

Estruturas de controle de fluxo.

Estruturas de seleção:

if

if/else

switch

Estruturas de controle

- Normalmente, os comandos em um programa são executados um depois do outro, na sequência em que estão escritos.
- Isto é chamado de execução sequencial.
- Vários comandos em C/C++ permitem ao programador especificar que o próximo comando a ser executado poder ser um outro, que não é o próximo na sequência.
- Isto se chama de **transferência de controle**.

Estruturas de seleção

C/C++ oferece três tipos de estruturas de seleção.

- A estrutura de seleção **if**
 - executa (seleciona) uma ação se uma condição for verdadeira (**true**)
 - ou pula a ação se a condição for falsa (**false**)
- A estrutura de seleção **if / else**
 - executa uma ação se uma condição for verdadeira (**true**)
 - e executa uma ação diferente se a condição for falsa (**false**)
- A estrutura de seleção **switch** executa uma de varias ações diferentes, dependendo do valor de uma expressão inteira.

Tomada de decisões

- A estrutura **if** permite a um programa tomar decisões com base na veracidade ou falsidade de alguma condição.
- Se a condição é verdadeira (**true**), o comando no corpo da estrutura **if** é executado.
- Se a condição for falsa (**false**), o comando do corpo não é executado.
- As condições em estruturas **if** podem ser definidas usando os operadores de igualdade e os operadores relacionais, resumidos a seguir.

Tomada de decisões: os operadores relacionais e de igualdade

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

Tomada de decisões: os operadores relacionais e de igualdade

Erro comum de programação

- Ocorrerá um erro de sintaxe se qualquer um dos operadores `==`, `!=`, `>=` e `<=` aparece com espaços entre seus dois símbolos.

Erro comum de programação

- Inverter a ordem do par de símbolos em qualquer dos operadores `!=`, `>=` e `<=` (escrevendo-os como `=!`, e `=<`, respectivamente) normalmente é um erro de sintaxe.

Estrutura de seleção **if**

Estrutura de seleção **if**

- A estrutura de seleção **if** (se) é uma estrutura de seleção única.
- Essa estrutura seleciona ou ignora um comando.
- Caso seja necessário trabalhar com mais de um comando os mesmos deverão ser agrupados em um bloco.

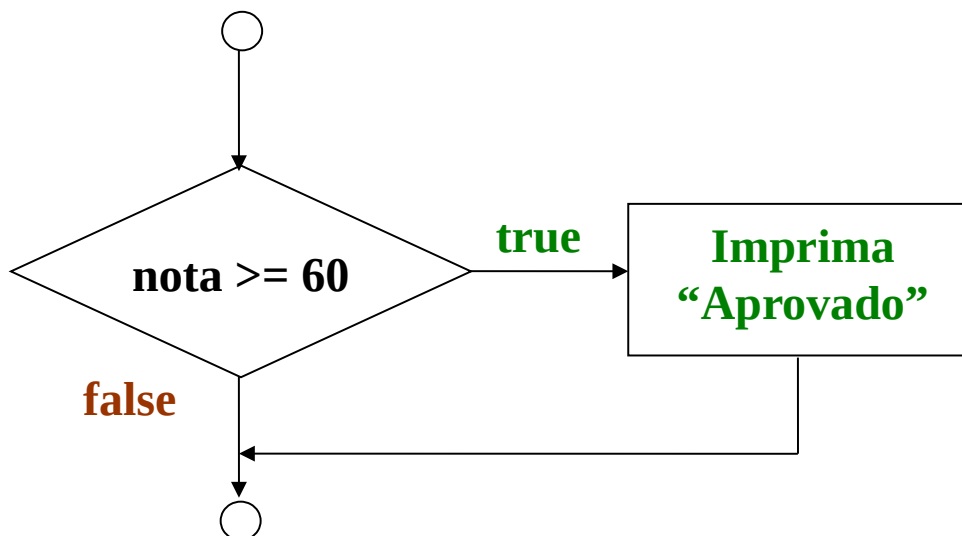
Estrutura de seleção **if**

- **Exemplo 1:** Analisar se a nota do aluno é suficiente para ser aprovado em uma prova. Considera-se aprovado quem tirar uma nota igual ou superior a 60.
- Comando em pseudocódigo:
Se a nota do estudante é maior que ou igual a 60
Imprima “Aprovado”
- Se a condição é verdadeira - **true**, então é impresso “Aprovado” e o próximo comando na sequência é “executado”.
- Se a condição é falsa - **false**, o comando de impressão é ignorado e o próximo comando na sequência é executado.
- O comando do pseudocódigo pode ser escrito em C/C++ como

```
if (nota >= 60)  
    printf("Aprovado \n");
```

```
if ( nota >= 60 )  
    cout << "Aprovado";
```

Estrutura de seleção **if**: fluxograma



Exemplo 1:

- Note que a estrutura **if** é uma estrutura de entrada e saída únicas.
- Na verdade, uma decisão pode ser tomada com base em qualquer expressão
 - se o valor da expressão é zero, ela é tratada como **false**
 - se o valor da expressão não é zero, ela é tratada como **true**.

Estrutura de seleção **if**

Exemplo 1:

```
1 //estrutura de controle if
2 #include <stdio.h>
3
4 int main()
5 {
6     // declaração de variaveis
7     float nota; // notas
8
9     printf("Estrutura de controle IF \n\n");
10
11     // Entrada de dados: nota
12     printf("Informe a nota: ");
13     scanf("%f",&nota);
14
15     if (nota >= 60)
16         printf("Aprovado \n");
17
18     return 0;
19 }
```

Estrutura de seleção **if**

Exemplo 2 (exercício):

- Faça um programa que receba um número inteiro e verifique se esse número é **positivo**.

