Lógica de Programação e Algoritmos

Estruturas de Múltipla Escolha

Profa. Eliane Oliveira Santiago

Estruturas de Múltipla Escolha

Uma estrutura de múltipla escolha também é uma estrutura de decisão em que um fluxo de controle é utilizado para decidir qual o fluxo que o algoritmo seguirá. Uma estrutura de múltipla escolha determina qual conjunto de comandos ou bloco será executado após uma opção ser avaliada e um caso para essa opção for detectado. Essa opção é representada por valores de quaisquer tipos de dados conhecidos. Note que somente um caso é executado, dentre todos os casos da estrutura de múltipla escolha.

Variáveis Discretas x Contínuas

Discretas

Possuem um conjunto finito, não vazio e conhecido de valores que a variável poderá assumir.

Exemplo: tempo, genero, idade, peso

Contínuas

- int n;
- double x;
- float y;

Estruturas de Decisão

Se… senão.. (if… else..)

- Usa variáveis discretas e contínuas
- Operadores relacionais >, >=, <, <=, ==, !=
- Opera com variáveis de todos os tipos

escolha (var)… caso (switch… case)

- Usa variáveis discretas
- Não usa operadores relacionais e é verificada apenas a relação de igualdade
- Opera com variáveis do tipo inteiro e caracter

Exemplo de Variável Discreta

declare tempo: **char**

escreva("Como está o tempo?")

leia(tempo)

//o usuário digitará E se ensolarado; N se Nevoento...

- E: Ensolarado
- N: Nevoento
- B: Nublado
- C: Chuvoso

- Bom
- Frio
- Ruim
- Limpo
- Garoando
- Relampejando
- Geando
- Ventando
- Agradável

Estrutura de múltipla escolha

Uma estrutura de múltipla escolha (na liguagem de programação switch .. case) pode ser construída dentro de uma estrutura de decisão (na linguagem de programação if ... else), seja ela simples, composta ou encadeada, assim como qualquer estrutura de decisão pode ser construída dentro de uma estrutura de múltipla escolha.

Uma estrutura de múltipla escolha pode ser simples ou encadeada. Veremos a seguir como e quando trabalhar com cada uma delas.

Pseudocódigo

```
escolha (<opção>)
   caso <opção_1> : <comandos_1>;
   caso <opção_2> : <comandos_2>;
   ...
   caso <opção_n> : <comandos_n>;
   caso contrário  : <mensagem>;
fimescolha;
```

Sintaxe

Exemplo

```
declare var: inteiro
leia(var)
escolha (var)
   caso 1: escreva("O valor da variável var é 1");
   caso 2: escreva("O valor da variável var é 2");
   caso 3: escreva("O valor da variável var é 3");
```

Pseudocódigo

```
escolha (<opção>)
   caso <caso_1> : <comandos_1>;
   caso <caso_2> : <comandos_2>;
   ...
   caso <caso_n> : <comandos_n>;
   caso contrário  : <mensagem>;
fimescolha;
```

```
declare tempo: caracter
escreva("Como está o tempo?")
leia(tempo)

escolha (tempo)
   caso 'N' : escreva("nublado");
   caso 'E' : escreva("ensolarado");
   caso 'C' : escreva("chuvoso");
   ...
   caso 'V' : escreva("nevoento");
   caso contrário  : escreva("invalido!");
fimescolha;
```

Exemplo

```
declare var: inteiro
leia(var)
escolha (var)
   caso 1: escreva("O valor da variável var é 1");
   caso 2: escreva("O valor da variável var é 2");
   caso 3: escrever ("O valor da variável var é 3");
```

Explicação do exemplo

Nesse exemplo, a variável **var** possui um valor numérico **inteiro** e vai passar por uma avaliação, ou seja, o valor da variável **var** será avaliada. Se o resultado de retorno dessa avaliação for o número inteiro 1, então o comando **escreva(...);** do **caso 1** será executado, se for o número inteiro 2, então o comando **escreva(...);** do **caso 2** será executado, se for o número inteiro 3, então o comando **escreva(...);** do **caso 3** será executado e se for diferente de 1,2 ou 3, então o **escreva(...);** do **caso contrário** será executado.

