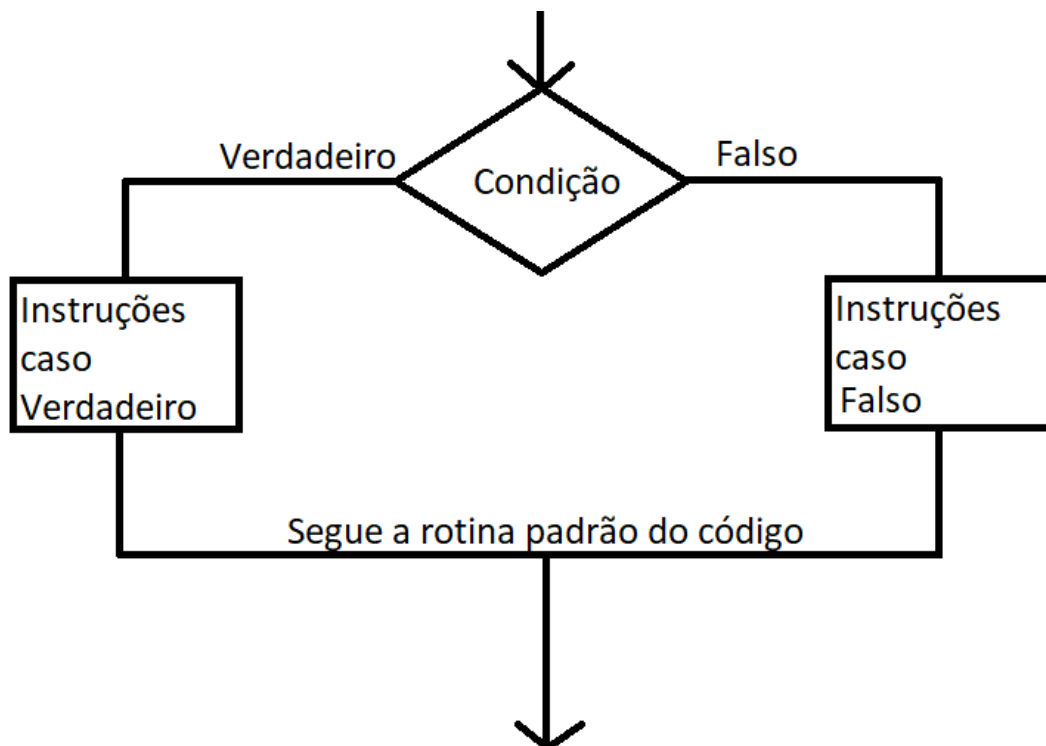


Módulo 3 - Comandos de controle

Estrutura condicional

Até agora estávamos vendo programas que funcionavam linearmente, isto é, executam os comandos um a um até chegar ao fim. Mas existem estruturas capazes de alterar o andamento de nosso programa de acordo com uma ou mais condições. Veremos a frente quais são elas.

Começando pela estrutura `if`. Ela executa o trecho de código em seu bloco (dentro das chaves) somente se uma condição for verdadeira. Veja a representação:



Estrutura Condicional `if`:

O `if` é a estrutura condicional mais básica. Ela permite que você execute um bloco de código somente se uma condição for verdadeira. Aqui está a sintaxe básica:

```
if (condicao) {  
    // Código a ser executado se a condição for verdadeira  
}
```

Estrutura Condicional `if-else` :

O `if-else` permite que você execute um bloco de código se a condição for verdadeira e outro bloco de código se a condição for falsa. Aqui está a sintaxe:

```
if (condicao) {  
    // Código a ser executado se a condição for verdadeira  
} else {  
    // Código a ser executado se a condição for falsa  
}
```

Estrutura Condicional `if-else if-else` :

Essa estrutura é usada quando você tem várias condições para verificar em uma sequência. Ela permite que você teste várias condições em ordem e execute o bloco de código correspondente à primeira condição verdadeira. Aqui está a sintaxe:

```
if (condicao1) {  
    // Código a ser executado se a condição1 for verdadeira  
} else if (condicao2) {  
    // Código a ser executado se a condição2 for verdadeira  
} else {  
    // Código a ser executado se nenhuma das condições anteriores for verdadeira  
}
```

Para checar uma condição, é utilizado os **operadores relacionais**. Os operadores relacionais, também conhecidos como operadores de comparação, são utilizados em programação para comparar valores e produzir resultados booleanos (verdadeiro ou falso) com base em uma condição de comparação. Eles são amplamente usados em estruturas condicionais, como `if`, `while`, `for`, etc., para tomar decisões com base em testes de igualdade, desigualdade, maior que, menor que e assim por diante.