Estrutura de seleção **if/else**

A estrutura de seleção **if / else**

- A estrutura de seleção **if / else** (se/senão) é uma estrutura de seleção dupla - seleciona entre duas ações diferentes.
- Permite ao programador especificar que uma ação deve ser executada quando a condição é verdadeira - **true** e uma outra ação quando a condição é falsa – **false**.

A estrutura de seleção **if / else**

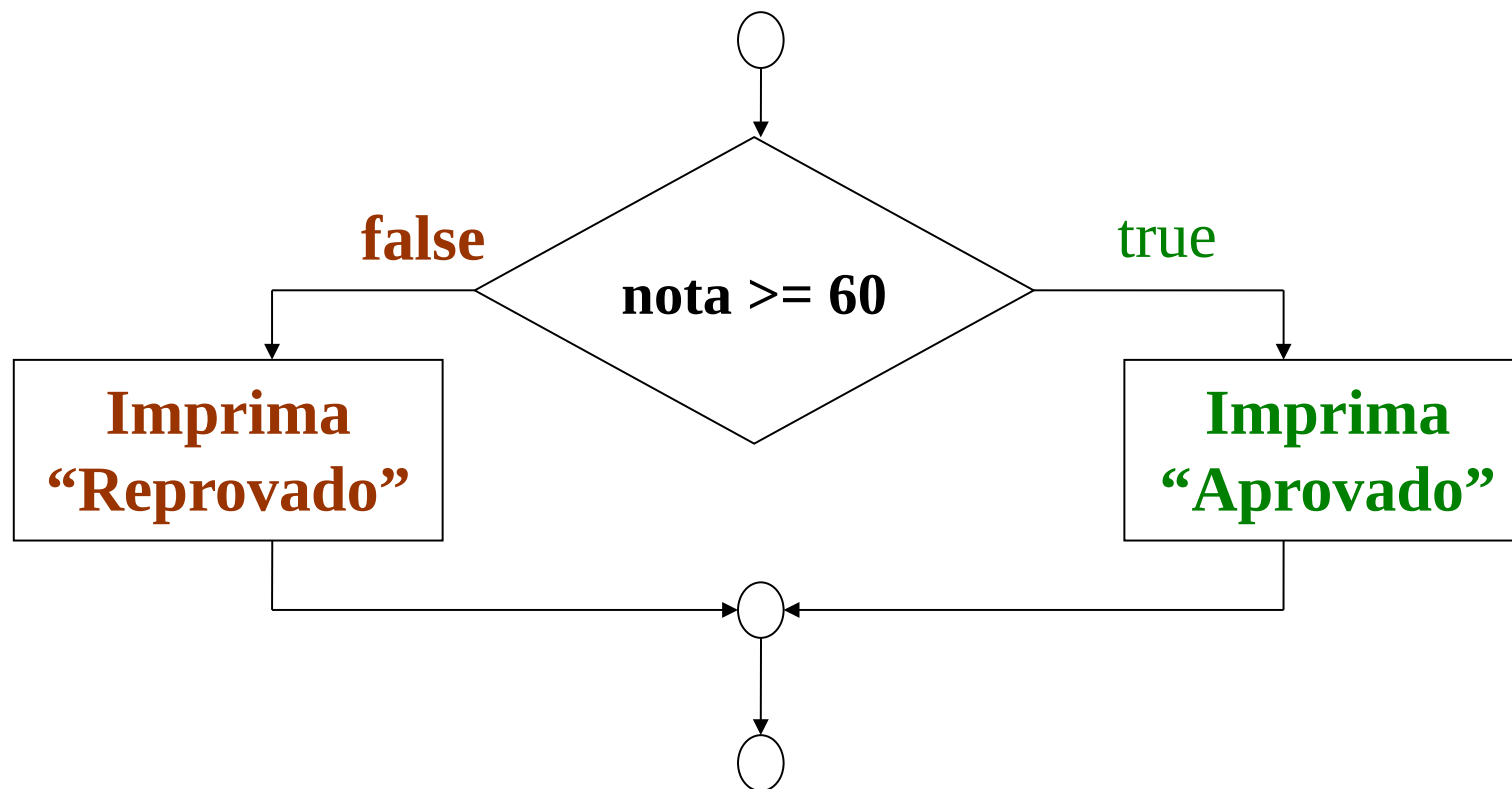
- **Exemplo 3:** Analisar se o aluno foi aprovado ou reprovado em uma prova. Considera-se aprovado quem tirar uma nota igual ou superior a 60.
- Por exemplo, o comando de pseudocódigo:
Se a nota do estudante é maior que ou igual a 60
 Imprime “Aprovado”
senão
 imprime “Reprovado”
imprime **Aprovado** se a nota do estudante é maior que ou igual a 60
e imprime **Reprovado** se a nota do estudante é menor que 60.
- A estrutura **if / else** pode ser escrita em C/C++ como:

```
if (nota >= 60)
    printf("Aprovado \n");
else
    printf("Reprovado \n");
```

```
if ( nota >= 60 )
    cout << "Aprovado";
else
    cout << "Reprovado";
```

A estrutura de seleção **if / else**:

Exemplo 3: Fluxograma



- imprime **Aprovado** se a nota do estudante é maior que ou igual a 60
- imprime **Reprovado** se a nota do estudante é menor que 60.

