Disciplina: Lógica de programação **Professora:** Sandra Hedler de Amorim **E-mail:** sandra-famorim@educar.rs.gov.br

Revisão 2

Objetivos da Aula

Revisão sobre Estruturas Simples(se...entao...fimse) e Compostas(se...entao...senao...fimse)

Estudo para dar início aos desvios condicionais, dentro dos nossos algoritmos.

Seleção Simples:

Sintaxe:

```
se (condição) entao
//bloco verdade
fimse
```

Descreve as ações que serão executadas a partir de uma condição verdadeira.

Significa se essa condição for verdadeira executa o bloco verdade. Mas se a condição não for verdadeira e sim falsa, não executa o bloco verdade e segue executando o algoritmo a partir do fimse.

O que é uma condição? É uma expressão lógica ou uma expressão relacional:

Que trabalha com os operadores relacionais:

- = igual
- > maior que
- < menor que
- >= maior ou igual
- <= menor ou igual
- <> diferente

E trabalha com os operadores lógicos:

```
nao negação de operaçãoou disjunção de operaçãoe conjunção de operação
```

Uma condição tem que ter um operador relacional.

```
Seleção Composta:
Sintaxe:
se (condição) entao
//bloco verdade
```

senao

//bloco falso

fimse

A estrutura de seleção composta descreve as ações que serão executadas a partir da condição verdadeira e também descreve ações caso essa condição seja falsa.

Se a condição for verdadeira então executa o bloco verdade e finaliza o **fimse**, segue executando as instruções no algoritmo.

Se a condição não for verdadeira, não executa o bloco verdade e vai pro **senao** e executa o bloco falso, e finaliza o **fimse** e segue executando até o fim algoritmo.

A seguir um exemplo de um algoritmo para fixar essa estrutura.

Um algoritmo que lê um número inteiro, verifica se esse número é diferente de zero para testar se esse número é par ou ímpar. Ou seja, faz uma validação antes da verificação se o número é par ou ímpar.

```
senao
escreval(n," é ímpar")
fimse

senao
escreva("Erro, o número deveria ser diferente de zero! ")
fimse
```

fimalgoritmo

Onde temos o nome do algoritmo exemplo1, e a variável n que vai receber o número que vamos verificar se é par ou ímpar e a variável aux, vai ajudar calcular o resto da divisão por 2 para verificar se o número vai ser par ou ímpar. Todas essas variáveis declaradas, sendo do tipo inteiro, para trabalhar com números pares e ímpares tem que ser do tipo inteiro.

No corpo do algoritmo temos a primeira linha de instrução pedindo para o usuário digitar um número inteiro diferente de zero. E assim que o usuário confirmar teclando ENTER esse número vai ficar armazenado temporariamente na variável n.

Após a verificar desse número, porque antes de saber se é par ou ímpar temos que validar. Interno a essa estrutura um comando de seleção composto para verificar se esse número é par ou ímpar.

Então primeiro fizemos a verificação se o número digitado pelo usuário que está armazenado em n, for diferente de zero (0) e então verifica se é par ou ímpar. Se não emite uma mensagem de erro dizendo que esse número deveria ser diferente de zero, e finaliza a estrutura composta.

Devemos sempre cuidar, que cada comando de seleção **se** aberto é finalizado com **fimse**.

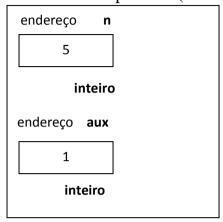
Têm-se uma estrutura composta e dentro dela tenho outra estrutura de seleção composta, primeiro finalizo essa estrutura composta que está dentro dessa estrutura. Então sempre cuidar onde está finalizando essa estrutura composta.

Vamos fazer o **teste de mesa** do que acabamos de fazer no algoritmo a cima para terem uma melhor clareza do que foi feito:

Quando fazemos o teste de mesa, estamos simulando o que acontece na tela do usuário e na memória do computador quando o algoritmo estiver em execução. Sabemos que o algoritmo é executado de cima para baixo, da esquerda para direita.

Teste de Mesa

Memória Computador (RAM)



Tela do usuário

exemplo1
Digite um número inteiro diferente de zero: 5
5 é impar

Primeiro executa a linha com o nome do algoritmo e abre a tela do usuário já com o nome desse algoritmo

Segundo, na seção de declaração de variáveis, acontece a locação de endereço de memória para as variáveis n e aux, para armazenar o conteúdo nessas variáveis. Está dizendo para o computador alocar endereços de memórias referenciando como n e aux.

Após inicia o corpo do algoritmo temos um comando de saída (escreva) que dispara para tela a mensagem para o usuário.

Na linha seguinte o comando de entrada (leia), que é o responsável para mostrar na tela o ponto de inserção intermitente esperando o usuário digitar, por exemplo, o usuário digita o número 5, e confirmar com ENTER. Esse comando de entrada captura o valor digitado e armazena na variável n.

Na próxima linha tem a estrutura de seleção que verifica se o conteúdo dessa variável é diferente de zero e sendo verdadeiro entra no bloco verdade e faz o processamento de aux <- n mod 2. Nesse momento o computador vai calcular uma expressão aritmética. Pega o conteúdo da memória de n que é 5 e divide por 2 e retorna o resto da divisão inteira que é 1. Esse valor vai ser armazenado na variável aux que é endereço de memória referenciado para armazenar o valor desse processamento.

E a seguir verifica se aux, ou seja, o conteúdo da variável aux é igual a 0. O conteúdo da variável aux é 1, então a condição retorna falso e vai executar o bloco de falso mostrando na tela 5 e seguido da cadeia de caracteres que é ímpar. E finaliza o fimse, fimse e fim algoritmo.

Com fim do algoritmo já não existe mais os endereços reservado para as variáveis e o usuário pode fechar a tela do computador que mostra essas saídas de dados.