O compilador C considera todos os valores diferente de zero como verdadeiro, e o zero como falso. Portanto, as as operações relacionais sempre irão retornar algo igual ou diferente de zero.

Os operadores relacionais da linguagem C são:

1. **`==` (igual a):** Este operador verifica se dois valores são iguais.

Exemplo: `5 == 5` é verdadeiro, enquanto `5 == 10` é falso.

2. **`!=` (diferente de):** Este operador verifica se dois valores não são iguais.

Exemplo: `5 != 10` é verdadeiro, enquanto `5 != 5` é falso.

3. **> (maior que):** Este operador verifica se o valor da esquerda é maior que o valor da direita.

Exemplo: `10 > 5` é verdadeiro, enquanto `5 > 10` é falso.

4. **< (menor que):** Este operador verifica se o valor da esquerda é menor que o valor da direita.

Exemplo: `5 < 10` é verdadeiro, enquanto `10 < 5` é falso.

5. **>= (maior ou igual a):** Este operador verifica se o valor da esquerda é maior ou igual ao valor da direita.

Exemplo: `10 >= 10` é verdadeiro, enquanto `5 >= 10` é falso.

6. **<= (menor ou igual a):** Este operador verifica se o valor da esquerda é menor ou igual ao valor da direita.

Exemplo: `5 <= 10` é verdadeiro, enquanto `10 <= 5` é falso.

```
// > -> Maior que
// >= -> Maior ou igual que
// < -> Menor que
// <= -> Menor ou igual que
// == -> Igual
// != -> Diferente

#include <stdio.h>

// Para mostrar como C enxerga as relações
int main(void){
    int a = 1;
    int b = 1;

    printf("%d", 33 == 33); // Aqui só resulta em 1 e 0;
}
```

Testando com duas variáveis quem é maior que quem:

```
int x = 20;
int y = 10;

if(x > y){
    printf("x é maior que y");
} else {
    printf("x é menor que y");
}
```

É possível ter ifs dentro de outros if:

```
// Programa que testa uma entrada e compara mais dois numeros
#include <stdio.h>

int main(void){
    int a, b, c;

    printf("Insira a chave: ");
    scanf("%d", &a);

    if(a == 5){
        printf("\nAgora digite o valor de A: ");
        scanf("%d", &b);

        printf("\nAgora digite o valor de B: ");
        scanf("%d", &c);

        if (b > c){
            printf("B e maior que C");
        } else if ( b == c){
            printf("B e igual a C");
        } else {
            printf("B e menor que C");
        }
    } else {
        printf("Chave incorreta");
    }
}
```

Operador ternário, if de uma linha

O operador ternário na linguagem de programação C é uma forma compacta de escrever uma expressão condicional. Ele é frequentemente utilizado para simplificar o código quando você precisa tomar uma decisão com base em uma condição. O operador ternário é representado da seguinte forma:

```
expressao_condicional ? valor_se_verdadeiro : valor_se_falso;
```

Aqui está como ele funciona:

- **expressao_condicional** é uma expressão booleana (ou seja, algo que resulta em verdadeiro ou falso).
- Se **expressao_condicional** for avaliada como verdadeira, o valor após o **?** será retornado.
- Se **expressao_condicional** for avaliada como falsa, o valor após **:** será retornado.

Aqui está um exemplo simples:

```
int idade = 18;  
char* status = (idade >= 18) ? "Adulto" : "Menor de idade";
```

Neste exemplo, se a variável `idade` for maior ou igual a 18, a expressão `(idade >= 18)` será verdadeira, e a variável `status` receberá a string "Adulto". Caso contrário, se a expressão for falsa, `status` receberá a string "Menor de idade".

O operador ternário pode ser uma maneira concisa de lidar com decisões simples em seu código, mas deve ser usado com moderação para manter a legibilidade. É importante que a expressão condicional seja clara e fácil de entender para evitar confusões.