

MC-102 — Aula 07

Expressões Relacionais, Lógicas e Comandos Condicionais

Instituto de Computação – Unicamp

22 de Março de 2013

Roteiro

- 1 Expressões relacionais
- 2 Expressões lógicas
- 3 Comandos condicionais
- 4 Exercícios
- 5 Informações Extras: O comando `switch`

Revisão

- Tipos de variáveis (int, float, double, char).
- Modificadores (short, unsigned, long).
- char → 1 caractere – usa aspas simples '+', '/', 'a', 'A' ... (tabela ASCII).
- Operadores +, -, *, /, % e suas precedências
- Divisão inteira.
- printf, scanf
- ++, --

Revisão

- Já vimos que constantes e variáveis são expressões.

Exemplo

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

- Vimos também que operações aritméticas também são expressões.

Exemplo

```
a = 2 + 2;  
a = 10 / 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 da linguagem C são:

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

Expressões relacionais

- $\langle \textit{expressao} \rangle == \langle \textit{expressao} \rangle$: Retorna verdadeiro quando as expressões forem iguais.

Ex: $a == b$

- $\langle \textit{expressao} \rangle != \langle \textit{expressao} \rangle$: Retorna verdadeiro quando as expressões forem diferentes.

Ex: $a != b$

Expressões relacionais

- $< \textit{expressao} > > < \textit{expressao} >$: Retorna verdadeiro quando a expressão da esquerda tiver valor maior que a expressão da direita.
Ex: $a > b$
- $< \textit{expressao} > < < \textit{expressao} >$: Retorna verdadeiro quando a expressão da esquerda tiver valor menor que a expressão da direita.
Ex: $a < b$

Expressões relacionais

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

O que será impresso pelo programa?

```
#include <stdio.h>

int main(){
    int a, b;

    printf("%d\n", 9 > 3);

    printf("%d\n", (3*4)/2 != (2*3) );

    a = 1;
    b = -1;
    printf("%d\n", a!=b);
}
```

Expressões lógicas

Expressões lógicas são aquelas que realizam uma operação lógica (**ou**, **e**, **não**, etc...) e retornam verdadeiro ou falso (como as expressões relacionais).

Operadores Lógicos

Na linguagem C temos os seguintes operadores lógicos:

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

Expressões lógicas

- $< \text{expressao} > \&\& < \text{expressao} >$: Retorna verdadeiro quando ambas as expressões são verdadeiras. Sua tabela verdade é:

Op_1	Op_2	Ret
V	V	V
V	F	F
F	V	F
F	F	F

Qual o resultado da expressão lógica abaixo?

a = 0;

b = 0;

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

Expressões lógicas

- $\langle \text{expressao} \rangle \ || \ \langle \text{expressao} \rangle$: Retorna verdadeiro quando pelo menos uma das expressões é verdadeira. Sua tabela verdade é:

Op_1	Op_2	Ret
V	V	V
V	F	V
F	V	V
F	F	F

Qual o resultado da expressão lógica abaixo?

a = 0;

b = 1;

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

Expressões lógicas

- $! < \textit{expressao} >$: Retorna verdadeiro quando a expressão é falsa e vice-versa. Sua tabela verdade é:

Op_1	Ret
V	F
F	V

Qual o resultado da expressão lógica abaixo?

$a = 0;$

$b = 1;$

$!(a != b)$

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.



Bloco de comandos

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

Exemplo

```
main(void)
{           ← Início do bloco de comandos
    int a;
    a=1;
}           ← Fim do bloco de comandos
```

Comandos condicionais

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

```
if (expressão relacional ou lógica)
    comando; ou
if (expressão relacional ou lógica) {
    comandos
}
```

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

Comandos condicionais

O programa abaixo determina se um valor é ímpar.

```
#include <stdio.h>

int main () {
    int a;
    scanf("%d", &a);
    if ((a % 2) != 0) {
        printf ("O valor é ímpar.\n");
    }
}
```

Comandos condicionais

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

```
#include <stdio.h>

int main () {
    int a;
    scanf("%d", &a);
    if (a % 2) {
        printf ("O valor é ímpar.\n");
    }
}
```

Comandos condicionais

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

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

Comandos condicionais

Exemplo: Determinando o menor de dois números:

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

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

    if(a < b){
        printf("O menor numero e: %d\n", a);
    }else{
        printf("O menor numero e:%d\n",b);
    }
}
```

Comandos condicionais

- Note que o **if** é um comando, e como tal pode aparecer dentro do bloco de comandos de outro **if**.

Exemplo: Usando apenas operadores relacionais, vamos escrever um programa que lê um número e verifica se este é:

- Par e menor que 100.
- Par e maior ou igual a 100.
- Ímpar e menor que 100.
- Ímpar e maior ou igual a 100.

