0;
26 }
```

Exercícios propostos:

- Escreva um programa que leia de 3 números inteiros. O programa deve verificar qual dos valores é o menor, o intermediário e o maior.
- Supondo as entradas 25, 12, 22, a saída seria: imprime:

```
MENOR VALOR: 12
VALOR INTERMEDIARIO: 22
MAIOR VALOR: 25
```

Exercícios propostos:

- Codifique um programa que faça leitura de 3 valores inteiros.
- Estes valores representam os lados de um triângulo.
- O programa verifica a condição de existência do triângulo. Caso exista o triângulo, o mesmo é classificado em como equilátero, isósceles ou um triângulo qualquer. Se o triângulo não existir, uma mensagem é impressa para o usuário.

Exercícios propostos:

- Codifique um programa que faça a de dois valores inteiros. Esses valores devem ser fornecidos ao programa através da leitura pelo teclado. A seguir o programa lê um caractere, que deve ser '+' ou '-' ou '*' ou '/', e realiza a operação indicada pelo caractere sobre os valores reais lidos.
- O programa deve imprimir os valores e o resultado da operação realizada sobre eles, como mostra o exemplo a seguir:

```
PRIMEIRO VALOR: 4
SEGUNDO VALOR: 5
Operação:  +
4 + 5 = 9
```

Próxima Aula

- Comando condicionais
  - Sim, isso mesmo! Mais comandos de decisão.