A estrutura de seleção **if / else**:

Exemplo 3: código

```
1 //estrutura de controle if else
2 #include <stdio.h>
3
4 int main()
5 {
6     // declaração de variaveis
7     float nota; // notas
8
9     printf("Estrutura de controle IF ELSE \n");
10
11     // Entrada de dados: nota
12     printf("Informe a nota: ");
13     scanf("%f",&nota);
14
15     // if else
16     if (nota >= 60)
17         printf("Aprovado \n");
18     else
19         printf("Reprovado \n");
20
21     return 0;
22 }
```

A estrutura de seleção **if / else**:

Exemplo 4 (exercício):

- Faça um programa que receba um número inteiro e verifique se esse número é **positivo** ou **negativo**.

A estrutura de seleção **if / else**

Estruturas **if / else** aninhadas testam múltiplos casos colocando estruturas de seleção **if / else** dentro de outras estruturas de seleção **if / else**.

Exemplo 5: Dependendo da nota digitada pelo usuário:

- Imprimirá A para notas maiores que ou iguais a 90,
- B para notas no intervalo de 80 a 89,
- C para notas no intervalo 70 a 79,
- D para notas no intervalo 60 a 69
- e R para as demais notas.

*Se nota do estudante for maior que ou igual a 90
imprima "A"*

senão

*Se nota do estudante for maior que ou igual a 80
Imprima "B"*

senão

*Se nota do estudante for maior que ou igual a 70
Imprima "C"*

senão

*Se nota do estudante for maior que ou igual a 60
Imprima "D"*

senão

Imprima "R"

```
if ( nota >= 90)
    printf("A");
else
    if ( nota >= 80)
        printf("B");
    else
        if ( nota >= 70)
            printf("C");
        else
            if ( nota >= 60)
                printf("D");
            else
                printf("R");
```

A estrutura de seleção **if / else**

```
if ( nota >= 90)
    printf("A");
else
    if ( nota >= 80)
        printf("B");
    else
        if ( nota >= 70)
            printf("C");
        else
            if ( nota >= 60)
                printf("D");
            else
                printf("R");
```

Se a nota for maior ou igual que 90 só o comando de impressão depois do primeiro teste será executado.
(mesmo que todas as outras condições serão **true**)

Se a nota não for maior ou igual que 90 a parte **else** do comando **if / else** será executado.

A estrutura de seleção **if/else**

- Outra forma de escrever a mesma estrutura:

```
if ( nota >= 90)
    printf("A");
else if ( nota >= 80)
    printf("B");
else if ( nota >= 70)
    printf("C");
else if ( nota >= 60)
    printf("D");
else
    printf("R");
```

- As duas formas são equivalentes.
- A segunda forma evita os “níveis” de deslocamento do código para a direita.

Exemplo 5:

```
1 //Exemplos, estruturas de controle
2 #include <stdio.h>
3
4 int main()
5 {
6     // declaração de variaveis
7     float nota; // notas
8
9     // Entrada de dados: nota
10    printf("Informe a nota: ");
11    scanf("%f",&nota);
12
13    if ( nota >= 90)
14        printf("A");
15    else if ( nota >= 80)
16        printf("B");
17    else if ( nota >= 70)
18        printf("C");
19    else if ( nota >= 60)
20        printf("D");
21    else
22        printf("R");
23
24    return 0; // indica que o programa terminou com sucesso
25 }
```

A estrutura de seleção **if / else**

Dica de desempenho

- Em uma estrutura **if / else** aninhada, teste as condições que são mais prováveis de serem verdadeiras no início da estrutura **if / else** aninhada.
- Isso permitirá que a estrutura **if / else** aninhada seja executada mais rapidamente.
- Seria menos eficiente testar os casos pouco frequentes primeiro.

A estrutura de seleção **if / else**

- A estrutura de seleção **if** espera somente um comando no seu corpo.
- Para incluir vários comandos no corpo de um **if**, inclua os comandos entre chaves (**{** e **}**).
- Um conjunto de comandos entre chaves é chamado de **comando composto** ou **bloco**.

Tomada de decisões

- Se estrutura `if` ou `if else` contem corpo composto de múltiplos comandos devemos colocar os comandos que integram o corpo entre um par de chaves `{ }`

Boa prática de programação

Em um programa não deve haver mais que um comando por linha.

Erro comum de programação

- ❖ Colocar um ponto e vírgula imediatamente após o parêntese da direita em uma estrutura `if`, é frequentemente um erro de lógica (embora não seja um erro de sintaxe).
- ❖ O ponto e vírgula pode fazer com o corpo da estrutura `if` seja considerado vazio, de maneira que a estrutura `if` não execute nenhuma ação, independentemente do fato da condição ser verdadeira ou não.
- ❖ Pior ainda, o comando do corpo original da estrutura `if` se tornaria agora um comando em seguida à estrutura `if`, e seria sempre executado (frequentemente levando o programa a produzir resultados incorretos).

A estrutura de seleção **if / else**

Exemplo (exercício) 6:

- Incluir um comando composto de dois operadores de impressão em uma estrutura **if / else** no exercício anterior:

```
if ( nota >= 60 )
{
    printf ( " Aprovado.\n " );
    printf ( " Parabens!!! \n " );
}
else
{
    printf ( " Reprovado.\n" );
    printf ( " Você deve fazer este curso de novo.\n" );
}
```

Operador ternario (? :)

- C/C++ oferece operador condicional (? :) bastante semelhante à estrutura `if / else`.
- O operador condicional é o único operador ternário (ele aceita três operandos)
- Os operandos, juntamente com o operador condicional, formam uma expressão condicional:
 - o primeiro operando é uma condição;
 - o segundo operando é o valor para a expressão condicional, se a condição é `true`;
 - o terceiro operando é o valor para a expressão condicional se a condição é `false`.
- Por exemplo:

```
printf( "%s", nota >= 60 ? "Aprovado" : "Reprovado" );
```

contém uma expressão condicional que imprime

- “Aprovado” se a condição `nota >= 60` for `true`
- “Reprovado” se a condição for `false`.

