ALGORITMOS

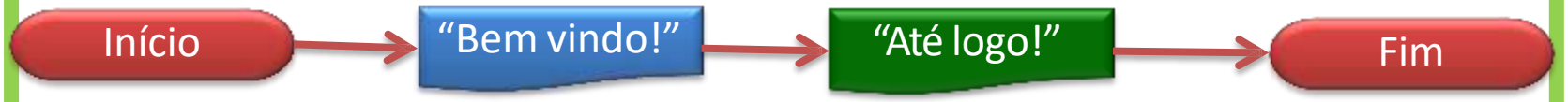
ROTEIRO DA AULA

- Conceitos
- Testes condicionais
- Operadores relacionais
- Testes condicionais aninhados
- Condições compostas
- Seleção múltipla

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA SUL-RIO-GRANDENSE Campus Passo Fundo

PROGRAMAS SEQUÊNCIAS

- Até agora, nossos programas eram puramente sequenciais:



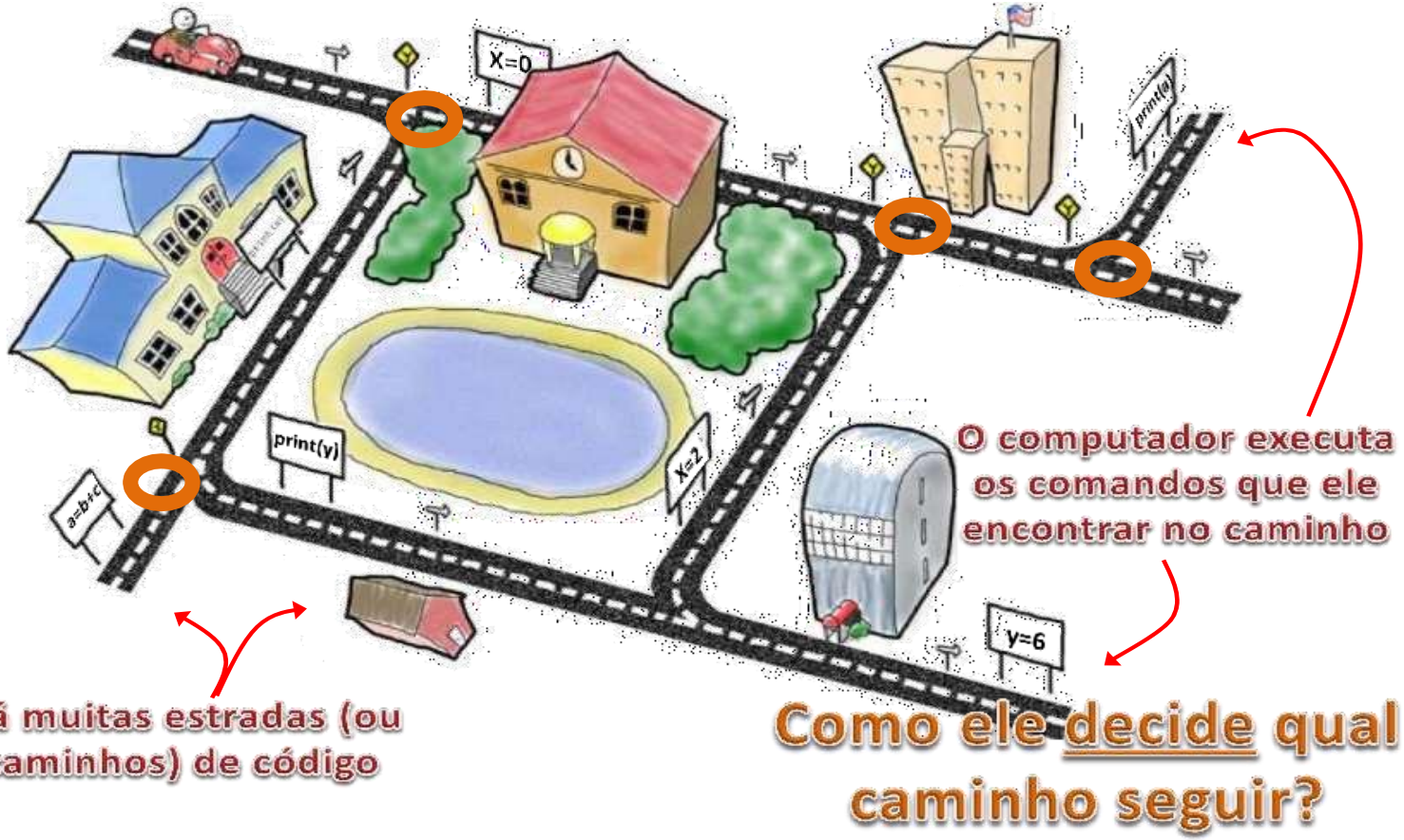

```
# Um exemplo
escreva("Bem vindo!")
escreva("Até logo!")
```

– São como estradas sem **desvios**!

