

MC102 – Algoritmos e Programação de Computadores

Instituto de Computação

UNICAMP

Primeiro Semestre de 2014

Roteiro

- 1 Expressões relacionais
- 2 Expressões lógicas
- 3 Comandos condicionais
- 4 Comando `switch`

Expressões aritméticas

- Já vimos que constantes e variáveis são expressões.

Exemplos:

```
a = 10;  
a = b;
```

- Vimos ainda que operações aritméticas também são expressões.

Exemplos:

```
a = 2 + 2;  
a = 10 / (float) 3;  
a = a + 1;
```

Expressões relacionais

Expressões relacionais são aquelas que realizam uma comparação entre duas expressões e retornam:

- 1 Zero (0), se o resultado é falso.
- 2 Um (1) ou qualquer outro número diferente de zero, se o resultado é verdadeiro.

Operadores relacionais

Os operadores relacionais são:

- $==$: igualdade.
- $!=$: diferente.
- $>$: maior que.
- $<$: menor que.
- $>=$: maior ou igual que.
- $<=$: menor ou igual que.

Expressões relacionais

- $\langle \text{expressão} \rangle == \langle \text{expressão} \rangle$: retorna verdadeiro quando as expressões forem iguais.

Ex.: $a == b$

- $\langle \text{expressão} \rangle != \langle \text{expressão} \rangle$: retorna verdadeiro quando as expressões forem diferentes.

Ex.: $a != b$

Expressões relacionais

- $\langle \text{expressão} \rangle > \langle \text{expressão} \rangle$: retorna verdadeiro quando a expressão da esquerda tiver valor maior que a expressão da direita.
Ex.: $a > b$
- $\langle \text{expressão} \rangle < \langle \text{expressão} \rangle$: retorna verdadeiro quando a expressão da esquerda tiver valor menor que a expressão da direita.
Ex.: $a < b$

Expressões relacionais

- $\langle \text{expressão} \rangle \geq \langle \text{expressão} \rangle$: retorna verdadeiro quando a expressão da esquerda tiver valor maior ou igual que a expressão da direita.
Ex.: $a \geq b$
- $\langle \text{expressão} \rangle \leq \langle \text{expressão} \rangle$: retorna verdadeiro quando a expressão da esquerda tiver valor menor ou igual que a expressão da direita.
Ex.: $a \leq b$

Expressões lógicas

São aquelas que realizam uma operação lógica (OU, E, NÃO, etc...) e retornam verdadeiro ou falso (como as expressões relacionais).

Operadores lógicos

- `&&`: operador E.
- `||`: operador OU.
- `!`: operador NÃO.

Expressões lógicas

- `<expressão1> && <expressão2>`: retorna verdadeiro quando ambas as expressões são verdadeiras. Sua tabela verdade é:

<code><expressão1></code>	<code><expressão2></code>	resultado
V	V	V
V	F	F
F	V	F
F	F	F

Exemplos:

```
(a == 0) && (b == 0)
```

```
(x >= y) && (y >= z) && (x != z)
```

Expressões lógicas

- `<expressão1> || <expressão2>`: retorna verdadeiro quando pelo menos uma das expressões é verdadeira. Sua tabela verdade é:

<code><expressão1></code>	<code><expressão2></code>	resultado
V	V	V
V	F	V
F	V	V
F	F	F

Exemplo:

`(a == 0) || (b == 0)`

`(x == y) || (y == z) || (x == z)`

Expressões lógicas

- ! <expressão>: retorna verdadeiro quando a expressão é falsa e vice-versa. Sua tabela verdade é:

<expressão>	resultado
V	F
F	V

Exemplo:

!(a == 0)

Simplificações úteis

- $!(a == b)$ é equivalente a: $(a != b)$
- $!(a != b)$ é equivalente a: $(a == b)$
- $!(a > b)$ é equivalente a: $(a <= b)$
- $!(a < b)$ é equivalente a: $(a >= b)$
- $!(a >= b)$ é equivalente a: $(a < b)$
- $!(a <= b)$ é equivalente a: $(a > b)$

Comandos condicionais

Um comando condicional é aquele que permite decidir se um determinado bloco de comandos deve ou não ser executado, a partir do resultado de uma expressão relacional ou lógica.



Comandos condicionais

- O principal comando condicional da linguagem C é o `if`:

```
if (expressão lógica)
    comando;
```

ou:

```
if (expressão lógica) {
    comando;
    ...
    comando;
}
```

- Os comandos são executados somente se a expressão lógica for verdadeira.

Bloco de comandos

- É um conjunto de instruções agrupadas.
- Limitado pelos caracteres { e }.