Note que somente um dos casos é executado e que o tipo de dado da variável **var** é numérico **inteiro** assim como os valores de cada um dos casos.

Note que o uso do comando **caso contrário** serve para os valores não previstos que a variável **var** possa assumir.

Numa estrutura de múltipla escolha, os valores escolhidos para cada caso não precisam ser únicos, isto é, cada caso da estrutura pode ser representado por um conjunto de valores específicos.

Nesse caso, a estrutura de múltipla escolha simples para pseudocódigo segue a seguinte regra sintática:

Múltiplos casos para um bloco de comandos.

```
escolha (<opção>)
   caso <caso_1> : <comandos_1>;
   caso <caso_2>, <caso_3>, <caso_4> : <comandos_2>;
   caso <caso_5> ... <caso_10> : <comandos_3>;
   ...
   caso <caso_11>, <caso_15> ... <caso_20> : <comandos_4>;
   caso contrário  : <mensagem>;
fimescolha;
```

Exemplo

```
escolha (var)
   caso 1 :  escreva ("O valor da variável var é 1");
   caso 2 ... 4 :  escreva ("O valor da variável var pode ser 2, 3 ou 4");
   caso 7, 15, 25 :  escreva("O valor da variável var pode ser 7, 15 ou 25");
   caso 8, 11 ... 14 :  escreva("O valor da variável var pode ser 8 , 11, 12, 13 ou 14");
   caso_contrário  : escreva("opção inválida!");
fimescolha;
```

Nesse exemplo, a variável var possui um valor numérica **inteiro** vai passar por uma avaliação, ou seja, o valor da variável **var** será avaliado. Se o resultado de retorno dessa avaliação for o número inteiro 1, então o comando escreva(...); do caso 1 será executado, se for o número inteiro 2, 3 ou 4, então o comando escreva(...); do caso 2 ... 4 será executado, se for o número inteiro 7, 15, ou 25, então o comando escreva(...); do caso 7, 15, 25 será executado, se for 8, 11, 12, 13 ou 14, então o comando escreva(...); do caso 8, 11 .. 14 será executado e se for diferente de qualquer um dos valores anteriores, então o escreva(...); do caso contrário será executado. Note que somente um dos casos é executado e que o tipo de dado da variável var é numérica **inteiro** assim como os valores de cada um dos casos.

Estrutura de Múltipla Escolha Encadeada

Uma estrutura de múltipla escolha encadeada pode ser utilizada quando o algoritmo precisa testar um conjunto de valores diferentes antes de executar um conjunto de comandos associados a esses valores. Dizemos que essa estrutura é encadeado, pois há estruturas de múltipla escolha dentro de outras estruturas de múltipla escolha. Para qualquer caso, cada opção é avaliada e para cada resultado, um conjunto de comandos dentro da estrutura **escolha**/**caso** pode ser executado. Não existe limite para a estrutura de múltipla escolha encadeada, podemos ter quantas estruturas de múltipla escolha encadeadas forem necessárias. Note que, para qualquer estrutura de múltipla escolha, no máximo, um único conjunto de comando será executado, aquele cuja avaliação de opção for escolhida.

Sintaxe

```
escolha (<opção_1>)
  caso <caso_1> : escolha (<opção_2>)
                    caso <caso_11> : escolha (<opção_3>)
                                       caso <caso_111> : <comandos_111>;
                                       caso <caso_112> : <comandos_112>;
                                       ...
                                       caso <caso_11n> : <comandos_11n>;
                                       caso contrário  : <mensagem>;
                                     fimescolha; // opção 3
                    caso <caso_12> : <comandos_12>;
                    ...
                    caso <caso_1n> : <comandos_1n>;
                    caso contrário  : <mensagem>;
                  fimescolha; // opção 2
  caso <caso_2> : escolha (<opção_4>)
                    caso <caso_21> : <comandos_21>;
                    caso <caso_22> : <comandos_22>;
                    ...
                    caso <caso_2n> : <comandos_2n>;
                    caso contrário  : <mensagem>;
                  fimescolha; // opção 4
  caso <caso_3> : <comandos_3>;
  ...
  caso <caso_n> : <comandos_n>;
  caso contrário  : <mensagem>;
fimescolha; // opção 1
```