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE
Campus Passo Fundo

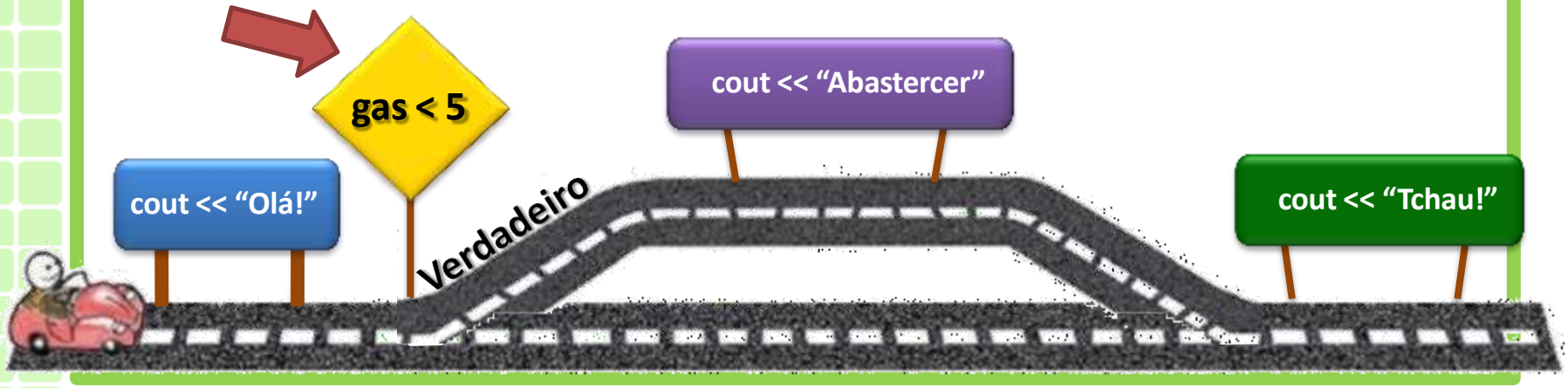
NECESSIDADE DE ESCOLHAS

- O mundo real é uma estrada sem desvios?



CONCEITOS

- Como o computador escolhe um caminho?
  - Com um desvio condicional
- Condicional?
  - Decisão com base em uma condição
    - Proposição verdadeira ou falsa