Comandos condicionais

```
#include <stdio.h>

int main(){
    int a;
    printf("Digite um número inteiro:");
    scanf("%d", &a);
    if( a % 2 == 0){
        if( a < 100)
            printf("O número é par e menor que 100\n");
        else
            printf("O número é par e maior ou igual a 100\n");
    }else{
        if ( a < 100)
            printf("O número é ímpar e menor que 100\n");
        else
            printf("O número é ímpar e maior que 100\n");
    }
}
```

Se você pudesse usar operadores lógicos, como você poderia refazer este programa?

Comandos condicionais

```
#include <stdio.h>

int main(){
    int a;
    printf("Digite um número inteiro:");
    scanf("%d", &a);
    if( ( a % 2 == 0) && (a<100) )
        printf("O número é par e menor que 100\n");
    if( ( a % 2 == 0) && (a>=100) )
        printf("O número é par e maior ou igual a 100\n");
    if( ( a % 2 != 0) && (a<100) )
        printf("O número é ímpar e menor que 100\n");
    if( ( a % 2 != 0) && (a>=100) )
        printf("O número é ímpar e maior que 100\n");
}
```

Comandos condicionais

```
if (cond1)
    if (cond2)
        comando1;
else
    comando2;
```

Quando o comando2 é executado?

Comandos condicionais

```
if (cond1)
  if (cond2)
    comando1;
  else
    comando2;
```

Quando o comando2 é executado?

Comandos condicionais

```
if (cond1) {  
    if (cond2)  
        comando1;  
} else  
    comando2;
```

Quando o comando2 é executado?

Use chaves e indentação para deixar claro a qual bloco de comando(s) pertence cada comando!

if-else-if Encaixados

- Uma coisa muito comum em programação é o teste de várias alternativas.
- Podemos usar uma construção simples com ifs:

```
int main () {  
    int ra;  
    scanf("%d", &ra);  
    if (ra == 10129)  
        printf("Maria Cândida Moreira Telles\n");  
    if (ra == 33860)  
        printf("Larissa Garcia Alfonsi\n");  
    if (ra == 33967)  
        printf("Leonardo Kozlowiski Kenupp\n");  
    if(...  
    ....  
}
```

if-else-if Encaixados

- Porém todos os testes condicionais serão executados!!
- Quando apenas uma de várias alternativas é verdadeira podemos usar a construção if-else-if:

```
int main () {  
    int a;  
    scanf("%d", &a);  
    if (a == 10129)  
        printf("Maria Cândida Moreira Telles\n");  
    else if (a == 33860)  
        printf("Larissa Garcia Alfonsi\n");  
    else if (a == 33967)  
        printf("Leonardo Kozlowiski Kenupp\n");  
    else if (....  
        ....  
    else  
        printf("Nenhum aluno com RA informado!");  
}
```

if-else-if Encaixados

- 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 ifs é verdadeira.

Exercícios

A solução abaixo está correta para classificar um número como par e menor que 100, ou par e maior ou igual a 100, etc, como no exemplo visto anteriormente?

```
#include <stdio.h>

int main(){
    int a;
    printf("Digite um número inteiro:");
    scanf("%d", &a);
    if( ( a % 2 == 0) && (a<100) )
        printf("O número é par e menor que 100\n");
    else if( a>=100 )
        printf("O número é par e maior ou igual a 100\n");

    if( ( a % 2 != 0) && (a<100) )
        printf("O número é ímpar e menor que 100\n");
    else if (a>=100)
        printf("O número é ímpar e maior que 100\n");
}
```

Exercícios

- Escreva um programa que lê um número inteiro do teclado e imprime "SIM" se o número for par e maior do que 10, ou for ímpar e menor do que 50. Caso contrário o programa deve imprimir "NAO".

Exercícios

- Escreva um programa lê três números e imprime o maior deles.

Exercícios

- Escreva um programa lê três números e os imprime em ordem (ordem crescente).

Informações Extras: O comando switch

- O objetivo do comando `switch` é simplificar uma construção `if-else-if` encaixados quando as condições ocorrem sobre uma variável **inteira** ou **caracter**:

Sintaxe

```
switch (variável inteira) {  
    case valor:  comandos  
    break;  
    case valor:  comandos  
    break;  
}
```

Informações Extras: O comando switch

```
printf("Digite o RA: ");
scanf("%d", &a);
switch(a) {
    case 10129:
        printf("Maria Cândida Moreira Telles\n");
        break;
    case 33860:
        printf("Larissa Garcia Alfonsi\n");
        break;
    case 33967:
        printf("Leonardo Kozlowiski Kenupp\n");
        break;
}
```

Informações Extras: O 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`

Informações Extras. Valor padrão

- Você pode 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 encaixados).

Sintaxe

```
switch (variável inteira) {  
    valor:  comandos break;  
    default: comandos  
}
```


Informações Extras. Valor padrão

```
printf("Digite o RA: ");
scanf("%d", &a);
switch(a) {
case 10129:
    printf("Maria Cândida Moreira Telles\n");
    break;
case 33860:
    printf("Larissa Garcia Alfonsi\n");
    break;
default:
    printf("O aluno não está matriculado\n");
}
```