Exemplo:

```
int main() {           ← início do bloco de comandos
    int a;
    a = 1;
    ...
}                       ← fim do bloco de comandos
```

Comandos condicionais

O programa a seguir verifica se um valor é ímpar.

```
#include <stdio.h>

int main() {
    int a;

    printf("Entre com um numero inteiro: ");
    scanf("%d", &a);

    if ((a % 2) != 0) {
        printf("Numero impar\n");
    }

    return 0;
}
```

Comandos condicionais

Lembrando como C representa os valores Falso e Verdadeiro, o programa pode ser alterado da seguinte forma:

```
#include <stdio.h>

int main() {
    int a;

    printf("Entre com um numero inteiro: ");
    scanf("%d", &a);

    if (a % 2) {
        printf("Numero impar\n");
    }

    return 0;
}
```

Comandos condicionais

Neste caso, como apenas um comando deve ser executado pelo comando condicional, podemos omitir as chaves.

```
#include <stdio.h>

int main() {
    int a;

    printf("Entre com um numero inteiro: ");
    scanf("%d", &a);

    if (a % 2)
        printf("Numero impar\n");

    return 0;
}
```

Comandos condicionais

- Uma variação do comando if é o if/else, cuja sintaxe é:

```
if (expressão lógica) {  
    comandos executados se a expressão é verdadeira  
} else {  
    comandos executados se a expressão é falsa  
}
```

Comandos condicionais

O programa a seguir verifica se um valor é par ou ímpar.

```
#include <stdio.h>
```

```
int main() {
```

```
    int a;
```

```
    printf("Entre com um numero inteiro: ");
```

```
    scanf("%d", &a);
```

```
    if (a % 2)
```

```
        printf("Numero impar\n");
```

```
    else
```

```
        printf("Numero par\n");
```

```
    return 0;
```

```
}
```

Comandos condicionais

O programa a seguir determina o menor entre dois números:

```
#include <stdio.h>

int main() {
    int a, b;

    printf("Entre com dois numeros inteiros: ");
    scanf("%d %d", &a, &b);

    if (a < b)
        printf("O menor numero eh: %d\n", a);
    else
        printf("O menor numero eh: %d\n", b);

    return 0;
}
```

Comandos condicionais

O programa a seguir compara dois números inteiros.

```
#include <stdio.h>

int main() {
    int a, b;

    printf("Entre com dois numeros inteiros: ");
    scanf("%d %d", &a, &b);

    if (a == b)
        printf("Os dois numeros sao iguais.\n");
    else {
        if (a > b) {
            printf("O primeiro numero eh o maior.\n");
        } else {
            printf("O segundo numero eh o maior.\n");
        }
    }
    return 0;
}
```


Comandos condicionais

```
if (condição1)
  if (condição2)
    comando1;
else
  comando2;
```

Quando o comando2 é executado?

Comandos condicionais

```
if (condição1)
  if (condição2)
    comando1;
  else
    comando2;
```

Quando o comando2 é executado?

Comandos condicionais

```
if (condição1) {  
    if (condição2)  
        comando1;  
} else  
    comando2;
```

Quando o comando2 é executado?

if-else-if aninhados

Algo muito comum em programação é o teste de várias alternativas. Neste caso, pode-se usar uma construção simples com `if`'s:

```
#include <stdio.h>

int main() {
    int ra;
    scanf("%d", &ra);
    if (ra == 95584)
        printf("Andre Rodrigues Oliveira\n");
    if (ra == 134042)
        printf("Carla Negri Lintzmayer\n");
    if (ra == 109230)
        printf("Filipe de Oliveira Costa\n");
    if ...
    ...
    return 0;
}
```

if-else-if aninhados

Porém, todos os testes condicionais serão executados. Quando apenas uma de várias alternativas é verdadeira, pode-se usar a construção if-else-if:

```
#include <stdio.h>

int main() {
    int ra;
    scanf("%d", &ra);
    if (ra == 95584)
        printf("Andre Rodrigues Oliveira\n");
    else if (ra == 134042)
        printf("Carla Negri Lintzmayer\n");
    else if (ra == 109230)
        printf("Filipe de Oliveira Costa\n");
    else if ...
        ...
    else
        printf("RA nao encontrado!\n");
    return 0;
}
```

if-else-if aninhados

O programa a seguir compara dois números inteiros.

```
#include <stdio.h>

int main() {
    int a, b;

    printf("Entre com dois numeros inteiros: ");
    scanf("%d %d", &a, &b);

    if (a == b) {
        printf("Os dois numeros sao iguais.\n");
    } else if (a > b) {
        printf("O primeiro numero eh o maior.\n");
    } else {
        printf("O segundo numero eh o maior.\n");
    }

    return 0;
}
```

if-else-if aninhados

- Na construção `if-else-if`, quando uma condição é verdadeira, o bloco de comandos correspondente será executado.
- Após a execução do bloco de comandos, as outras alternativas não serão testadas.
- O último `else` (sem `if`) pode ser utilizado como uma opção padrão quando nenhuma das condições dos `if`'s é verdadeira.