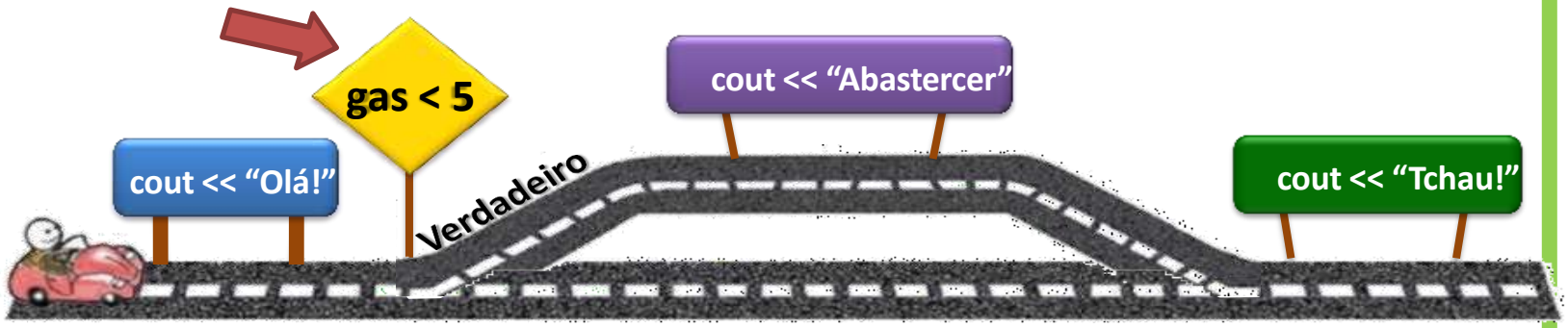
CONCEITOS

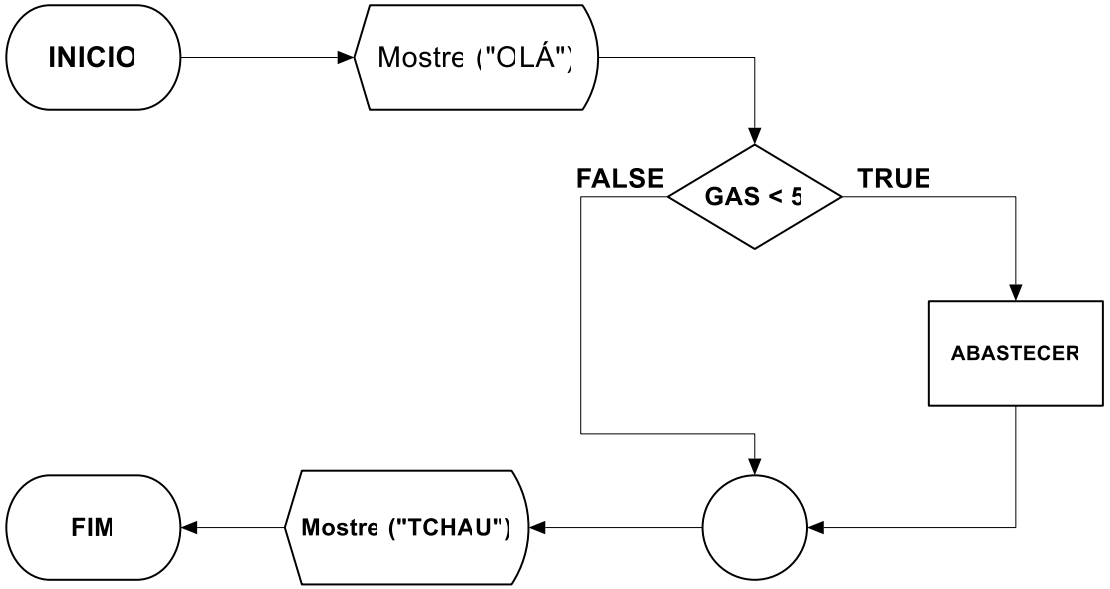
- Frequentemente na construção de algoritmos, vamos nos defrontar com problemas, onde é necessário uma entre duas ou mais situações possíveis.
- As estruturas de seleção ou testes condicionais são utilizadas para fazer comparações de tal forma que possamos simular uma decisão e escolher apenas um dos caminhos possíveis para seguir.

CONCEITOS

- Há a subordinação da execução de um ou mais comandos à veracidade de uma condição.
- É uma expressão lógica que, quando inspecionada terá como resposta TRUE (Verdadeiro) ou FALSE (Falso)
- Pode executar várias ações no caminho escolhido
- Sempre seguirá por um único caminho, ou seja, se a decisão for verdadeira seguirá para as instruções verdadeiras, se a decisão for falsa seguirá pelas instruções do caminho falso.

REPRESENTAÇÃO NO DIAGRAMA





INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE
Campus Passo Fundo

OPERADORES RELACIONAIS

- Operadores Relacionais: comparadores

Operador Relacional	C++	Exemplo	Significado
=	==	x == 2	X é igual a 2?
≠	!=	x != 2	X é diferente de 2?
>	>	x > 2	X é maior que 2?
≥	>=	x >= 2	X é maior ou igual a 2?
<	<	x < 2	X é menor que 2?
≤	<=	x <= 2	X é menor ou igual a 2?

