Estruturas de controle

Prof. Thiago Felski Pereira, MSc.

Operadores Relacionais

- Os operadores relacionais trabalham como comparações, igualdades e desigualdades. Eles verificam os valores dos operandos, que ficam cada um de um lado da operação, retornando **VERDADEIRO** ou **FALSO**

OPERADOR	SIGNIFICADO	EXEMPLO
>	maior que	5 > 4 = **V**
<	menor que	5 < 4 = **F**
>=	maior ou igual a	5 >= 4 = **V**
<=	menor ou igual a	5 <= 4 = **F**
!=	diferente de	5 <> 4 = **V**
==	igual a	5 == 4 = **F**

Exercícios

- Resolva as expressões a seguir (Considere que todos os números são inteiros)
  - 7 > 8 - 1
  - 5 + 3 == 8
  - 7 % 4 > 3
  - 19 / 3 % 10
  - 5 * 2 == 20 / 2

Exercícios

- Resolva as expressões a seguir (Considere que todos os números são inteiros)
  - 7 > 8 - 1 **FALSO**
  - 5 + 3 == 8 **VERDADEIRO**
  - 7 % 4 > 3 **FALSO**
  - 19 / 3 % 10 **NÃO É UMA OPERAÇÃO RELACIONAL**
    - Mas pode ser usada como tal, pois em C/C++ o inteiro zero é falso e os demais valores são verdadeiros
  - 5 * 2 == 20 / 2 **VERDADEIRO**

Operadores Lógicos

- Os operadores lógicos são aqueles que analisam condições, são operadores que analisaram operandos com valores lógicos 1 ou 0, ou então, Verdadeiro ou Falso. Uma situação pode ou não ser verdadeira, ou falsa, dependendo da condição em que se encontra. Temos três operações lógicas básicas: E (AND), OU (OR) e NÃO (NOT). Para podermos resolver expressões que contenham operações lógicas, precisamos consultar as TABELAS VERDADE de cada um desses operadores.

Operador	Operação Lógica	Exemplo	Prioridade
! (Negação)	! lógica	!(x >= y)	1
&&, and (e)	op1 && op2	m < n && i > j	2
||, or (ou)	op1 || op2	m == 5 || n != 10	3

Operação Não (!)

- O operador NÃO ou NOT nega a entrada, portanto, se o operador vale 1 (V) e vai se transformar em 0 (F), e vice-versa

Operando a	!a (not a)
0 (false)	1 (true)
1 (true)	0 (false)

- Note que o operador relacional de diferente (!=) é uma igualdade negada

Operação E (&&)

- O operador E é o mesmo que o operador multiplicação, você multiplica os valores lógicos (1 ou 0) dos operandos 1 e 2 e obtém o resultado desejado

OPERANDO 1	OPERANDO 2	OP1 && OP2
0 (false)	0 (false)	0 && 0 = 0 (false)
0 (false)	1 (true)	0 && 1 = 0 (false)
1 (true)	0 (false)	1 && 0 = 0 (false)
1 (true)	1 (true)	1 && 1 = 1 (true)

Operação OU (||)

- O operador OU é o mesmo que o operador soma, você soma os valores lógicos (1 ou 0) dos operandos 1 e 2 e obtém o resultado desejado

OPERANDO 1	OPERANDO 2	OP1 || OP2
0 (false)	0 (false)	0 || 0 = 0 (false)
0 (false)	1 (true)	0 || 1 = 1 (true)
1 (true)	0 (false)	1 || 0 = 1 (true)
1 (true)	1 (true)	1 || 1 = 1 (true)

- Observe que o resultado só será 0 (false) quando os operandos 1 e 2 forem também 0 (false), caso contrário, o resultado será sempre 1 (true). Assim fica mais fácil você memorizar

Exercícios

- Resolva as expressões a seguir (Considere que todos os números são inteiros)
  - 7 + 6 >= 8 || 5 < 2
  - 7 == 7 % 8 && (! falso)
  - ! (5 < 2 || 7 < 2 * 4)
  - 13 / 2 * 2 == 13
  - 7 % 2 == 1 && 8 % 2 == 0

Exercícios

- Resolva as expressões a seguir (Considere que todos os números são inteiros)