Operador ternario (? :)

Exemplo 7 (exercício):

- Utilizando **operador ternário** escreva um programa que receba um número e verifique se esse número é **positivo** ou **negativo**.

Exemplo 8:

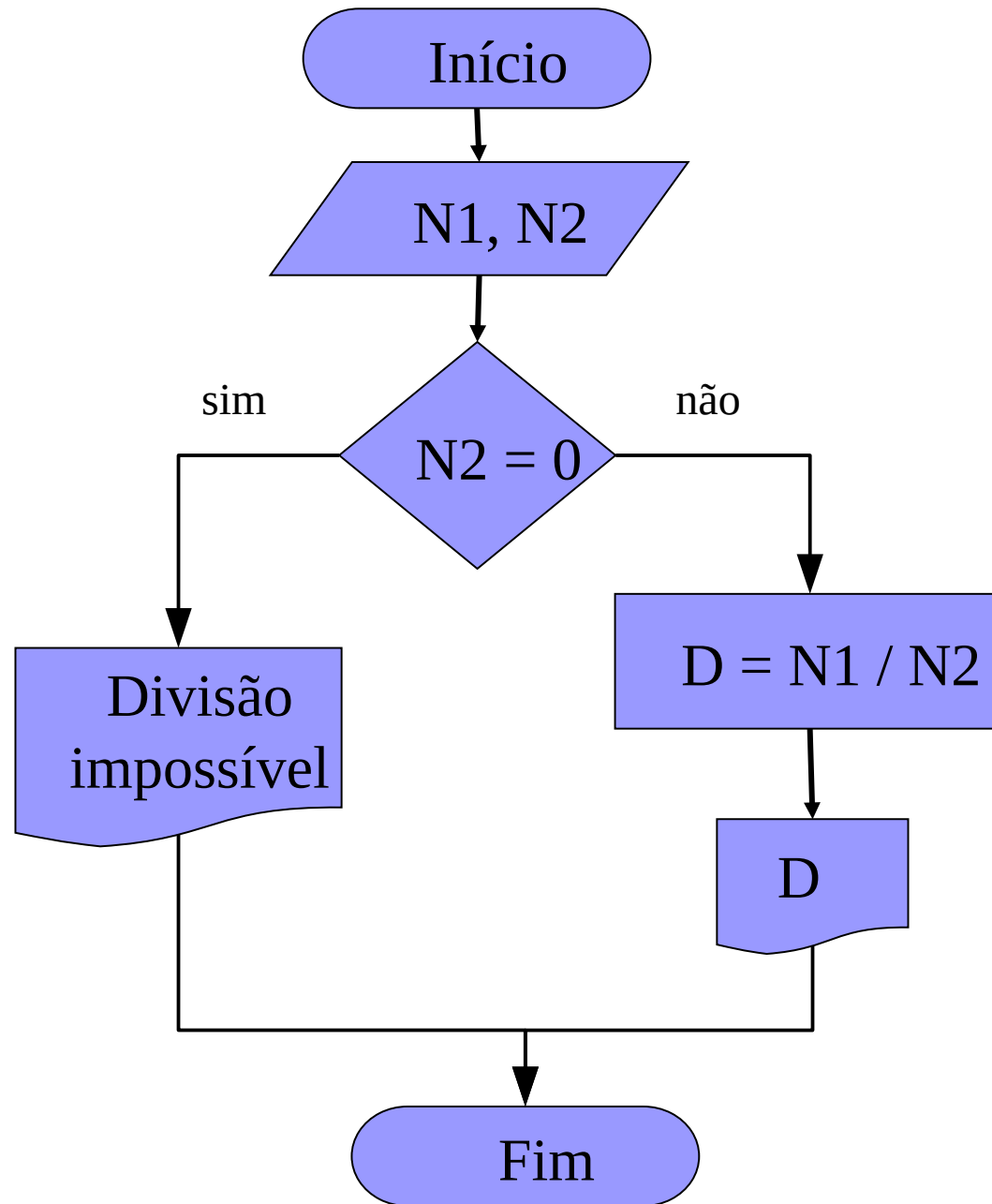
Calcular o resultado de divisão de dois números.

Algoritmo

Passo 1: Receber dois números que serão divididos.

Passo 2: Se o segundo número for igual a zero a divisão não poderá ser feita; caso contrário, dividir os números e apresentar o resultado.

Exemplo 8: Fluxograma



Exemplo 8: Pseudocódigo

```
algoritmo "divisão"
var
  n1: inteiro
  n2: inteiro
  d: real
inicio
  escreva("Digite dois números: ")
  leia(n1, n2)
  se (n2 <> 0) entao
    d <- n1 / n2
    escreval("Divisão: ", d)
  senao
    escreval("A divisão não é possível")
  fimse
finalgoritmo
```