TIPOS DE CONDICIONAL

- Seleção simples: Quando necessitamos testar uma única condição
- O comando condicional "**se**" executa um bloco de instruções caso uma expressão lógica seja **VERDADEIRA**.

```
se <condição>
    instruções
fim se
```

CONDIÇÃO SIMPLES

- Usado para fazer comparação de tal forma que possamos simular uma decisão:

```
Se (condição)
{
        Comando;
}
```

```
Se (condição)

        Comando;
```

- Vários comandos:

```
Se (condição)
{
        Comando;
        Comando;
        Comando;
}
```

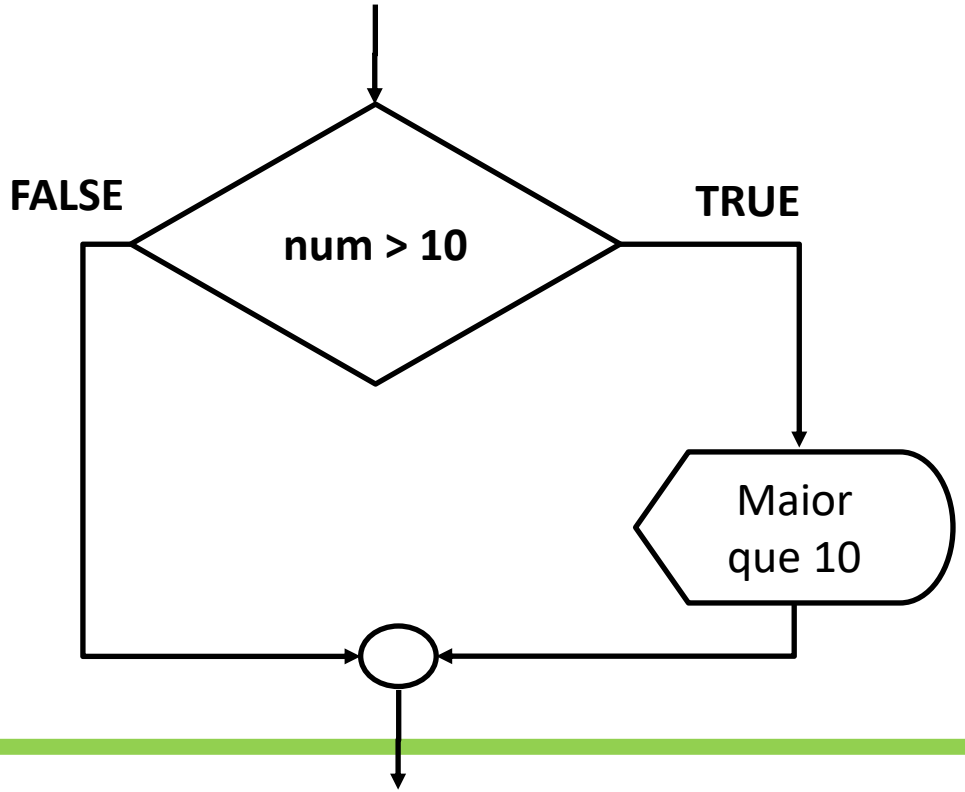
CONDIÇÃO SIMPLES

- Se a condição for verdadeira a ação sob a cláusula então será executada, caso contrário (for falsa) encerra-se a seleção (fimse).
- Português estruturado:

```
8  inicio
9      num <- 15
10     se (num > 10) entao
11         escreva ("Número maior que 10.")
12     fimse
13 fimalgoritmo
```

CONDIÇÃO SIMPLES

- Diagrama de blocos com seleção simples e ações somente em caso de verdadeiro



INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE
Campus Passo Fundo

CONDIÇÃO SIMPLES

- Programa na linguagem C++ com seleção simples e ações somente em caso de verdadeiro:

```
main()
{
    int n = 15;
    if(n > 10)
    {
        cout << "Número maior que 10";
    }
}
```

- As chaves são opcionais quando existe somente uma instrução a ser executada

```
main()
{
    int n = 15;
    if(n > 10)
        cout << "Número maior que 10";
}
```

CONDIÇÃO SIMPLES COM SENÃO

- Caso exista uma ação a ser executada em caso de resultado falso, pode-se acrescentar a cláusula “senao” e as respectivas instruções.

```
se <condição>
    instruções
senão
    instruções
fim se
```

