

Programação de Computadores

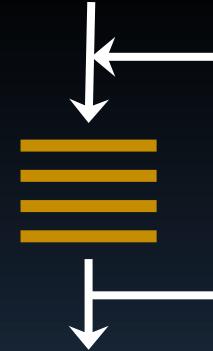
# INSTRUÇÃO DE DESVIO IF

# Introdução

- Um programa é uma **sequência** de instruções
  - Elas podem se repetir
  - Elas podem ser desviadas (saltadas)



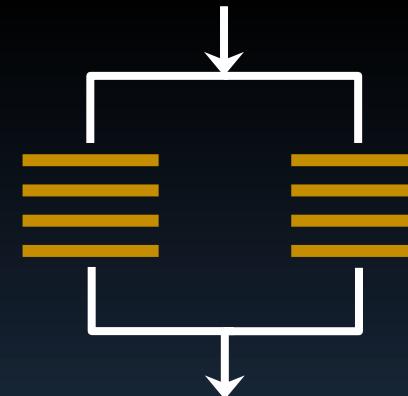
Execução  
Sequencial



Repetição  
de Instruções



Desvio  
Simples



Desvio  
Múltiplo

# Introdução

- Computadores fazem mais que:
  - Armazenar dados (tipos básicos e compostos)
  - Repetir instruções (laços for, while e do-while)
- Os desvios permitem que seja feita a tomada de decisão
  - É a base para o comportamento inteligente
  - C++ possui duas instruções de desvio:
    - **if** e **switch**

# A Instrução if

- Utiliza-se o **if** para decidir sobre uma determinada ação
  - Se a nota for maior que **7** o aluno está aprovado
  - Se o sensor detectar movimento dispare o alarme
  - Se os objetos colidirem emita o som de explosão
  - Se o nível do rio exceder o limite abra a comporta
- A instrução if possui **duas formas:**
  - **if**
  - **if else**

# A Instrução if

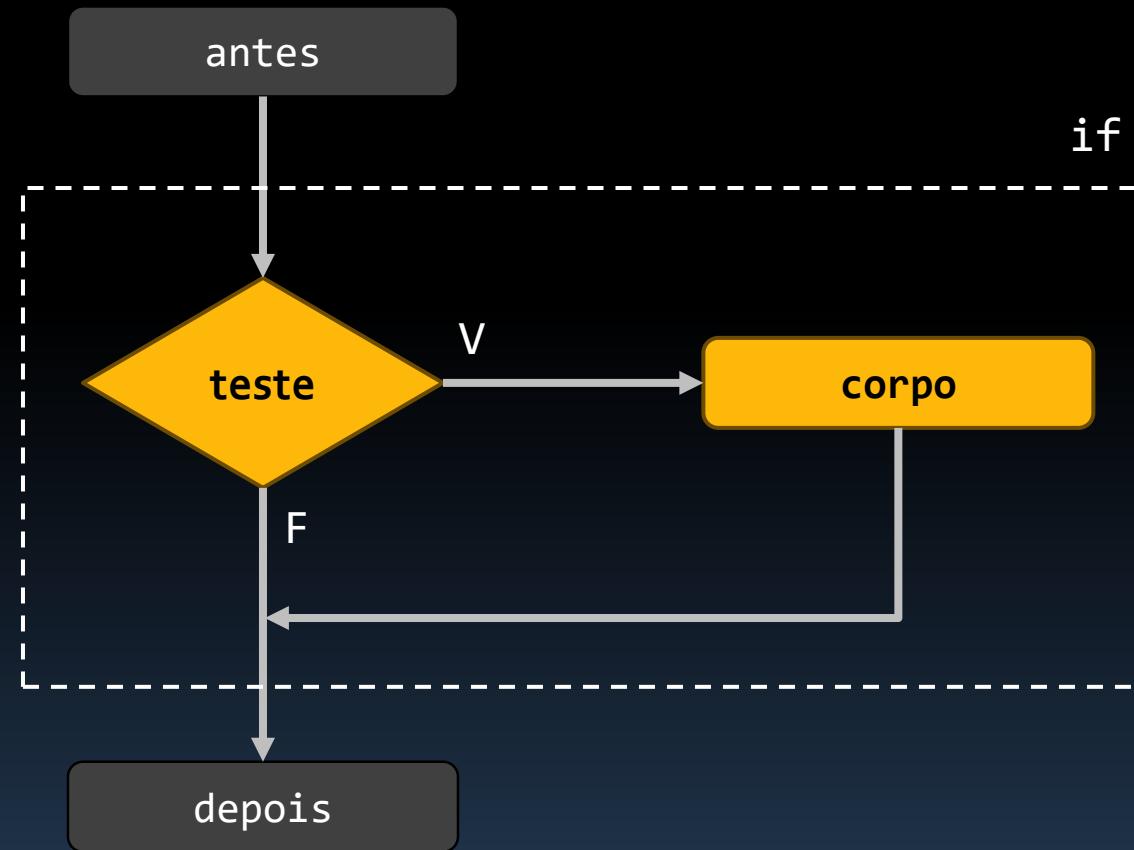
- A instrução if executa uma instrução (ou bloco de instruções) se o teste for satisfeito