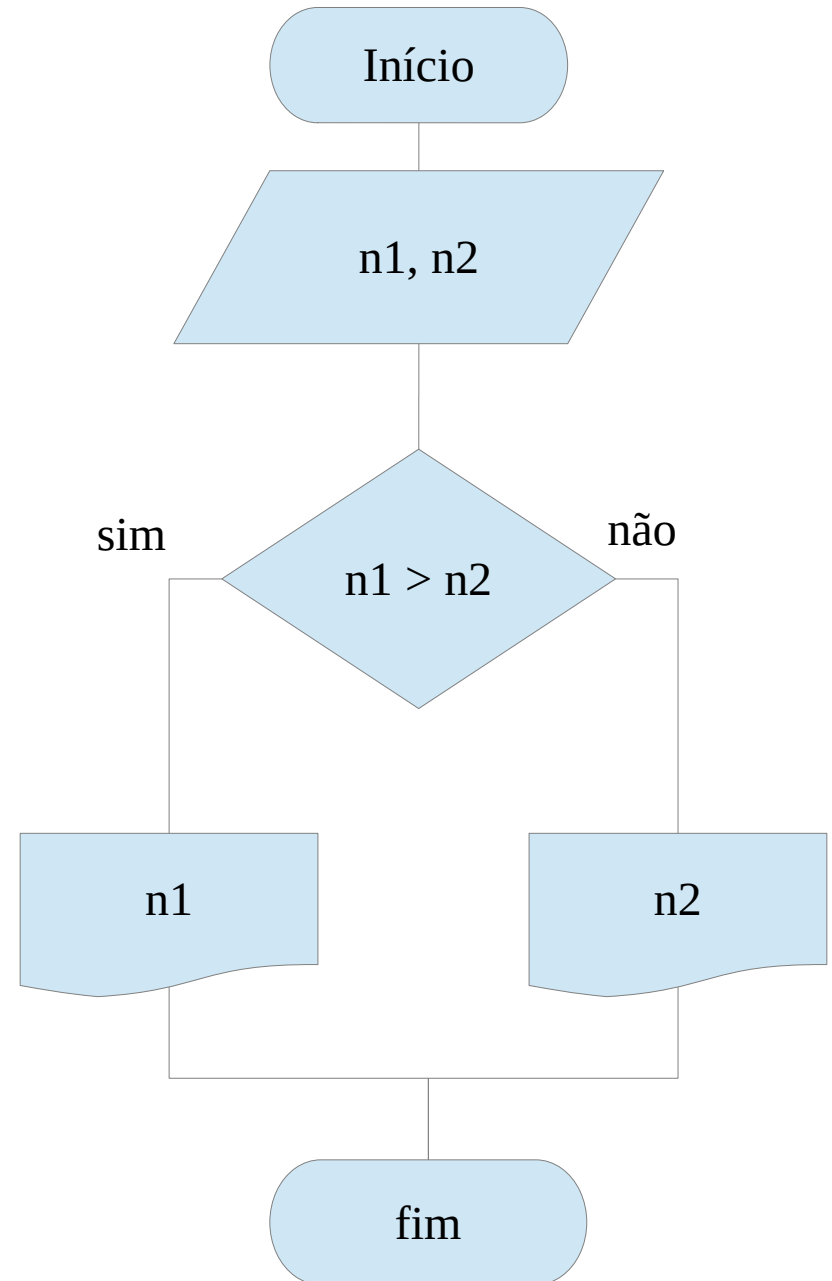
```
5  {
6  // declarações de variável
7  float n1; // primeiro numero
8  float n2; // segundo numero
9  float div; // resultado da divisão
10
11  printf("Divisão de dois números: \n ");
12  // Entrada de dados: recebe 2 numeros
13  printf("Informe o 1o numero: ");
14  scanf("%f", &n1);
15
16  printf("Informe o 2o numero: ");
17  scanf("%f", &n2);
18
19  if ( n2 != 0 )
20  {
21      div = n1 / n2;
22      printf ("\n Resultado da divisao: %f \n", div);
23  }
24  else
25      printf ("\n Divisão por 0! \n");
26
27  return 0; // indica que o programa terminou com sucesso
28 }
```


Exemplo 9:

O programa deve receber dois números e mostrar qual é o maior deles.

Exemplo 9

```
var
  n1: real
  n2: real
inicio
// Seção de Comandos
  escreva("Digite o 1. número: ")
  leia(n1)
  escreva("Digite o 2. número: ")
  leia(n2)
  escreva("Maior número: ")
  se (n1 > n2) entao
    escreval(n1)
  senao
    escreval(n2)
  fimse
finalgoritmo
```