Exemplo

```
escolha (var)
  caso 1 . . 3 : escolha (var)
                    caso 1 , 2 : escolha (var)
                                    caso 1 : escreva("o valor da variável var é 1");
                                    caso 2 : escreva("o valor da variável var é 2");
                                    caso contrário : escreva("opção inválida");
                                  fimescolha;
                    caso 3 : escreva("o valor da variável var é 3");
                    caso contrário : escreva("opção inválida");
                  fimescolha;
  caso 4 , 5  : escolha (var)
                    caso 4 : escreva("o valor da variável var é 4");
                    caso 5 : escreva("o valor da variável var é 5");
                    caso contrário : escrever("opção inválida");
                  fimescolha;
  caso 6       : escreva("o valor da variável var é 6");
  caso contrário : escrever ("opção inválida");
fimescolha;
```

Sobre o exemplo

Nesse exemplo, a variável **var** possui um valor **inteiro** e vai passar por uma avaliação, ou seja, o valor da variável **var** será avaliado. Se o resultado de retorno dessa avaliação for o número inteiro 6, então o comando **escreva(...);** do **caso 6** será executado, se for o número inteiro 4 ou 5, então uma nova estrutura de múltipla escolha é avaliado. Se nessa avaliação, o valor de **var** for o número 4, o comando **escreva(...);** do **caso 4** será executado, se for o número 5, o comando **escreva(...);** do **caso 5** será executado. Se o resultado da avaliação do primeiro **var** for o número inteiro 1, 2, ou 3, então uma nova estrutura de múltipla escolha é avaliada e assim por diante. Note que somente um dos casos é executado, ou seja, uma única mensagem **escreva(...);** será apresentada ao usuário e que o tipo de dado da variável **var** é **inteira** assim como os valores de cada um dos casos.

Desenvolva um algoritmo que receba o preço de um produto e seu código de origem e mostre o preço do produto junto de sua procedência, conforme tabela abaixo:

código de origem	região de procedência
1	norte
2, 5, 9	sul
3 , 10 até 15	leste
7 ou 20	oeste
qualquer outro	importado

```
Algoritmo Produto
inicio
   declare origem: inteiro;
   escreva("Informe o código de origem");
   leia(origem)

   escolha (origem)
     início
        caso 1                : escreva("Norte");
                                //break; colocar na linguagem de programação
         caso 2, 5, 9         : escreva("Sul");
                                //break;
         caso 3, 10 ... 15    : escreva("Leste");
                                //break;
         caso 7, 20           : escreva("Oeste");
                                //break;
         caso_contrário        : escreva("Importado!");
      Fim
   fimescolha;
```

Desenvolva um algoritmo que receba o preço de um produto e seu código de origem e mostre o preço do produto junto de sua procedência, conforme tabela abaixo:

código de origem	região de procedência
1	norte
2, 5, 9	sul
3 , 10 até 15	leste
7 ou 20	oeste
qualquer outro	importado

```
Algoritmo Produto
inicio
   declare regiao: caracter;
   escreva("Informe a região de precedência")
   leia(regiao)

   escolha (regiao)
        caso 'N'              : escreva("Norte");
                                escreva("Código de origem 1");
                                //break;
         caso 'S'             : escreva("Sul");
                                escreva("Códigos de origem 2, 5, e 9");
                                 //break;
         caso 'L'             : escreva("Leste");
                                escreva("Códigos de origem 3, 19 até 15");
                                 //break;
         caso 'O'             : escreva("Oeste");
                                escreva("Códigos de origem 7 ou 20");
                                 //break;
         caso_contrário        : escreva("importado");
   fimescolha;
```

Um exemplo em Linguagem de Programação

```
int mes = 9;
String mes_extenso;
switch (mes){
  case 1: mes_extenso = “Janeiro”;
            break;
  case 2: mes_extenso = “Fevereiro”;
           break;
  :
  :
 case 9: mes_extenso= “Setembro”;
            break;
 :
 case 12: mes_extenso= “Dezembro”;
            break;
 default: mes_extenso = “Mes Invalido!”;
}
```

