Comando de decisão - SE

- Sintaxe do comando:

  se <expressão_lógica> **então**

  <comandos_verdadeiro>

  OU

  se <expressão_lógica> **então**

  <comandos_verdadeiro>

  senão

  <comandos_falso>

  – Quando um comando de decisão é executado, apenas UMA das listas de comandos é executada: a **verdadeira** ou a **falsa**

  – **Exemplo** (decisões encadeadas)**:**

```
se A>12 então
    se B>18  então
        A ← 2;
    senão
        A ← 4;
```

Problema 7 – Bom dia, boa tarde, boa noite!

- **Sabemos que de 6 da manhã a meio-dia desejamos 'Bom dia!', assim como de meio-dia a 6 da noite desejamos 'Boa tarde!' e daí até o amanhecer desejamos 'Boa noite!'.**
- **Faça um programa que pede ao usuário para informar a hora certa (de 0 a 24) e exibe uma das saudações acima como resultado.**
- **Solução:**

```
Variáveis
    hora : inteiro;
Início
    escrever 'Informe a hora certa:';
    ler hora;
    se hora>=6 E hora<12 então
        escrever 'Bom dia';
    senão
        se hora>12 E hora<18 então
          escrever 'Boa tarde!';
        senão
          escrever 'Boa noite!';
Fim.
```

Problema 8 – 3 números em ordem

- **Escreva um programa que leia 3 números reais obrigatoriamente diferentes e imprima-os em ordem crescente.**
- **Solução:**

```
Variáveis
    num1, num2, num3 : real;
Início
    ler_numeros;
    se num1<num2 E num1<num3 então
        se num2<num3 então
            escrever num1, num2, num3;
        senão
            escrever num1, num3, num2;
    senão
        se num2<num1 E num2<num3 então
            se num1<num3 então
                escrever num2, num1, num3;
            senão
                escrever num2, num3, num1;
        senão
            se num1<num2 então
                escrever num3, num1, num2;
            senão
                escrever num3, num2, num1;
Fim.
Módulo ler_numeros; //(...implementar...)
```

Problema 9 – Quantos números pares?

- **Escreva um programa que leia do usuário 3 números pares e informe quantos são pares.**
- **Solução:**

```
Variáveis
    num1, num2, num3, contador : inteiro;
Início
    leia_numeros;
    contador ← 0;
    se num1%2 = 0 então
        contador ← contador + 1;
    se num2%2 = 0 então
        contador ← contador + 1;
    se num3%2 = 0 então
        Contador ← contador + 1;
    escrever contador, ' números são pares';
Fim.

Módulo ler_numeros;
[[
   escrever 'Informe três inteiros:';
   ler num1, num2, num3;
]];
```

Problema 10 – Jogo do par ou ímpar

- **No jogo do par ou ímpar, um jogador pede par e coloca um número inteiro. Um outro jogador pede ímpar e faz o mesmo. O jogador vencedor é aquele que tiver acertado se a soma é par ou ímpar.**
- **Faça um programa que requisita dois números inteiros e informa se a soma é par ou ímpar.**

Solução:

```
Variáveis:
    dedos1, dedos2 : inteiro;
Início
    escrever 'Número do jogador 1:';
    ler dedos1;
    escrever 'Número do jogador 2:';
    ler dedos2;
    se (dedos1+dedos2)%2=0 então
      escrever 'O resultado é PAR!';
    senão
      escrever 'O resultado é ÍMPAR!';
    Fim.
```

Comando caso

- **Utilizado quando uma mesma variável (ou expressão) precisar ter vários valores testados;**
- **Sintaxe do comando:**

```
caso <variável ou expressão> seja
    <valor1> : <comando1> ;
    <valor2> : <comando2> ;
    <valor3> : <comando3> ;
    .
    .
    .
    <valorN> : comandoN> ;
    senão
        <comando_senao> ;
fimcaso;
```

- **Testando intervalos com o comando caso:**
  - 1,3 – **valores inteiros 1, 2 e 3**
  - ‘A’ .. ‘Z’ – **caracteres de ‘A’ a ‘Z’**
  - 1,3, 12..21 – **inteiros 1 e 3 e inteiros de 12 a 21**

Problema 11 – Mestre Jedi

- **Um Jedi é um ser especial dotado de capacidade de controlar a Força (vide *Star Wars*). Assim que um Jedi é descoberto, um professor o ensina o caminho da Força para sua evolução em níveis:**

Nível	Título
0	Descoberto
1	Padawan
2	Cavaleiro Jedi
3	Mestre Jedi

Faça um programa que leia um número inteiro informado pelo usuário representando seu nível Jedi e escreva na tela seu respectivo título.