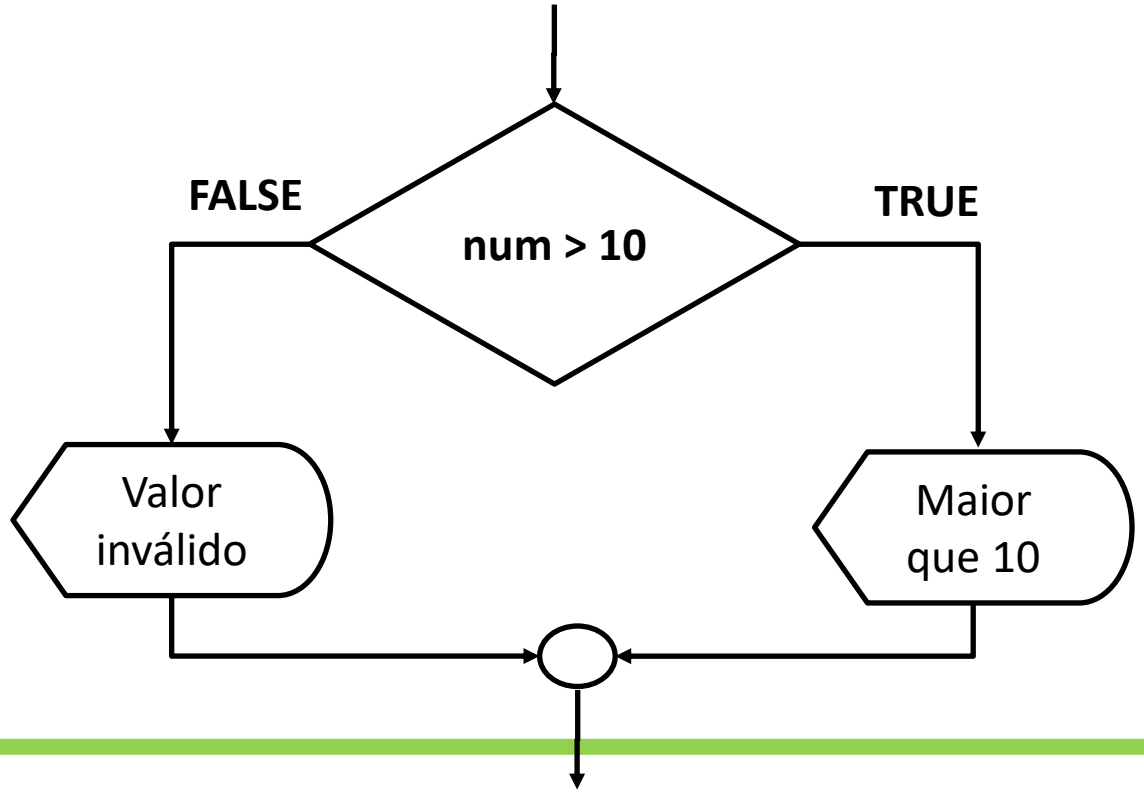
CONDIÇÃO SIMPLES COM SENÃO

- Exemplo: ler um número qualquer e caso o número seja maior que 10 exibir esta informação, caso contrário exibir número inválido.
- Português estruturado:

```
8  inicio
9  // Seção de Comandos
10     escreva ("Informe um número qualquer: ")
11     leia (num)
12     se (num > 10) entao
13        escreva ("Número maior que 10")
14     senao
15        escreva ("Número inválido.")
16     fimse
17 fimalgoritmo
```

CONDIÇÃO SIMPLES COM SENÃO

- Diagrama de blocos com seleção simples e ações se não for verdadeiro



CONDIÇÃO SIMPLES COM SENÃO

- Linguagem C++:

```
main()
{
    int n = 15;
    if(n > 10)
        cout << "Número maior que 10";
    else
        cout << "Valor inválido";
}
```