Comando switch

- O objetivo do comando `switch` é simplificar uma expressão onde as condições ocorrem sobre uma variável inteira ou caractere:

Sintaxe:

```
switch (variavel inteira) {  
    case valor1:  
        comandos;  
        break;  
    case valor2:  
        comandos;  
        break;  
    case valor3:  
        comandos;  
        break;  
}
```


Comando switch

```
switch (ra) {  
    case 95584:  
        printf("Andre Rodrigues Oliveira\n");  
        break;  
    case 134042:  
        printf("Carla Negri Lintzmayer\n");  
        break;  
    case 109230:  
        printf("Filipe de Oliveira Costa\n");  
        break;  
}
```

Comando switch

- Os comandos começam a ser executados a partir do ponto onde o valor da variável corresponde ao valor antes dos dois pontos (:).
- Executa todos os comandos até que encontre um comando `break` ou que chegue ao final do bloco de comandos do `switch`.

Valor padrão

- Pode-se utilizar uma condição default. A execução dentro da alternativa default ocorre se nenhuma outra condição foi verdadeira (assim como o último else do if-else-if aninhados).

Sintaxe:

```
switch (variavel inteira) {  
    case valor1:  
        comandos;  
        break;  
    case valor2:  
        comandos;  
        break;  
    ...  
    default:  
        comandos;  
}
```

Valor padrão

```
switch (ra) {  
    case 95584:  
        printf("Andre Rodrigues Oliveira\n");  
        break;  
    case 134042:  
        printf("Carla Negri Lintzmayer\n");  
        break;  
    case 109230:  
        printf("Filipe de Oliveira Costa\n");  
        break;  
    default:  
        printf("RA nao encontrado\n");  
}
```

Exemplo - Programas de uma máquina de lavar e secar

```
switch (programa) {  
    case 1:  
        printf("Lavar\n");  
    case 2:  
        printf("Enxaguar\n");  
    case 3:  
        printf("Centrifugar\n");  
    case 4:  
        printf("Secar\n");  
        break;  
    default:  
        printf("Programa invalido\n");  
}
```

Exercícios

- Escreva um programa que lê três números inteiros e imprime o menor deles.
- Escreva um programa que lê três números inteiros e os imprime em ordem crescente.
- Escreva um programa que, dadas duas datas, determine qual delas ocorreu cronologicamente primeiro. Cada data é composta de 3 números inteiros, um representando o dia, outro o mês e outro o ano.
- Escreva um programa que, dados os comprimentos de três segmentos de reta, determine se eles podem formar um triângulo e, em caso positivo, imprima se o triângulo é equilátero, isósceles ou escaleno.
- Altere o programa anterior para determinar se os três segmentos de reta dados podem formar um triângulo retângulo.

Exercício - Determinar o menor entre três números

```
#include <stdio.h>

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

    printf("Digite tres numeros: ");
    scanf("%d %d %d", &a, &b, &c);

    if ((a <= b) && (a <= c))
        printf("Menor: %d\n", a);
    else if ((b <= a) && (b <= c))
        printf("Menor: %d\n", b);
    else
        printf("Menor: %d\n", c);

    return 0;
}
```

Exercício - Ordenar três números

```
#include <stdio.h>

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

    printf("Digite tres numeros: ");
    scanf("%d %d %d", &a, &b, &c);

    if ((a <= b) && (a <= c)) { /* Menor numero: a */

        if (b <= c)
            printf("Ordem: %d, %d, %d\n", a, b, c);
        else
            printf("Ordem: %d, %d, %d\n", a, c, b);

    } ...
}
```


Exercício - Ordenar três números

```
... else if ((b <= a) && (b <= c)) { /* Menor numero: b */  
  
    if (a <= c)  
        printf("Ordem: %d, %d, %d\n", b, a, c);  
    else  
        printf("Ordem: %d, %d, %d\n", b, c, a);  
  
} else { /* Menor numero: c */  
  
    if (a <= b)  
        printf("Ordem: %d, %d, %d\n", c, a, b);  
    else  
        printf("Ordem: %d, %d, %d\n", c, b, a);  
  
}  
  
return 0;  
}
```