Exemplo 9

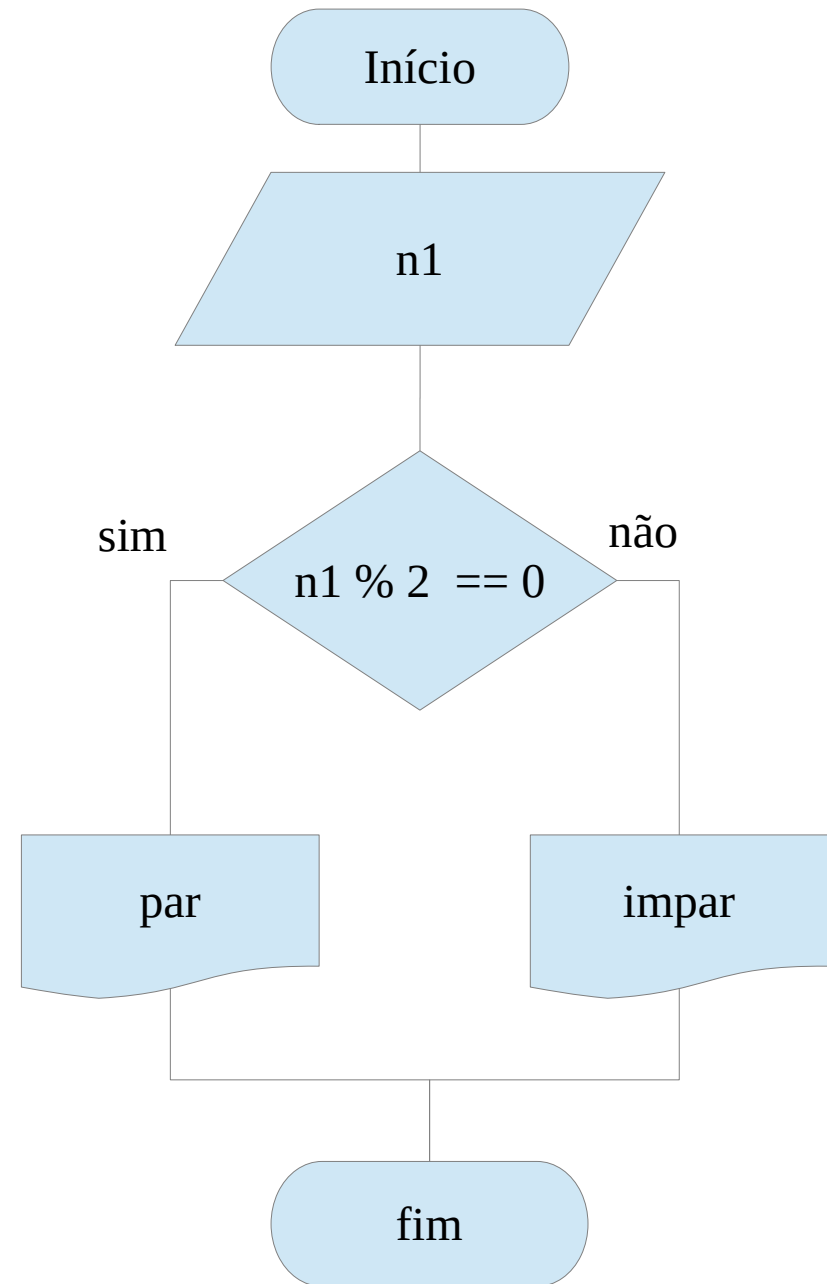
```
4  int main()  
5  {  
6      // declarações de variáveis  
7      float n1; // primeiro numero  
8      float n2; // segundo numero  
9  
10     printf("Maior de dois números: \n ");  
11     // Entrada de dados: recebe 2 numeros  
12     printf("Informe o 1o numero: ");  
13     scanf("%f", &n1);  
14  
15     printf("Informe o 2o numero: ");  
16     scanf("%f", &n2);  
17  
18     // Processamento e saída de dados  
19     printf("\n Maior numero: ");  
20     if ( n1 > n2 )  
21         printf ("%f", n1);  
22     else  
23         printf ("%f", n2);  
24  
25     return 0; // indica que o programa terminou com sucesso  
26 } // fim da função main
```

Exemplo 10:

O programa deve receber um um número inteiro e verificar (mostrar uma mensagem) se o número é par ou ímpar.

Exemplo 10

```
var  
n1: inteiro  
  
inicio  
// Seção de Comandos  
  escreva("Digite 1 número: ")  
  leia(n1)  
  
  se (n1 % 2 = 0) entao  
    escreval ("O número", n1, " é par")  
  senao  
    escreval("O número", n1, " é ímpar")  
  
fimse  
fimalgoritmo
```



Exemplo 10

```
4  int main()  
5  {  
6      // declarações de variáveis  
7      int n1; // numero  
8  
9      printf("O programa detecta se número é par ou impar: \n ");  
10     // Entrada de dados: recebe número inteiro  
11     printf("Informe o número: ");  
12     scanf("%i", &n1);  
13  
14     // Processamento e saída de dados  
15     if ( n1 % 2 == 0 )  
16         printf ("\n Par \n ");  
17     else  
18         printf ("\n Impar \n ");  
19  
20     return 0;  
21 }
```

Operadores lógicos

- Para testar condições múltiplas ao tomar uma decisão, executamos estes testes em comandos separados, ou em `if` ou estruturas `if/else` aninhadas, C/C++ oferece operadores lógicos que são usados para especificar condições mais complexas combinando condições simples.
- Os operadores lógicos são:
 - ➔ `&&` (E lógico, AND)
 - ➔ `||` (OU lógico, OR)
 - ➔ `!` (NAO lógico, também chamado de negação lógica, NOT)

Operadores lógicos: **AND (E)**

- Suponhamos que desejemos nos assegurar de que duas condições são ambas **true**, antes de escolhermos um certo caminho de execução.
- Neste caso, podemos usar o operador **E** lógico **&&**.
- A tabela mostra todas as quatro combinações possíveis de valores **false** e **true** para a **expressão1** e a **expressão2**.
- Tais tabelas são frequentemente chamadas de tabelas verdade.

expressão 1	expressão2	expressão1 && expressão2
0	0	0
0	1	0
1	0	0
1	1	1

Operadores lógicos: **OR (OU)**

- Agora vamos considerar o operador **|| (OU logico)**.
- Suponhamos que desejemos nos assegurar, em um certo ponto de um programa, que ou uma ou ambas de duas condições são **true** antes de escolhermos um certo caminho de execução.
- Neste caso, usamos o operador **|| (OU logico)**.

expressão 1	expressão2	expressão1 expressão2
0	0	0
0	1	1
1	0	1
1	1	1

Operadores lógicos: **NÃO (NOT)**

- C/C++ oferece o operador **!** (**negação lógica**) para possibilitar ao programador “inverter” o significado de uma condição.
- Diferentemente dos operadores **&&** e **||** que combinam duas condições (operadores binários), o operador de negação lógica tem somente uma única condição como operando (operador unário).

expressão	! expressão
0	1
1	0

Exemplo 11:

O programa deve receber três números inteiros e mostrá-los em ordem crescente.

Suponha que o usuário digitará três números diferentes.