TESTE CONDICIONAL ENCADEADO

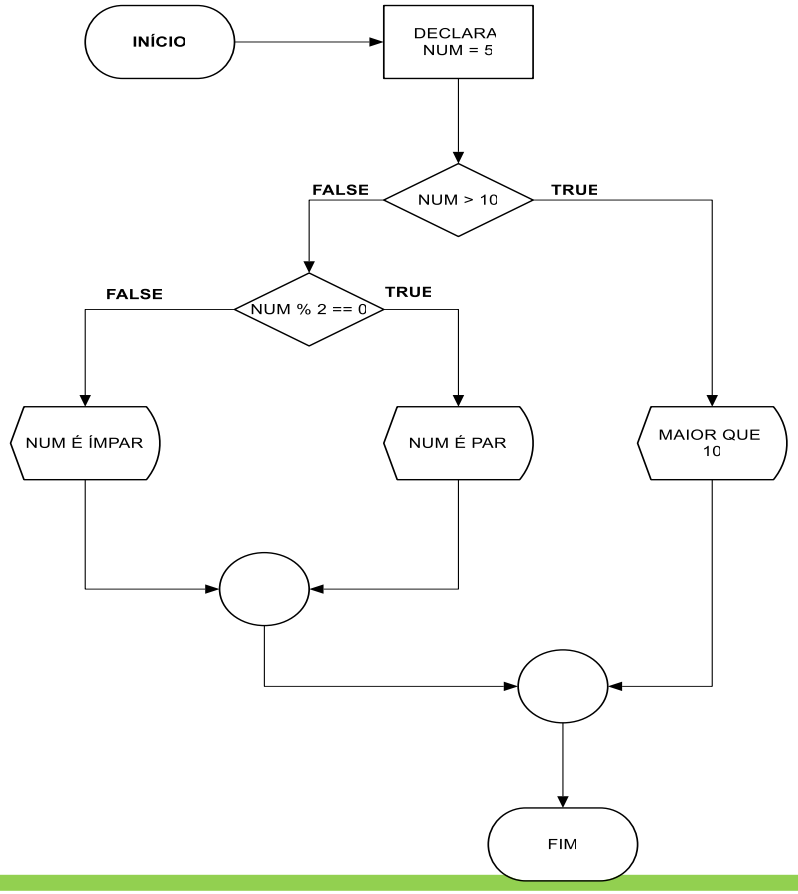
- Seleção encadeada ou aninhada consiste em caso verdadeiro ou falso iniciar um novo teste “se”.

```
8  inicio
9      num <- 5
10     se (num > 10) entao
11         escreva ("Número maior que 10.")
12     senao
13         se (num mod 2 = 0) entao
14             escreva ("É par")
15         senao
16             escreva ("É ímpar")
17         fimse
18     fimse
19 fimalgoritmo
```

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE
Campus Passo Fundo

TESTE CONDICIONAL ENCADEADO

- Diagrama de Blocos com seleção encadeada ou aninhada:



TESTE CONDICIONAL ENCADEADO

- Programa em C++ com seleção encadeada ou aninhada:

```
main()
{
    int n = 6;
    if(n > 10)
        cout << "Número maior que 10";
    else if(n % 2 == 0)
        cout << "Número é par";
    else
        cout << "Número é ímpar";
}
```

CONDIÇÃO COMPOSTA

- Condição composta: ocorre quando existe mais de uma condição a ser avaliada no mesmo teste.
- Obrigatoriamente devem ser analisadas com um operador lógico entre as condições.

Operador Lógico	**C++**
E	&&, and
OU	||, or
OU EXCLUSIVO	^, xor
NÃO	!

CONDIÇÃO COMPOSTA

- Tabela Verdade: é uma ferramenta de natureza matemática muito utilizada no campo do raciocínio lógico.
- Seu objetivo é verificar a validade lógica de uma proposição composta.