  - 7 + 6 >= 8 || 5 < 2
    - verdadeiro || falso = verdadeiro
  - 7 == 7 % 8 && (! falso)
    - verdadeiro && verdadeiro = verdadeiro
  - ! (5 < 2 || 7 < 2 * 4)
    - !(falso || falso) = verdadeiro
  - 13 / 2 * 2 == 13
    - falso
  - 7 % 2 == 1 && 8 % 2 == 0
    - verdadeiro && verdadeiro = verdadeiro

Exercícios

- Qual das alternativas pode ser utilizada para verificar se X pertence ao intervalo [8, 19] (incluindo o 8 e o 19)
  - a) X >=8 || X <= 19
  - b) X == 8 && X == 19
  - c) X >= 8 && X <= 19
  - d) X > 8 || x < 19
  - e) X < 8 && x > 19

Exercícios

- Qual das alternativas pode ser utilizada para verificar se X pertence ao intervalo [8, 19] (incluindo o 8 e o 19)
  - a) X >=8 || X <= 19
  - b) X == 8 && X == 19
  - **c) X >= 8 && X <= 19**
  - d) X > 8 || x < 19
  - e) X < 8 && x > 19

Estruturas de controle

- Estruturas de controle dizem respeito a ordem em que as instruções são executadas em um computador

- As estruturas de controle podem ser classificadas em 3 tipos básicos de execução
  - **Sequencial**: em um algoritmo sequencial se executam todas as instruções na sequência em que elas aparecem, sem omissões, escolhas ou repetições

  - **Condicional**: quando existe um desvio condicional um bloco de código pode ou não ser executado dependendo de um determinado teste lógico
    - O sucesso em programar códigos com teste lógico dependem diretamente da habilidade com operadores relacionais e lógicos

  - **Repetição**: Em uma estrutura de repetição um determinado bloco de código pode ou não ser executado enquanto um determinado teste lógico for verdadeiro

Estruturas de controle sequencial

- A estrutura de desvio condicional deve ser utilizada quando se quer que uma condição seja analisada e:
  - caso esta condição seja verdadeira, o(s) comando(s) logo abaixo do teste lógico será(ão) executado(s)
  - caso esta condição seja falsa, outro(s) comando(s) será(ão) executado(s)

- Os desvios condicionais podem ser de três tipos:
  - Simples
  - Compostos
  - Encadeados

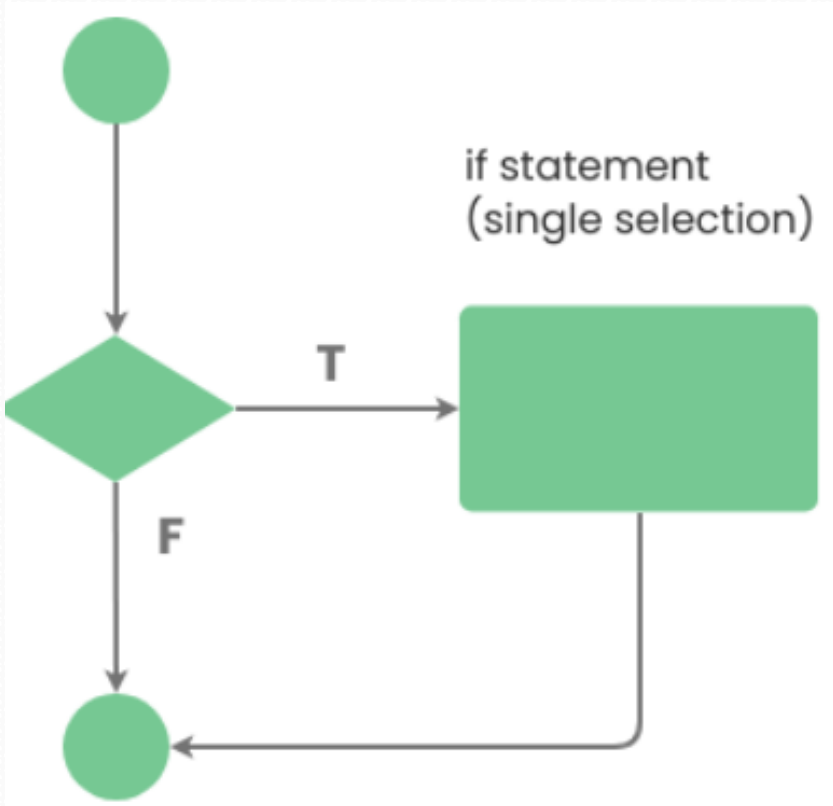
Estruturas de controle condicional

- **Comando IF Simples**
  - SE a condição avaliada for Verdadeira, ENTÃO, execute os comandos deste bloco. SE a condição avaliada for Falsa, ENTÃO saia deste bloco e retorne ao programa principal.