Desenvolva um algoritmo que receba o preço de um produto e seu código de origem e mostre o preço do produto junto de sua procedência, conforme tabela abaixo:

código de origem	região de procedência
1	norte
2, 5, 9	sul
3 , 10 até 15	leste
7 ou 20	oeste
qualquer outro	importado

```
Algoritmo Produto
  início_algoritmo
    // declaração de variáveis e/ou constantes
    Var
      preco real;
      codigo inteiro;

    // mensagem ao usuário
    escreva("Digite o preço e o código de origem do produto");
    // entrada de dados
    ler (preco, codigo);
    // processamento de dados
    escolha (codigo)
      caso 1            : // entrada de dados
                          escreva(preco , " - Norte");
      caso 2, 5, 9      : // saída de resultados
                          escreva(preco ," - Sul");
      caso 3, 10 .. 15 : // saída de resultados
                          escreva(preco , " - Leste");
      caso 7 , 20       : // saída de resultados
                          escreva(preco , "- Oeste");
      caso contrário : // saída de resultados
                          escreva(preco , "- Importado");
    fimescolha;
  fim_algoritmo.
```

Desenvolva um algoritmo que receba dois valores numérico inteiro e o símbolo da operação conforme tabela abaixo, calcule e mostre a operação efetuada:

Símbolo da operação	Nome da operação
+	adição
-	subtração
*	multiplicação
/	divisão

```
Algoritmo Operacoes
inicio
    declare             n1, n2: inteiro
                        resultado: real
                        operador: caracter

    escreva("Menu de Operadores")
    escreva("[+] Adicao")
    escreva("[-] Subtração")
    escreva("[/] Divisão")
    escreva("[*] Multiplicação")

    escreva("Digite o primeiro termo: ")
    leia(n1)
    escreva("Digite o segundo termo: ")
    leia(n2)
    escreva("Digite a operação desejada: ")
    leia(operador)

    escolha(operador)
    início
          caso '+'      :   escreva("Somar os dois termos")
                            resultado <- n1 + n2
                            escreva("A divisão maior pelo menor eh
                            igual a ", resultado)
                            //break
          caso '-'      :   escreva("A dirença entre os dois termos é")
                            casocontrario:  escreva("Operador
                            inválido!")
                            fimescolha
                            Fim.
                            escreva("A diferenca de ", n1, "e ", n2, "
                            eh igual a ", resultado)
                            //break
          caso '*'      :   escreva("O produto dos dois termos é")
                            resultado <- n1 * n2
                            escreva("O produto de ", n1, "e ", n2, "
                            eh igual a ", resultado)
                            //break
          caso '/'      :   escreva("A divisão dos termos é")
                            se(n1>=n2)
                            resultado <- n1 / n2
                            senao
                            resultado <- n2 / n1

                            escreva("A divisão maior pelo menor eh igual a ", resultado)
                            //break
          casocontrario:    escreva("Operador inválido!")
    fimescolha
Fim.
```

Desenvolva um algoritmo que receba dois valores numérico inteiro e o símbolo da operação conforme tabela abaixo, calcule e mostre a operação efetuada:

Símbolo da operação	Nome da operação
+	adição
-	subtração
*	multiplicação
/	divisão

```
Algoritmo Cálculo
  início_algoritmo
    // declaração de variáveis e/ou constantes
    Declarar
      d numérico_real;
      res, num1, num2 numérico_inteiro;
      oper alfanumérico;

    // mensagem ao usuário
    escreva("Digite dois inteiros e a operação");
    // entrada de dados
    ler (num1, num2, oper);
    // processamento de dados
    escolha (oper)
      caso '+'  : res ← num1 + num2;
                  // saída de resultados
                  escreva("A soma de " , num1 , " com " , num2 , " é " , res);
      caso '-'  : res ← num1 - num2;
                  // saída de resultados
                  escreva("A diferença de " , num1 , " com " , num2 , " é " , res);
      caso '*'  : res ← num1 * num2;
                  // saída de resultados
                  escreva("O produto de " , num1 , " com " , num2 , " é " , res);
      caso '/'  : d ← num1 / num2;
                  // saída de resultados
                  escreva("A divisão de " , num1 , " com " , num2 , " é " , d);
      caso contrário : // saída de resultados
                       escreva("operação inválida");
    fimescolha;
  fim_algoritmo.
```