A	B	A e B	A ou B	A xor B	!A
V	V	V	V	F	F
V	F	F	V	V	F
F	V	F	V	V	V
F	F	F	F	F	V

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE
Campus Passo Fundo

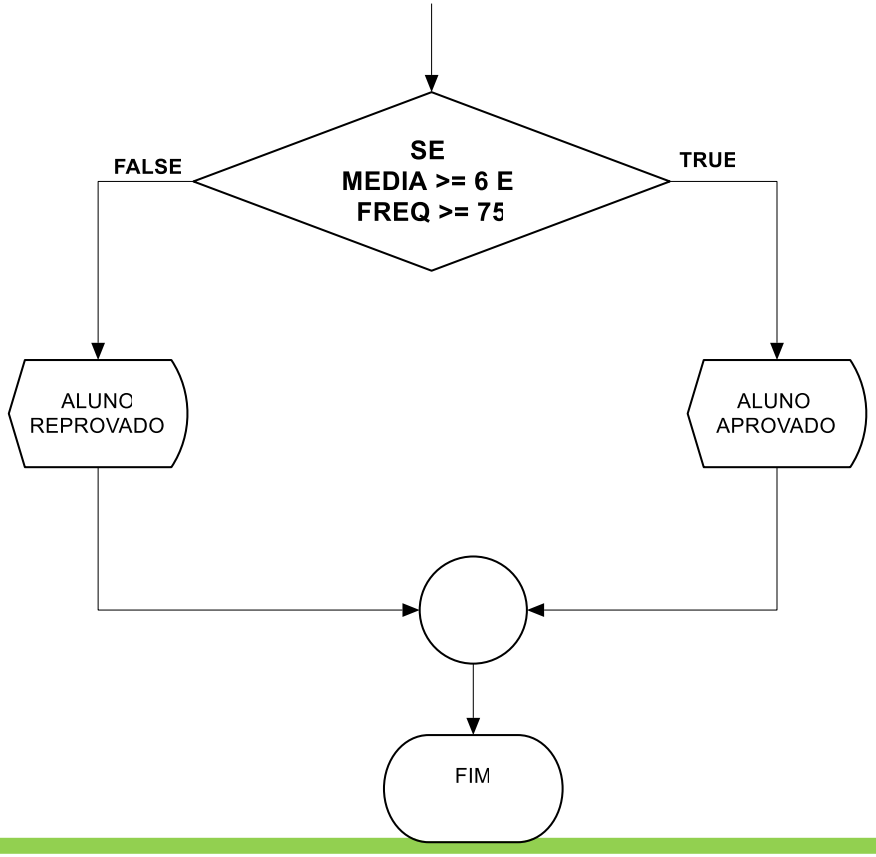
CONDIÇÃO COMPOSTA

- Exemplo em pseudocódigo:

```
9  inicio
10     media <- 7
11     freq <- 80
12     se (media >= 6) e (freq >= 75) entao
13        escreva ("Aluno aprovado.")
14     senao
15        escreva ("Aluno reprovado")
16     fimse
17 fimalgoritmo
```

CONDIÇÃO COMPOSTA

- Exemplo no diagrama de blocos:



CONDIÇÃO COMPOSTA

- Exemplo na linguagem C++:

```
main()
{
    int frequencia = 95;
    float media = 7.5;

    if ((media >= 6) && (frequencia >= 75))
        cout << "Aluno aprovado";
    else
        cout << "Aluno reprovado";
}
```

- Exibe: Aluno aprovado

CONDIÇÃO COMPOSTA

- Exemplo na linguagem C++:

```
main()
{
    int frequencia = 54;
    float media = 8;

    if ((media >= 6) && (frequencia >= 75))
        cout << "Aluno aprovado";
    else
        cout << "Aluno reprovado";
}
```

- Exibe: Aluno reprovado

CONDIÇÃO COMPOSTA

- Exemplo na linguagem C++:

```
main()
{
    int frequencia = 54;
    float media = 8;

    if ((media >= 6) || (frequencia >= 75))
       cout << "Aluno aprovado";
    else
       cout << "Aluno reprovado";
}
```

- Exibe: Aluno aprovado

SELEÇÃO MÚLTIPLA

- Em algumas situações é apresentado várias condições possíveis, neste caso podemos escrever nossos algoritmos usando a instrução escolha, que possui a sintaxe apresentada:

SELEÇÃO MÚLTIPLA

```
escolha <expressão-de-teste>
  caso <exp11>
       <sequência-de-comandos-1>
  caso <exp21>
       <sequência-de-comandos-2>
  outrocaso
       <sequência-de-comandos-extra>
fimescolha
```