- **Sintaxe**

C#

if (teste(s) lógico(s)) {
... Bloco de Instruções ...
}



UNIVALI

Estruturas de controle sequencial

- Exemplos de desvio condicional simples

```
if (media >= 6.0){
    cout<<"aluno aprovado";
}
```

```
if (media >= 6.0 && frequencia >= 0.75){
    cout<<"aluno aprovado";
}
```

- **Exercício**
  - Faça um algoritmo em que o usuário informa 3 médias de um aluno e, caso ele tenha obtido media igual ou superior a 6 informe que o aluno foi aprovado

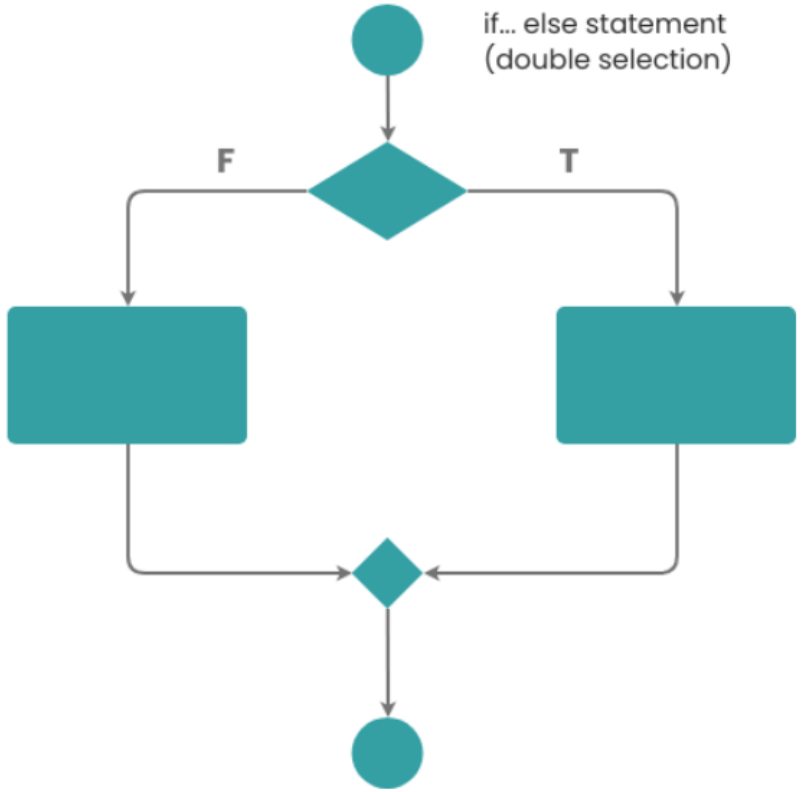
Estruturas de controle condicional

- **Comando IF Composto**
- Se a condição do if for verdadeira será executado o bloco de instruções entre o if e o fechamento de chaves do if. Caso sejam falsa, será executado o bloco de instruções entre as chaves do else.

- **Sintaxe**

C#

if (teste(s) lógico(s)) {
  ... Bloco de Instruções ...
} else {
  ... Bloco de Instruções ...
}



UNIVALI

Estruturas de controle sequencial

- **Exercício**
  - Mas ficou faltando algo no exercício anterior. E se fosse necessário imprimir que o aluno foi reprovado? Tente resolver o problema com o uso de desvio condicional composto