Exercícios de Estrutura de Múltipla Escolha Simples e Encadeada

1. Desenvolva um algoritmo que receba o nome e os dados para cálculo da área de uma figura geométrica conforme tabela abaixo, calcule e mostre a área da figura geométrica:

Figura geométrica	Fórmula
quadrado	lado * lado
triângulo	(base * altura) / 2
retângulo	base * altura
trapézio	((Base maior + base menor) * altura) / 2

2. 2. Desenvolva um algoritmo que receba dois valores reais e o código do produto notável conforme tabela abaixo, calcule e mostre o valor do produto notável:

Código	Produto Notável	Fórmula
1	Quadrado da diferença de dois números	(a – b) * (a – b)
2	Quadrado da soma de dois números	(a + b) * (a + b)
3	soma do quadrado de dois números	a * a + b * b
4	Diferença do quadrado de dois números	a * a – b * b
5	produto da soma com a diferença de dois números	(a – b) * (a + b)

Exercícios de Estrutura de Múltipla Escolha Simples e Encadeada

3. Desenvolva um algoritmo que receba o nome de um lugar e mostre para o usuário o que ele deve fazer nesse lugar, conforme tabela abaixo:

Lugar	O que fazer
Escola	Estudar
Banco	Pagar contar
Farmácia	Comprar remédios
Casa	Descansar
Correio	Remeter cartas

4. Desenvolva um algoritmo que receba um dia da semana e mostre qual(is) a(s) disciplina(s) você tem naquele dia da semana.

5. Desenvolva um algoritmo que receba um mês do ano e mostre qual(is) o(s) feriado(s) daquele mês.

6. Desenvolva um algoritmo que receba a data de nascimento de uma pessoa e mostre qual o signo dessa pessoa.

Exercícios de Estrutura de Múltipla Escolha Simples e Encadeada

7. Desenvolva um algoritmo que receba o valor de dois números inteiros, o símbolo da operação aritmética desejada, calcule e mostre o resultado da operação aritmética, conforme a tabela abaixo:

símbolo	Operação aritmética
+	adição
-	subtração
*	Multiplicação
/	divisão
^	Potenciação
m	Resto da divisão
q	Quociente da divisão

8. Desenvolva um algoritmo que receba a idade e o peso de uma pessoa, verifique e mostre em qual grupo de risco essa pessoa se encaixa, conforme a tabela abaixo:

Idade	Peso		
	Até 60 (inclusive)	Entre 60 e 90 (inclusive)	acima de 90
menores de 20	9	8	7
de 20 a 50	6	5	4
maiores de 50	3	2	1

Bibliografias

BÁSICA

- GOMES, Ana Fernanda A. Campos, Edilene Aparecida V. Fundamentos da Programação de Computadores – Algoritmos, Pascal e C/C++. Prentice Hall, 2007.
- CARBONI, Irenice de Fátima. Lógica de Programação. Thomson.
- XAVIER, Gley Fabiano Cardoso. Lógica de Programação - Cd-rom. Senac São Paulo – 2007.

COMPLEMENTAR

- FORBELLONE, André Luiz Villar. Eberspache, Henri Frederico. Lógica de Programação – A construção de Algoritmos e Estrutura de Dados. Makron Books, 2005.
- LEITE, Mário - Técnicas de Programação – Brasport - 2006.
- PAIVA, Severino – Introdução à Programação – Ed. Ciência Moderna – 2008.
- PAULA, Everaldo Antonio de. SILVA, Camila Ceccatto da. Lógica de Programação –Viena – 2007.
- CARVALHO, Fábio Romeu, ABE, Jair Minoro.Tomadas de decisão com ferramentas da lógica paraconsistente anotada: Método Paraconsistente de Decisão (MPD), Editora Edgard Blucher Ltda. - 2012.

Dica de solução

```
escolha (mes)
   caso 3 :
                        escreva(“mes de março”);
                        escolha(dia)
                                    caso 1 .. 20: escreva(“peixes”)
                                    caso 21 .. 31: escreva(“áries”)
                        fim_escolha
   caso 4 :
                        escreva (“abril”)
                        escolha(dia)
                                    caso 1..20: escreva(“Áries”)
                                    caso 21..30: escreva(“touro”)
                        fim_escolha.
   caso contrário  : <mensagem>;
fimescolha;
```