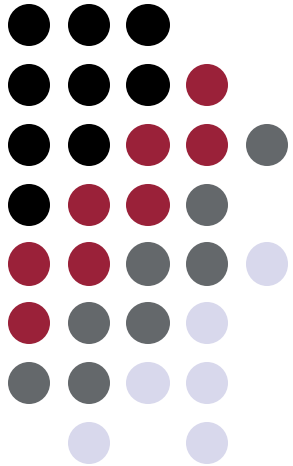
A04 Expressões e Comandos Condicionais

CSI030 – Programação de Computadores I

Prof. Dr. George H. G. Fonseca
Universidade Federal de Ouro Preto



Agenda

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

Expressões

- Expressões aritméticas: cálculos envolvendo variáveis e constantes

- Expressões relacionais: realizam 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 valor ≠ 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

Operadores de Igualdade

- exp1 == exp2

  Retorna 1 quando as expressões são iguais e 0 caso contrário:

  2 == 2 retorna 1
  3 == 4 retorna 0

- exp1 != exp2

  Retorna 1 quando as expressões são diferentes e 0 caso contrário:

  2 != 2 retorna 0
  3 != 4 retorna 1

Operadores Relacionais

- exp1 > exp2

  Retorna 1 quando exp1 é maior que exp2 e 0 caso contrário:

3 > 2	retorna 1
3 > 3	retorna 0
3 > 4	retorna 0

- exp1 >= exp2

  Retorna 1 quando exp1 é maior ou igual que exp2 e 0 caso contrário:

3 >= 2	retorna 1
3 >= 3	retorna 1
3 >= 4	retorna 0

Operadores Relacionais

- exp1 < exp2

  Retorna 1 quando exp1 é menor que exp2 e 0 caso contrário:

3 < 2	retorna 0
3 < 3	retorna 0
3 < 4	retorna 1

- exp1 <= exp2

  Retorna 1 quando exp1 é menor ou igual que exp2 e 0 caso contrário:

3 <= 2	retorna 0
3 <= 3	retorna 1
3 <= 4	retorna 1

Exercícios

- Supondo as seguinte variáveis:

```
int A = 3;
int B = 7;
int C = 4;
```

calcule o valor das expressões :

```
(A + C) > B
B >= (A + 2)
C == (B - A)
(B + A) <= C
(C + A) > B
(A + B) < 10
```

Exercícios

- Supondo as seguinte variáveis:

```
int A = 3;
int B = 7;
int C = 4;
```

calcule o valor das expressões :

0	(A + C) > B
1	B >= (A + 2)
1	C == (B - A)
0	(B + A) <= C
0	(C + A) > B
0	(A + B) < 10

Expressões Lógicas

- São expressões que realizam uma operação lógica (E, OU, NEGAÇÃO) entre duas expressões e retornam:
  - 0, se o resultado é falso
  - 1, se o resultado é verdadeiro

Operadores Lógicos

- Os operadores lógicos em C são:
  - && operador E
  - || operador OU
  - ! operador negação

Operador &&

- exp1 && exp2

  Retorna verdadeiro se exp1 E exp2 são verdadeiras

exp1	exp2	exp1 && exp2
Verdadeiro (1)	Verdadeiro (1)	Verdadeiro (1)
Verdadeiro (1)	Falso (0)	Falso (0)
Falso (0)	Verdadeiro (1)	Falso (0)
Falso (0)	Falso (0)	Falso (0)

Operador ||

- exp1 || exp2

  Retorna verdadeiro se exp1 OU exp2 é verdadeiras

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

Operador !

- ! exp1

  Retorna verdadeiro se exp1 é falsa
  Retorna falso se exp1 é verdadeira

exp1	! exp1
Verdadeiro (1)	Falso (0)
Falso (0)	Verdadeiro (1)

Equivalência entre Expressões Lógicas

- ! (a == b) equivale a (a != b)
- ! (a == b) equivale a (a != b)
- ! (a != b) equivale a (a == b)
- ! (a > b) equivale a (a <= b)
- ! (a >= b) equivale a (a < b)
- ! (a < b) equivale a (a >= b)
- ! (a <= b) equivale a (a > b)

Exercícios

- Supondo as seguinte variáveis:

```
int A = 5;
int B = 4;
int C = 3;
int D = 6;
```

calcule o valor das expressões :

```
(A > C) && (C <= D)
(A + B) > 10 || (A + B) == (C + D)
(A >= C) && (D >= C)
```

Exercícios

- Supondo as seguinte variáveis:

```
int A = 5;
int B = 4;
int C = 3;
int D = 6;
```

calcule o valor das expressões :