Estruturas de controle sequencial

- **Exercício**
  - Mas ficou faltando algo no exercício anterior. E se fosse necessário imprimir que o aluno foi reprovado? Tente resolver o problema com o uso de desvio condicional composto

```
1  using System;
2  public class Program {
3      public static void Main() {
4          double a1, a2, a3, media;
5          Console.WriteLine("Informe 3 notas");
6          a1 = Convert.ToDouble( Console.ReadLine() );
7          a2 = Convert.ToDouble( Console.ReadLine() );
8          a3 = Convert.ToDouble( Console.ReadLine() );
9          media = (a1 + a2 + a3) / 3;
10         if (media >= 6.0) {
11             Console.WriteLine("Aluno aprovado com média " + media);
12         } else {
13             Console.WriteLine("Aluno reprovado com média " + media);
14         }
15     }
16 }
```

Estruturas de controle sequencial

- **Exercício** (esse é um pouco mais desafiador)
- Solicite ao usuário um ano e informe se o mesmo é um ano bissexto ou não.
- Ano bissexto é aquele que é
  - divisível por 4, mas não pode ser divisível por 100.
  - A não ser que seja divisível por 400.

Estruturas de controle sequencial

- **Exercício** (esse é um pouco mais desafiador)
- Solicite ao usuário um ano e informe se o mesmo é um ano bissexto ou não.
- Ano bissexto é aquele que é
  - divisível por 4, mas não pode ser divisível por 100.
  - A não ser que seja divisível por 400.

```
1  using System;
2  public class Program {
3      public static void Main() {
4          int ano;
5          Console.Write("Informe um ano: ");
6          ano = Convert.ToInt32( Console.ReadLine() );
7
8          if (ano % 4 == 0 & ano % 100 == 0 || ano%400 == 0) {
9              Console.WriteLine("O ano: " + ano + " é bissexto");
10         } else {
11             Console.WriteLine("O ano: " + ano + " não é bissexto");
12         }
13     }
14 }
```

Estruturas de controle sequencial

- **Comando IF encadeado**
  - Como visto anteriormente o desvio condicional permite definir conjuntos de instruções que serão executados caso uma condição seja satisfeita ou não
  - Desvio Condicional também se trata de uma instrução, logo é possível incluir desvios condicionais um dentro do outro
  - E quando isso é feito? Quando se possui três ou mais instruções a serem testadas
  - Ao se encadear um desvio deve-se tomar cuidado, pois uma chave fechada em um local errado pode mudar toda a lógica do programa
  - Existem várias formas de encadeamento, dependendo do que é solicitado.

- **Sintaxe**
  - ...

Estruturas de controle sequencial

- ...
- **Sintaxe**

C#

```
if (teste(s) lógico(s)) {
      ... Bloco de Instruções ...
} else if (teste(s) lógico(s)) {
      ... Bloco de Instruções ...
} else {
     ... Bloco de Instruções ...
}
```