Verifica se a expressão de teste é verdadeira

```
if (teste)  
    corpo;
```

Instrução (ou bloco de instruções) a ser executada

# A Instrução if



if

```
antes;  
if (teste)  
    corpo;  
depois;
```

# A Instrução if

- O teste é freqüentemente uma expressão relacional
  - As mesmas usadas nos laços de repetição
  - Ele também é convertido para um valor booleano:
    - Zero é convertido para false
    - Qualquer valor não nulo é convertido para true

```
if (teste)  
    corpo;
```

- Se o teste é falso a instrução (ou bloco de instruções) é simplesmente saltada

# A Instrução if

```
#include <iostream>
using namespace std;

int main()
{
    char ch;
    int espacos = 0;
    int total = 0;

    cin.get(ch);
    while (ch != '.') // encerra no final da frase
    {
        if (ch == ' ') // verifica se ch é um espaço
            espacos++;
        total++;         // total de caracteres
        cin.get(ch);
    }
    cout << espacos << " espaços e " << total << " caracteres na frase.\n";
}
```

# A Instrução if

- Saída do Programa:

O desenhista era  
um visionário.

3 espaços e 30 caracteres na frase.

- **espacos++**  
é executado apenas  
quando o caractere é  
um espaço
- **total++**  
é executado para cada  
repetição do laço while

# A Instrução if else

- O **if else** decide qual entre **duas instruções** (ou dois blocos de instruções) é executada

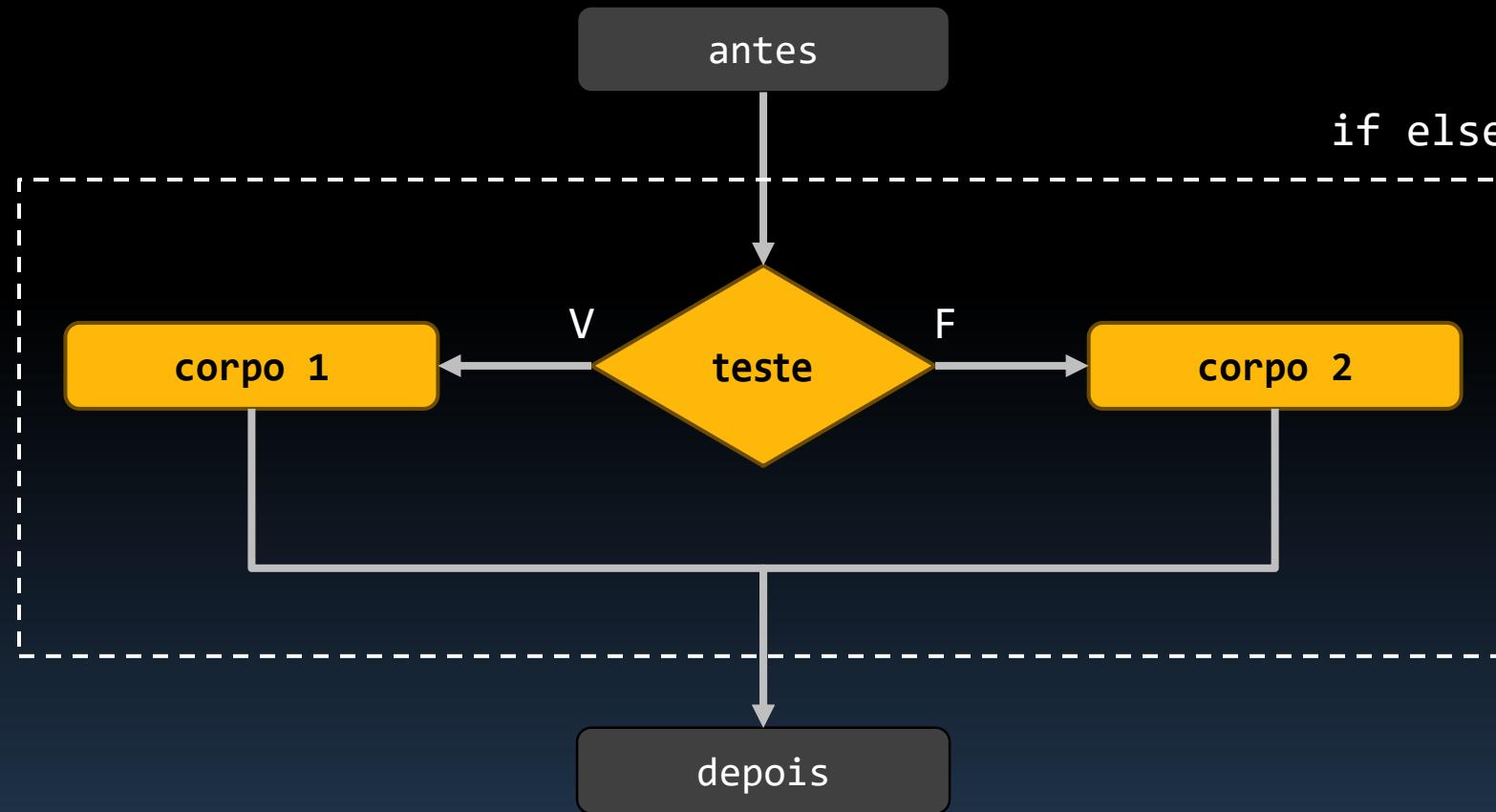
Verifica se o teste é verdadeiro

```
if (teste)
    corpo1; ——
else
    corpo2;
```