1	`(A > C) && (C <= D)`
1	`(A + B) > 10 || (A + B) == (C + D)`
1	`(A >= C) && (D >= C)`

Exercícios

- Supondo as seguinte variáveis:

```
int A = 5;
int B = 4;
int C = 3;
char C1 = 'A';
char C2 = 'a';
int L = 0;
```

calcule o valor das expressões :

```
B == A * C && L
C1 == C2 || 'F' != 'Q'
A * C / B > A * B * C
! L
```

Exercícios

- Supondo as seguinte variáveis:

```
int A = 5;
int B = 4;
int C = 3;
char C1 = 'A';
char C2 = 'a';
int L = 0;
```

calcule o valor das expressões :

0	`B == A * C && L`
1	`C1 == C2 || 'F' != 'Q'`
0	`A * C / B > A * B * C`
1	`! L`

Exercícios

- Supondo as seguinte variáveis:

```
int A = 3;
int B = 5;
int C = 8;
int D = 7;
int X = 1;
```

calcule o valor das expressões :

```
!( X > 3)
(X < 1) && (!(B > D))
!(D < 0) && (C > 5)
!((X > 3) || (C < 7))
(A > B) || (C > B)
(X < 1) && (B >= D)
(D < 0) || (C > 5)
!( D > 3) || !( B < 7)
```

Exercícios

- Supondo as seguinte variáveis:

```
int A = 3;
int B = 5;
int C = 8;
int D = 7;
int X = 1;
```

calcule o valor das expressões :

1	`!( X > 3)`
0	`(X < 1) && (!(B > D))`
1	`!(D < 0) && (C > 5)`
1	`!((X > 3) || (C < 7))`
1	`(A > B) || (C > B)`
0	`(X < 1) && (B >= D)`
1	`(D < 0) || (C > 5)`
0	`!( D > 3) || !( B < 7)`

Bloco de Comandos

- Um bloco de comandos é um conjunto de comandos delimitados por { e } e define o **escopo** das variáveis

- Escopo de uma variável é a região do programa em que a variável pode ser acessada

```
{
      int x;
      x = 1;
      x = x + 10;
}
x = x * 2;
```

Variável x pode ser usada

ERRO! Variável x não está disponível mais

Comandos Condicionais

- Premitem alterar o fluxo de execução, escolhendo se um bloco de comandos deve ser executado ou não, com base em uma expressão

- Em C, existem os seguintes comandos condicionais
  - **if**
  - **else**
  - **switch**

Comandos Condicionais: if

- Sintaxe:

```
if(expressao)
{
     comando1;
     comando2;
     ...
     comandoN;
}
```

ou

```
if(expressao)
     comando_unico;
```

Apesar de ambas formas serem aceitas, é recomendável o uso de { } para delimitar a instruções a serem executadas

Comandos Condicionais: if-else

- Variação do comando if que inclui um bloco a ser executado quando a expressão é falsa

- Sintaxe:

```
if(expressao)
{
    comandos_para_expressao_verdadeira;
}
else
{
    comandos_para_expressao_falsa;
}
```

Comandos Condicionais: if-else

- Um comando if aparecer no bloco de comandos de outro comando if:

```
if(exp1)
{
     if(exp2)
     {
          comando;
     }
}
```

Exercício

```
 1  #include <stdio.h>
 2
 3  int main()
 4  {
 5      int x = 5, y = 7;
 6      if(x > 4)
 7      {
 8          printf("1\n");
 9          if(x > 6)
10              printf("2\n");
11          if(y > 9)
12              printf("3\n");
13          else
14              printf("4\n");
15      }
16      else
17          printf("5\n");
18      return 0;
19  }
```

O quê o programa ao lado imprime?

Comandos Condicionais: if-else

```
 1  #include <stdio.h>
 2
 3  int main()
 4  {
 5      int x = 5, y = 7;
 6      if(x > 4)
 7      {
 8          printf("1\n");
 9          if(x > 6)
10              printf("2\n");
11          if(y > 9)
12              printf("3\n");
13          else
14              printf("4\n");
15      }
16      else
17          printf("5\n");
18      return 0;
19  }
```

Imprime:
1
4

Comandos Condicionais: if-else

- Quando queremos uma dentre várias alternativas podemos usar uma sequência de ifs encaixados (else if)

```
if(operacao == 1)
{
    comando1;
}
else if(operacao == 2)
{
    comando2;
}
else if(opracao == 3)
{
    comando3;
}
else
{
    comando;
}
```

Comandos Condicionais: if-else

- Exemplo:

```
if(dia == 1) {
    printf("Domingo");
} else if(dia == 2) {
    printf("Segunda-feira");
} else if(dia == 3) {
    printf("Terca-feira");
} else if(dia == 4) {
    printf("Quarta-feira");
} else if(dia == 5) {
    printf("Quinta-feira");
} else if(dia == 6) {
    printf("Sexta-feira");
} else if(dia == 7) {
    printf("Sabado");
} else {
    printf("Dia de semana invalido");
}
```