C#

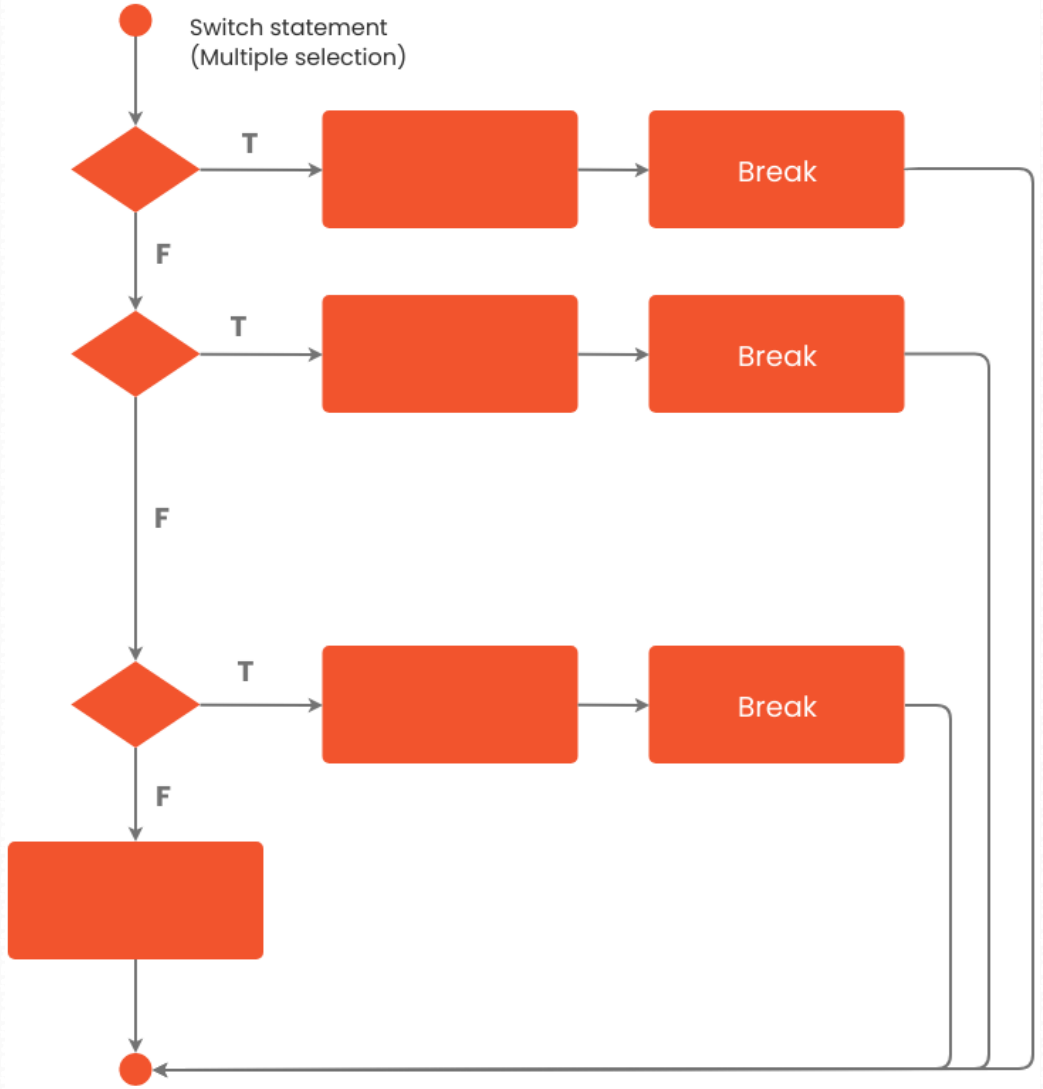
```
if (teste(s) lógico(s)) {
      ... Bloco de Instruções ...
      if (teste(s) lógico(s)) {
          ... Bloco de Instruções ...
      }
} else if (teste(s) lógico(s)) {
       ... Bloco de Instruções ...
}
```

Estruturas de controle sequencial

- **Exercício**
  - Solicite ao usuário um valor inteiro e informe se o mesmo é positivo, negativo ou zero
  - Solicite ao usuário as medidas dos lados de um triângulo e informe se o mesmo é equilátero, escaleno ou isósceles
    - Equilátero = 3 lados iguais
    - Escaleno = 2 dos 3 lados iguais
    - Isósceles = os 3 lados são diferentes
  - Faça um algoritmo que solicite ao usuário a média e a frequência final de uma disciplina e informe se o aluno foi aprovado ou reprovado, e em caso de reprovação se foi por nota ou média.

Estruturas de controle sequencial

- **Comando switch**
  - Chamado também de Desvio por Associação.
  - Encontrada uma associação, a instrução é executada.
  - A variável é testada contra uma lista de inteiros ou constantes caracteres.
- Dica: Utilize o comando switch para menus, pois ele fica mais claro que o comando if ... else.



UNIVALI

Estruturas de controle sequencial

- **Sintaxe do Switch**

- Armadilha: Ao esquecer um break no comando switch, o compilador não emitirá mensagem de erro, mas não será efetuado o que você desejava

C#

```
switch (expressao_de_controle)
{
    case <constante_1> :
        <comando(s)>;
        break;
    case <constante_2> :
        <comando(s)>;
        break;
    ...
    ...
    case <constante_n>
         <comando(s)>;
         break;
     default:
         <comando(s)>;
}
```

Estruturas de controle sequencial

- **Operador ternário :?**
  - Atividade para casa
  - Pesquise sobre como utilizar esse tipo de desvio, por mais que ele seja pouco utilizado é sempre bom saber doque se trata quando se deparar com ele.

Obrigado pela atenção

contato: Felski@univali.br

UNIVALI