

Expressões relacionais, Expressões lógicas e Comandos condicionais em C

Disciplina de Programação de Computadores I
Universidade Federal de Ouro Preto

Agenda

- Operadores e expressões de igualdade e relacionais
- Operadores e expressões lógicas
- Comandos condicionais: if e switch



Expressões

- Expressões:
 - constantes: `1 ; 30.1 ; 'a' ; "casa"`
 - variáveis: `int x ; char a ; float F ; double y`
 - expressões aritméticas: `10 + 5 ; x / 2.0 ; y % z`
 - expressões relacionais: realizam uma comparação entre duas expressões e retornam verdadeiro ou falso
 - expressões lógicas: realizam uma operação lógica (e, ou, negação) entre duas expressões e retornam verdadeiro ou falso

Expressões Relacionais

São expressões que realizam uma comparação entre duas expressões e retornam:

- 0, se o resultado é FALSO;
- 1 (ou qualquer número diferente de zero), se o resultado é VERDADEIRO.

Operadores de Igualdade e Relacionais

Os operadores de igualdade, em C, são:

- **==** : igual;
- **!=** : diferente;

Os operadores relacionais, em C, são:

- **>** : maior que;
- **>=** : maior ou igual que;
- **<** : menor que;
- **<=** : menor ou igual que.

Expressões de Igualdade: igual e diferente

exp1 == exp2 :

retorna 1 quando as expressões são iguais ou 0, caso contrário:

1 == 1 : retorna 1

3 == 4 : retorna 0

exp1 != exp2 :

retorna 1 quando as expressões são diferentes ou 0, caso contrário:

1 != 1 : retorna 0

3 != 4 : retorna 1

Expressões Relacionais: maior e maior ou igual

exp1 > exp2 :

retorna 1 quando exp1 é maior que exp2 ou 0, caso contrário:

5 > 3 : retorna 1

3 > 5 : retorna 0

3 > 3 : retorna 0

exp1 >= exp2 :

retorna 1 quando exp1 é maior ou igual que exp2 ou 0, caso contrário:

5 >= 3 : retorna 1

3 >= 5 : retorna 0

3 >= 3 : retorna 1

Expressões Relacionais: menor e menor ou igual

exp1 < exp2 :

retorna 1 quando exp1 é menor que exp2 ou 0, caso contrário:

3 < 5 : retorna 1

5 < 3 : retorna 0

3 < 3 : retorna 0

exp1 <= exp2 :

retorna 1 quando exp1 é menor ou igual que exp2 ou 0, caso contrário:

3 <= 5 : retorna 1

5 <= 3 : retorna 0

3 <= 3 : retorna 1

Expressões lógicas

- São expressões que realizam uma operação lógica (e, ou, negação) entre duas expressões e retornam:
 - 0, se o resultado é FALSO;
 - 1 (ou qualquer número diferente de zero), se o resultado é VERDADEIRO.

Operadores lógicos

Os operadores lógicos, em C, são:

- && : operador E
- | | : operador OU
- ! : operador de NEGAÇÃO

Expressões Lógicas: Operador E

exp1 && exp2

retorna verdadeiro quando exp1 E exp2 são verdadeiras

| exp1 | exp2 | Resultado |
|----------------|----------------|----------------|
| Verdadeiro (1) | Verdadeiro (1) | Verdadeiro (1) |
| Verdadeiro (1) | Falso (0) | Falso (0) |
| Falso (0) | Verdadeiro (1) | Falso (0) |
| Falso (0) | Falso (0) | Falso (0) |

Expressões Lógicas: Operador OU

exp1 || exp2

retorna Verdadeiro (1) quando exp1 OU exp2 é verdadeira

| exp1 | exp2 | Resultado |
|----------------|----------------|----------------|
| Verdadeiro (1) | Verdadeiro (1) | Verdadeiro (1) |
| Verdadeiro (1) | Falso (0) | Verdadeiro (1) |
| Falso (0) | Verdadeiro (1) | Verdadeiro (1) |
| Falso (0) | Falso (0) | Falso (0) |

Expressões Lógicas: Operador de NEGAÇÃO

! exp1

retorna Verdadeiro (1) quando exp1 é falsa

retorna Falso (0) quando exp1 é verdadeira

| exp1 | Resultado |
|----------------|----------------|
| Verdadeiro (1) | Falso (0) |
| Falso (0) | Verdadeiro (1) |

Equivalência entre expressões lógicas

- $\neg (a == b)$ equivale a $(a != b)$
- $\neg (a != b)$ equivale a $(a == b)$
- $\neg (a > b)$ equivale a $(a <= b)$
- $\neg (a >= b)$ equivale a $(a < b)$
- $\neg (a < b)$ equivale a $(a >= b)$
- $\neg (a <= b)$ equivale a $(a > b)$

Bloco de Comandos

- Um Bloco de comandos é um conjunto de comandos delimitados por { e } e define o escopo das variáveis.
- Escopo de variável é a região do programa em que a variável pode ser acessada.

```
{
```

```
    int x;
```

```
    x = 1;
```

```
    x = x + 10;
```

```
}
```

```
int x; // Este x ocupa uma memória diferente do x anterior!
```

```
x = 5;
```

Comandos Condicionais

- Permitem alterar o fluxo de execução, escolhendo se um bloco de comandos deve ou não ser executado, com base em uma expressão lógica ou relacional
- Em C, existem os seguintes comandos condicionais:
 - if
 - if-else
 - switch

Comandos Condicionais: if

Sintaxe:

```
if ( <expressão lógica ou relacional>
    comando_único;
```

Ou:

```
if ( <expressão lógica ou relacional> ) {
    comandos;
    outros_comandos;
}
```

- O comando_único ou o bloco de comandos é executado quando <expressão lógica ou relacional> é verdadeira.