Instrução executada  
se o teste é verdadeiro

Instrução executada  
se o teste é falso

# A Instrução if else



```
antes;  
if (teste)  
    corpo1;  
else  
    corpo2;  
depois;
```

# A Instrução if else

```
#include <iostream>
using namespace std;

int main()
{
    char ch;
    cout << "Digite, que eu repito.\n";

    while ((ch = cin.get()) != '.') // encerra no final da frase
    {
        if (ch == '\n')
            cout << ch;           // exibe a nova linha
        else
            cout << ++ch;         // exibe outro caractere
    }
    cout << "\nPor favor desculpe a confusão.\n";
}
```

# A Instrução if else

- Saída do Programa:

Digite, que eu repito.

Eu estou muito satisfeito

Fv!ftupv!nvjup!tbujtgfjup

em usar este maravilhoso computador.

fn!vtbs!ftuf!nbsbwjmiptp!dpnqvubeps

Por favor desculpe a confusão.

- E se usarmos **ch + 1** no lugar de **++ch**

- O resultado é o mesmo?

# Formatando if else

- Para executar mais de uma instrução é preciso usar **blocos**
  - Existem várias formas de **organizar os blocos**
    - Enfatiza a estrutura de blocos:

```
if (ch == 'S')
{
    afavor++;
    cout << "Mais um a favor.\n";
}
else
{
    contra++;
    cout << "Este é contra.\n";
}
```

# Formatando if else

- Para executar mais de uma instrução é preciso usar **blocos**
  - Existem várias formas de **organizar os blocos**
    - Aproxima o bloco das palavras-chave **if** e **else**:

```
if (ch == 'S') {  
    afavor++;  
    cout << "Mais um a favor.\n";  
}  
else {  
    contra++;  
    cout << "Este é contra.\n";  
}
```

# Formatando if else

- Para executar mais de uma instrução é preciso usar **blocos**
  - Existem várias formas de **organizar os blocos**
    - Aproxima os blocos um do outro:

```
if (ch == 'S') {  
    afavor++;  
    cout << "Mais um a favor.\n";  
} else {  
    contra++;  
    cout << "Este é contra.\n";  
}
```

# Aninhando if else

- A instrução `if else` pode ser aninhada se for necessário escolher entre mais de duas opções

```
if (ch == 'A')  
    letraA++;
```

```
else
```

```
    if (ch == 'B')  
        letraB++;  
    else  
        outra++;
```

```
if (teste)  
    corpo1;  
else  
    corpo2;
```

Um if (ou if else) é tratado como uma única instrução

# Aninhando if else

- A construção if else pode ser **reorganizada** assim:

```
if (ch == 'A')
    letraA++;
else
    if (ch == 'B')
        letraB++;
else
    if (ch == 'C')
        letraC++;
```



```
if (ch == 'A')
    letraA++;
else if (ch == 'B')
    letraB++;
else if (ch == 'C')
    letraC++;
```

- Porém não existe uma instrução chamada **else if**

# Aninhando if else

```
#include <iostream>
using namespace std;
const int Fav = 27;
int main()
{
    int n;
    cout << "Digite um número entre 0 e 100: ";
    do
    {
        cin >> n;
        if (n < Fav)
            cout << "Muito baixo, tente novamente: ";
        else if (n > Fav)
            cout << "Muito alto, tente novamente: ";
        else
            cout << Fav << " é o meu favorito!\n";
    }
    while (n != Fav);
}
```

# Aninhando if else

- Saída do Programa:

```
Digite um número entre 0 e 100: 50
```

```
Muito alto, tente novamente: 25
```

```
Muito baixo, tente novamente: 37
```

```
Muita alto, tente novamente: 31
```

```
Muito alto, tente novamente: 27
```

```
27 é meu favorito!
```

- Reverter a comparação de igualdade previne erros

```
if (num = 3)    // atribuição indesejada  
if (3 == num)  // mesmo que if (num == 3)  
if (3 = num)   // erro na compilação
```

# Resumo

- As instruções condicionais permitem executar desvios na execução de um programa
  - if
  - if else
  - if's e if-else's aninhados
- Desvios são utilizados para tomar decisões
  - Constituem a forma mais básica de inteligência que podemos fornecer ao computador