Exemplo 11

```
var
  n1: real
  n2: real
  n3: real
inicio
// Seção de Comandos
  escreva("Digite o 1. número: ")
  leia(n1)
  escreva("Digite o 2. número: ")
  leia(n2)
  escreva("Digite o 3. número: ")
  leia(n3)
  escreval("Resultado: ")
```

Exemplo 11

```
se (n1 < n2) e (n1 < n3) e (n2 < n3) entao
    escreval(n1, n2, n3)
```

senao

```
se (n1 < n2) e (n1 < n3) e (n2 > n3)  entao
    escreval(n1,n3,n2)
```

senao

```
se (n1 < n2) e (n1 > n3) e (n2 > n3) entao
    escreval(n3,n1,n2)
```

senao

```
se (n1 > n2) e (n1 < n3) e (n2 < n3) entao
    escreval(n2,n1,n3)
```

senao

```
se (n1 > n2) e (n1 > n3) e (n2 < n3) entao
    escreval(n2,n3,n1)
```

senao

```
se (n1 > n2) e (n1 > n3) e (n2 > n3) entao
    escreval(n3,n2,n1)
```

fimse

fimse

fimse

fimse

fimse

fimse

fimalgoritmo

Exemplo 11

```
6 // declarações de variáveis
7 int n1, n2, n3; // numeros
8
9 printf("Mostra os números em ordem crescente: \n ");
10 // Entrada de dados: recebe 3 numeros
11 printf("Informe o 1o numero: ");
12 scanf("%i", &n1);
13
14 printf("Informe o 2o numero: ");
15 scanf("%i", &n2);
16
17 printf("Informe o 3o numero: ");
18 scanf("%i", &n3);
19
20 // Processamento e saída de dados
21 printf ("\n Numeros em ordem crecente: ");
22
23 if ( (n1 < n2) && (n1 < n3) && (n2 < n3))
24     printf("%i, %i, %i \n", n1, n2, n3);
25 else if ( (n1 < n2) && (n1 < n3) && (n2 > n3))
26     printf("%i, %i, %i \n", n1, n3, n2);
27 else if ( (n1 < n2) && (n1 > n3) && (n2 > n3))
28     printf("%i, %i, %i \n", n3, n1, n2);
29 else if ( (n1 > n2) && (n1 < n3) && (n2 < n3))
30     printf("%i, %i, %i \n", n2, n1, n3);
31 else if ( (n1 > n2) && (n1 > n3) && (n2 < n3))
32     printf("%i, %i, %i \n", n2, n3, n1);
33 else if ( (n1 > n2) && (n1 > n3) && (n2 > n3))
34     printf("%i, %i, %i \n", n3, n2, n1);
35
```

Exemplo 12 (exercício)

A nota final de um estudante é calculada a partir de três notas atribuídas:

- um trabalho de laboratório (com peso 0.2)
- uma avaliação semestral (com peso 0.3)
- um exame final (com peso 0.5)

Faça um programa que receba as três notas, calcule e mostre a média ponderada e dependendo da média apresente o conceito conforme tabela abaixo:

Média Ponderada	Conceito
8.0 – 10.0	A
7.0 – 7.9	B
6.0 – 6.9	C
5.0 – 5.9	D
0.0 – 4.9	E

Exemplo 12

```
var
  n1: inteiro
  n2: inteiro
  n3: inteiro
  r : real
inicio
// Seção de Comandos
  escreva("Digite a 1. nota: ")
  leia(n1)
  escreva("Digite a 2. nota: ")
  leia(n2)
  escreva("Digite a 3. nota: ")
  leia(n3)
  r <- (n1 * 0.2 + n2 * 0.3 + n3 * 0.5)
  escreval("A média ponderada é: ",r)
  se (r < 5) entao
    escreval("Conceito E")
  senao
    se (r < 6) entao
      escreval("Conceito D")
    senao
      se (r < 7) entao
        escreval("Conceito C")
```

```
    senao
      se (r < 8) entao
        escreval("Conceito B")
      senao
        se (r <=10) entao
          escreval("Conceito A")
        fimse
      fimse
    fimse
  fimse
finalgoritmo
```


Exemplo 12

```
6 // declarações de variáveis
7 float n1, n2, n3; // notas
8 float r; // resultado
9
10 printf("Calcula a nota média ponderada e conceito (A, B, C, D, E): \n ");
11 // Entrada de dados: recebe 3 notas
12 printf("Nota 1 (laboratorio): ");
13 scanf("%f", &n1);
14
15 printf("Nota 2 (semestral): ");
16 scanf("%f", &n2);
17
18 printf("Nota 3 (exame final): ");
19 scanf("%f", &n3);
20
21 // Processamento e saída de dados
22 r = n1 * 0.2 + n2 * 0.3 + n3 * 0.5;
23
24 printf("\n Nota media: %f", r);
25 printf("\n Conceito : ");
26
27 if ( r < 5 )
28     printf (" E \n");
29 else if ( r < 6 )
30     printf (" D \n");
31 else if ( r < 7 )
32     printf (" C \n");
33 else if ( r < 8 )
34     printf (" B \n");
35 else if ( r <= 10 )
36     printf (" A \n");
37
```