Comandos Condicionais: if-else

- Variação do comando if que inclui um bloco a ser executado quando a expressão relacional for falsa.

Sintaxe:

```
if ( <expressão lógica ou relacional> ) {  
    comandos_para_expressão_verdadeira;  
} else {  
    comandos_para_expressão_falsa;  
}
```

Comandos Condicionais: if dentro de outro if

- Um comando if pode aparecer no bloco de comandos de outro comando if.

```
if (<exp1>){  
    if(<exp2>){  
        comando2;  
    }  
}
```

Equivale a:

```
if (<exp1> && <exp2>){  
    comando2;  
}
```

- comando2 é executado se <exp1> e <exp2> são verdadeiras, ou seja, se <exp1> && <exp2> é verdadeira.

Comandos Condicionais: importância dos blocos

```
int x = 10;
```

```
if ( x%2 == 0 )
```

```
    if ( x > 5 )
```

```
        printf("Sou par e maior que 5!");
```

```
else
```

```
    printf("E agora? Sou ímpar? Sou par e menor igual a 5?");
```

O else é executado quando qual expressão for falsa?

Comandos Condicionais: importância dos blocos

```
int x = 10;
```

```
if ( x%2 == 0 )
```

```
    if ( x > 5 )
```

```
        printf("Sou par e maior que 5!");
```

```
    else
```

```
        printf("Sou par e menor igual a 5");
```

O else executa quando o if mais próximo for falso.

Comandos Condicionais: importância dos blocos

```
int x = 10;

if ( x%2 == 0 ){
    if ( x > 5 ){
        printf("Sou par e maior que 5!");
    } else {
        printf("Ufa! Sou par e menor ou igual a 5!");
    }
}
```

É melhor utilizar { e } para deixar claro o que se quer!

Comandos Condicionais: if's Sequenciais

- Quando queremos escolher uma dentre várias alternativas, podemos usar uma sequência de if's:

```
if (a == 1) {comando1;}
```

```
if (a == 2) {comando2;}
```

```
if (a == 3) {comando3;}
```

```
if (a != 1 && a != 2 && a != 3) {comando_falso;}
```

- Problema? Se a for 2, o terceiro teste também será executado!

Comandos Condicionais: if's Encaixados

- A solução é utilizar if's encaixados:

```
if (a == 1) {  
    comando1;  
} else if (a == 2) {  
    comando2;  
} else if (a == 3) {  
    comando3;  
} else {  
    comando_falso;  
}
```


Comandos Condicionais:

Funcionamento dos if's Encaixados

- Os testes são feitos até que um deles seja verdadeiro.
- O bloco de comandos do teste verdadeiro será executado.
- Os testes que vierem depois do primeiro teste verdadeiro não serão executados.
- O último else pode ser utilizado para um bloco a ser executado no caso de nenhum teste ser verdadeiro.

Exemplo de if's Sequenciais

```
if (dia == 1)
```

```
    printf("Domingo.\n");
```

```
if (dia == 2)
```

```
    printf("Segunda-feira.\n");
```

```
if (dia == 3)
```

```
    printf("Terça-feira.\n");
```

```
if (dia == 4)
```

```
    printf("Quarta-feira.\n");
```

```
if (dia == 5)
```

```
    printf("Quinta-feira.\n");
```

```
if (dia == 6)
```

```
    printf("Sexta-feira.\n");
```

```
if (dia == 7)
```

```
    printf("Sábado.\n");
```

```
if ( !(dia == 1 || dia == 2 || dia == 3  
    || dia == 4 || dia == 5 || dia == 6 ||  
    dia == 7) )
```

```
    printf("Dia inválido.\n");
```

Exemplo de if's Encaixados

if (dia == 1)

 printf("Domingo.\n");

else if (dia == 2)

 printf("Segunda-feira.\n");

else if (dia == 3)

 printf("Terça-feira.\n");

else if (dia == 4)

 printf("Quarta-feira.\n");

else if (dia == 5)

 printf("Quinta-feira.\n");

else if (dia == 6)

 printf("Sexta-feira.\n");

else if (dia == 7)

 printf("Sábado.\n");

else

 printf("Dia da semana inválido.\n");

O Comando switch

O comando switch substitui o uso de if's encaixados quando o teste é feito em uma variável dos tipos **int** ou **char**:

```
switch ( <variável int ou char>){  
    case val1: comandos_para_variável_igual_a_val1;  
        break;  
    case val2: comandos_para_variável_igual_a_val2;  
        break;  
    default: comandos_em_caso_de_falha_dos_testes;  
}
```

O Comando switch: Exemplo de variável int

```
int dia;  
switch(dia){  
    case 1:  
        printf("Primeiro dia."); break;  
  
    case 30:  
        printf("Último dia."); break;  
  
    default:  
        printf("Um dia qualquer.");  
  
}
```

O Comando switch: Exemplo de variável char

```
char letra;  
switch(letra){  
    case 'a':  
        printf("Primeira letra minúscula."); break;  
    case 'z':  
        printf("Última letra minúscula."); break;  
    default:  
        printf("Uma letra qualquer.");  
}
```

Referências Bibliográficas

- Material de aula do Prof. Ricardo Anido, da UNICAMP:
<http://www.ic.unicamp.br/~ranido/mc102/>
- Material de aula da Profa. Virgínia F. Mota:
<https://sites.google.com/site/viriniaferm/home/disciplinas>
- DEITEL, P; DEITEL, H. *C How to Program*. 6a Ed. Pearson, 2010.