Comandos Condicionais: switch

- Alternativamente, o comando switch substitui o uso de if's encaixados quando o teste é feito em uma variável

```
switch(variavel)
{
    case valor1:
        comandos_para_valor1;
        break;
    case valor2:
        comandos_para_valor2;
        break;
    default:
        comandos_caso_testes_falhem;
}
```

Comandos Condicionais: switch

- Exemplo:

```
switch(dia)
    {
        case 1:
            printf("Domingo");
            break;
        case 2:
            printf("Segunda");
            break;
        ...
        case 7:
            printf("Sabado");
            break;
        default:
            printf("Dia de semana invalido");
    }
```

Exercício
Par ou Ímpar

- Escreva um programa que lê um número e informa se é par ou ímpar

Exercício
Par ou Ímpar

```
 1  #include <stdio.h>
 2  #include <stdlib.h>
 3
 4  int main()
 5  {
 6      int n;
 7      printf("Informe um numero: ");
 8      scanf("%d", &n);
 9      if(n % 2 == 0)
10      {
11          printf("Numero e par!");
12      }
13      else
14      {
15          printf("Numero e impar");
16      }
17      return 0;
18  }
```

Exercício
Ordem Crescente

- Escreva um programa que lê dois números e os imprime em ordem crescente

Exercício
Ordem Crescente

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

int main()
{
    int x, y;
    printf("Informe o primeiro numero: ");
    scanf("%d", &x);
    printf("Informe o segundo numero: ");
    scanf("%d", &y);
    if(x > y)
    {
        printf("A ordem e %d %d", y, x);
    }
    else
    {
        printf("A ordem e %d %d", x, y);
    }
    return 0;
}
```

Exercício
Equação do Segundo Grau

- Escreva um programa que leia os coeficientes a, b e c de uma equação do segundo grau, calcule as raízes de acordo com a fórmula de Bhaskara e imprima as raízes ao usuário.

$$\Delta = b^2 - (4 * a * c)$$

$$x = \frac{-b \pm \sqrt{\Delta}}{2 * a}$$

O programa deve exibir que não há raízes reais caso delta seja menor que 0.

Exercício
Equação Segundo Grau

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main() {
    float a, b, c, delta, x1, x2;
    printf("Informe a: ");
    scanf("%f", &a);
    printf("Informe b: ");
    scanf("%f", &b);
    printf("Informe c: ");
    scanf("%f", &c);

    delta = (b * b) - (4 * a * c);

    if(delta > 0) {
        x1 = (-b + sqrt(delta)) / (2 * a);
        x2 = (-b - sqrt(delta)) / (2 * a);
        printf("As raizes da equacao sao: %.2f e %.2f\n", x1, x2);
    } else if(delta == 0) {
        x1 = (-b + sqrt(delta)) / (2 * a);
        printf("A unica raiz e: %.2f\n", x1);
    } else {
        printf("Nao ha raizes reais!\n");
    }
    return 0;
}
```

Exercício Calculadora

- Escreva um programa que leia dois números inteiros, um operador e simule uma calculadora simples:
  - Caso '+' seja o operador, o programa deve exibir a soma dos números
  - Caso '-' seja o operador, o programa deve exibir a diferença dos números
  - Caso '*' seja o operador, o programa deve exibir o produto dos números
  - Caso '/' seja o operador, o programa deve exibir a divisão dos números
  - Caso o operador não seja nenhum dos caracteres anteriores, o programa deve exibir operação inválida

- Use a estrutura switch case para resolver esse exercício

Exercício Calculadora

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

int main() {
    int n1, n2;
    float res;
    char op;
    printf("Informe a operacao:\n");
    scanf("%c", &op);
    printf("Informe os dois operandos:\n");
    scanf("%d %d", &n1, &n2);
    switch(op) {
    case '+':
        res = n1 + n2;
        printf("Resultado de %d %c %d: %.2f", n1, op, n2, res);
        break;
    case '-':
        res = n1 - n2;
        printf("Resultado de %d %c %d = %.2f", n1, op, n2, res);
        break;
    case '*':
        res = n1 * n2;
        printf("Resultado de %d %c %d: %.2f", n1, op, n2, res);
        break;
    case '/':
        res = (float) n1 / n2;
        printf("Resultado de %d %c %d: %.2f", n1, op, n2, res);
        break;
    default:
        printf("Operacao invalida!\n");
    }
    return 0;
}
```

Bibliografia

- Anido, R., Notas de aula. UNICAMP, disponível em http://www.ic.unicamp.br/~ranido/mc102/, acessado em Maio de 2015.
- Mota, V. F., Notas de aula. UFMG. Disponível em https://sites.google.com/site/virginiaferm/home/disciplinas acessado em Maio de 2015.
- Deitel, P, Deitel, H. C How to Program. 6a Ed. Pearson, 2010.