SELEÇÃO MÚLTIPLA

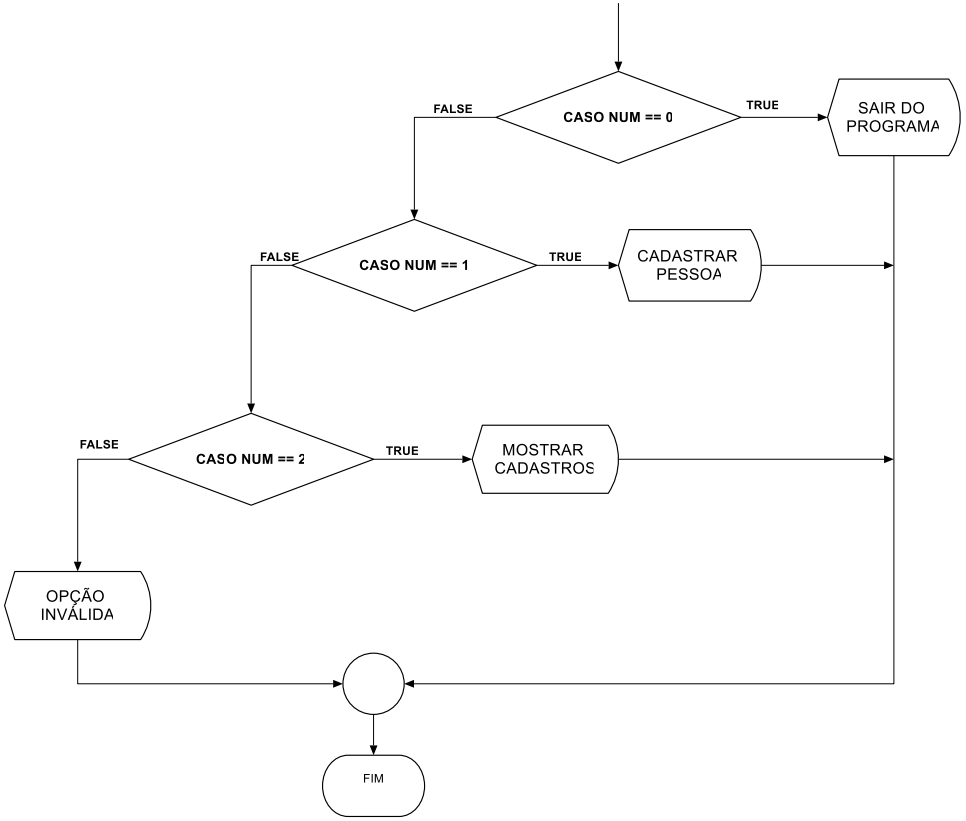
- **Exemplo pseudocódigo:**

```
Algoritmo “teste”
Var
   num : inteiro
Inicio
   escreva (“Digite sua opção”)
   leia(num)
   escolha (num)
      caso 0
            escreva(“Programa finalizado”)
      caso 1
            escreva(“Cadastrar pessoa”)
      caso 2
            escreva(“Mostrar cadastros”)
      outrocaso
            escreva (“Opção inválida”)
    fimescoelha
Fimalgoritmo
```

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE
Campus Passo Fundo

SELEÇÃO MÚLTIPLA

- Exemplo no diagrama de blocos:



SELEÇÃO MÚLTIPLA

- Exemplo em C++:

```
switch (num)
{
  case 0 :
    cout << "Sair do progrma";
    break;
  case 1 :
    cout << "Cadastrar pessoa";
    break;
  case 2 :
    cout << "Mostrar cadastros existentes";
    break;
  default :
    cout << "Opção inválida";
    break;
};
```

SELEÇÃO MÚLTIPLA

- Testando um valor dentro de um intervalo:

```
switch(codigo)
{
   case 10 ... 20:
      cout << "10...20" << endl;
      break;
   case 21 ... 30:
      cout << "21...30" << endl;
      break;
   default:
      cout << "Fora do intervalo";
      break;
}
```

SELEÇÃO MÚLTIPLA

- Vários case com uma única instrução

```
switch(mes)
{
case 1:
case 3:
case 5:
case 7:
case 8:
case 10:
case 12:
    cout << "O mês informado possui 31 dias" << endl;
    break;
case 4:
case 6:
case 9:
case 11:
    cout << "O mês informado possui 30 dias" << endl;
    break;
case 2:
    cout << "O mês informado possui 28 ou 29 dias" << endl;
    break;
default:
    cout << "O mês informado é inválido!" << endl;
    break;
}
```

SELEÇÃO MÚLTIPLA

- O comando switch “escolha” só testa igualdades
- A opção default “outrocaso” só será executada quando não entrar em nenhum comando “caso” anterior.
- O comando escolha é considerado mais eficiente que vários se/entao se.

REFERÊNCIAS

- ASCENCIO, Ana Fernanda Gomes. CAMPOS, Edilene Aparecida Veneruchi de. *Fundamentos da programação de computadores*. 2 ed. São Paulo: Pearson, 2007.
- FORBELLONE, André Luiz Villar. Lógica de programação: a construção de algoritmos e estruturas de dados. 3 ed. São Paulo: Prentice Hall, 2005.
- MORAES, Paulo Sérgio. Curso Básico de Lógica de Programação. Centro de Computação – Unicamp, 2000.
- STEINMETZ, Ernesto H. R.; FONTES, Roberto Duarte Cartilha Lógica de Programação. Editora IFB, Brasília - DF, 2013.