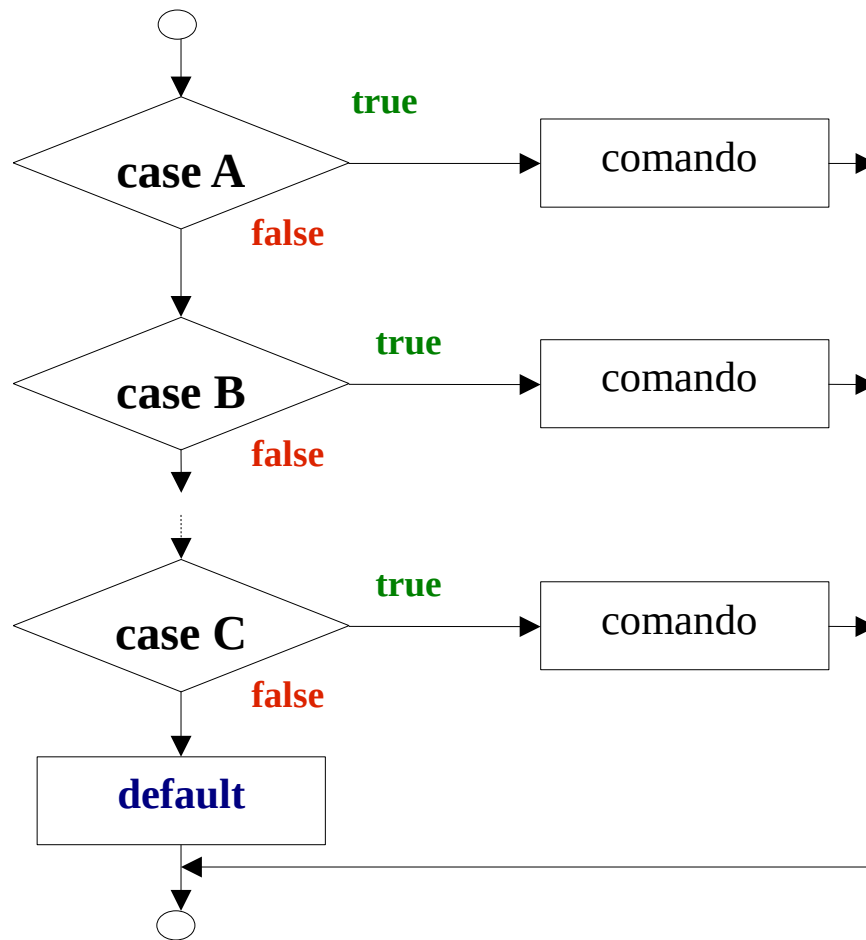
Estrutura de seleção múltipla **switch**

Estrutura de seleção múltipla **switch**

- O comando **switch** permite selecionar uma entre varias alternativas.
- O mesmo efeito pode ser atingido usando varias estruturas **if/else**.
- Dependendo do caso, uso do comando **switch** deixa o código do programa mais “limpo” e claro.
- A estrutura **switch** consiste em uma série de rótulos **case** e um caso **default** opcional.

Estrutura de seleção múltipla **switch**

```
switch ( condição )  
{  
    case A:  
        comando;  
        break;  
    case B:  
        comando;  
        break;  
    case C:  
        comando;  
        break;  
    default:  
        comando;  
        break;  
}
```



- O fluxograma da estrutura de seleção múltipla switch genérica (usando um **break** em cada **case**)
- O fluxograma torna claro que cada comando **break** no fim de um **case** faz com que o controle saia imediatamente da estrutura **switch**.
- **Boa prática de programação**
- Forneça um caso **default** em comandos **switch**.
- Os casos não-testados explicitamente em um comando **switch** sem um caso **default** são ignorados.

Exercício 13

```
1 // switch
2 #include <stdio.h>
3
4 int main ()
5 {
6     int n;
7
8     printf("\n Digite um numero inteiro de 1 ate 5: ");
9     scanf("%i", &n);
10
11     switch (n)
12     {
13         case 1:
14             printf("\n Um \n");
15             break;
16         case 2:
17             printf("\n Dois \n");
18             break;
19         case 3:
20             printf("\n Tres \n");
21             break;
22         default:
23             printf("\n Numero invalido! \n");
24             break;
25     }
26
27     return 0;
28 }
```

Estrutura de seleção múltipla **switch**

- A palavra-chave **switch** é seguida pelo nome de variável **n** entre parênteses.
- Isto é chamado de **expressão de controle**.
- O valor desta expressão é comparado com cada um dos rótulos **case**.
- Assuma que o usuário digitou numero **1**
- **1** é automaticamente comparado a cada case no **switch**.
- Se ocorre uma igualdade (**case 1**), os comandos para esse case são executados.
- Para caso 1 uma mensagem “Um” vai ser exibida e a estrutura **switch** imediatamente termina com o comando **break**.
- Note que, diferentemente de outras estruturas de controle, não é necessário incluir um comando **case** múltiplo entre chaves.

Exercício 14

Faça um programa que receba:

- o código correspondente ao cargo de um funcionário
- e seu salário atual

e calcula/mostra de acordo com a tabela:

- cargo
- valor do aumento
- novo salário (com aumento)

Código	Cargo	Percentual
1	Escriturário	50%
2	Secretário	35%
3	Caixa	20%
4	Gerente	10%
5	Diretor	Não tem aumento


```

6   int c;      // código
7   float s;    // salario
8   float s_n;  // novo salario
9   float a;    // aumento

10
11  printf("\n Salario atual: ");
12  scanf("%f", &s);

13
14  printf("\n Escolha o cargo do funcionario: \n"
15         "1 - Escrituario \n "
16         "2 - Secretario \n");
17  scanf("%i", &c);

18
19  switch (c)
20  {
21      case 1:
22          a = s * 0.5;
23          s_n = s + a;
24          printf ("\n Cargo: 1 - Escrituario  \n");
25          printf ("\n Valor de aumento:  %.2f \n", a);
26          printf ("\n Novo salario:  %.2f \n", s_n );
27          break;

28
29      case 2:
30          a = s * 0.35;
31          s_n = s + a;
32          printf ("\n Cargo: 2 - Secretario  \n");
33          printf ("\n Valor de aumento:  %.2f \n", a);
34          printf ("\n Novo salario:  %.2f \n", s_n );
35          break;

36
37      default:
38          printf(" \n Código de cargo invalido! \n ");
39          break;
40  }
41  return 0;

```