- **Solução**:

```
Variáveis:
    nivel : inteiro;
Início
    escrever 'Informe seu nível Jedi:';
    ler nivel;
    caso nivel seja
        0 : escrever 'Descoberto';
        1 : escrever 'Padawan';
        2 : escrever 'Cavaleiro Jedi';
        3 : escrever 'Mestre Jedi';
        senão
            escrever 'Nível inválido';
    fimcaso;
Fim
```

Bloco de comandos

- **Pela sintaxe dos comandos** SE **e** CASO**, um comando é executado de acordo com o resultado de uma expressão (verdadeira ou falsa) ou valor de uma variável.**
- **No entanto, às vezes é necessário que se execute mais de um comando. Nesse caso, é necessário criar um bloco de comandos.**
- **Sintaxe:**

```
[[
<comando1>;
<comando2>;
.
.
.
<comandoN>;
]]
```

- **Exemplo:**

```
se x>3 então
    [[
    a ← 2;
    b ← 4;
    ]]
```

Problema 12 – Assinaturas

- **A tabela seguinte mostra o código fictício de várias revistas e o valor de sua assinatura:**

Código	Revista	Assinatura Anual
101	Veja	R$ 297,00
102	Exame	R$ 184,00
103	Info	R$ 107,00
104	Coleção Info	R$ 139,86
105	Você S/A	R$ 90,00

- **Faça um programa que lê do usuário o código de uma revista como inteiro e escreve o nome da revista e o valor da assinatura anual. Solução:**

```
Variáveis
   codigo : inteiro;
Início
   escrever 'Diga o código da revista:';
   ler codigo;
   caso codigo seja
      101 : escrever 'Veja R$ 297,00';
      102 : escrever 'Exame R$ 184,00';
      103 : escrever 'Info R$ 107,00';
      104 : escrever 'Col. Info R$ 139,86';
      105 : escrever 'Você S/A R$ 90,00';
      senão
         escrever 'Código inválido';
   fimcaso;
Fim.
```

Problema 13 – Categorias do judô

- As competições do judô são divididas em categorias (ou classes) de acordo com o peso.
- Faça um programa que leia do usuário o peso de um atleta e exiba sua classe.
- Solução:

CLASSES	Senior (anos)
Super Ligeiro	até 44
Ligeiro	+ 44 a 48
M. Leve	+ 48 a 52
Leve	+ 52 a 57
M. Médio	+ 57 a 63
Médio	+ 63 a 70
M. Pesado	+ 70 a 78
Pesado	+ de 78
S. Pesado	.

```
Variáveis
   peso : inteiro;
Início
   escrever 'Informe o peso:';
   ler peso;
   se peso<=44 então
      escrever 'Super ligeiro';
   senão
      caso peso seja
         45..48 : escrever 'Ligeiro';
         49..52 : escrever 'M. Leve';
         53..57 : escrever 'Leve';
         58..63 : escrever 'M. Médio';
         64..70 : escrever 'Médio';
         71..78 : escrever 'M. Pesado';
         senão escrever 'Pesado';
      fimcaso;
Fim.
```

Problema 14 – Campeões do Mundo

- **Faça um programa que leia um ano e informe quem é o campeão mundial no final do ano.**

Local - Ano	Campeão
Uruguai - 1930	Uruguai
Itália - 1934	Itália
França - 1938	Itália
Brasil - 1950	Uruguai
Suíça - 1954	Alemanha
Suécia - 1958	Brasil
Chile - 1962	Brasil
Inglaterra - 1966	Inglaterra
México - 1970	Brasil

Local - Ano	Campeão
Alemanha - 1974	Alemanha
Argentina - 1978	Argentina
Espanha - 1982	Itália
México - 1986	Argentina
Itália - 1990	Alemanha
EUA - 1994	Brasil
França - 1998	França
Coréia e Japão – 2002	Brasil

```
Variáveis
    ano : inteiro;
Início
    escrever 'Informe o ano:';
    ler ano;
    se ano>=2002 então
       escrever 'Brasil';
    caso ano seja
       1930..1933 : escrever 'Uruguai';
       1934..1937 : escrever 'Itália';
       1938..1949 : escrever 'Itália';
       1950..1953 : escrever 'Uruguai';
       1954..1957 : escrever 'Alemanha';
       //(...)
       1998..2001 : escrever 'França';
    fimcaso;
Fim.
```

Procurando erros em programas

```
Variáveis
   raio : real;
Início
   escrever 'Informe o raio:';
   ler raio;
   area = 3,14 * QUAD(raio);
   se area > 10 então
      escrever 'Area maior que 10';
Fim.
```

```
Variáveis
   x,y,z : inteiro;
Início
   x ← 9;
   y ← 2;
   z ← x/y;
   x ← y+1;
   Y ← x/2;
   se x>y então
      z ← x + y – z;
   senão
      z ← x + y;
   se z<0 então
      escrever 'Negativo';
   se z>0 então
      escrever 'Positivo';
Fim.
```

Procurando erros em programas

```
Módulo teste_crescente4;
    [[
    se n1>=n2 E n3>=n4 então
       escrever 'Crescente';
    senão
       escrever 'Não crescente';
    ]];

Módulo contando4;
    [[
    se procurado=nome1 então
       contador ← 1;
    se procurado=nome2 então
       contador ← 2;
    se procurado=nome3 então
       contador ← 3;
    se procurado=nome4 então
       contador ← 4;
    ]];
```

Busca em números ordenados

```
Módulo busca_ordenada3;
    [[
    se x=n1 OU x=n2 OU x=n3 então
       achou←VERDADEIRO;
    senão
       achou←FALSO;
    ]];

Módulo busca_ordenada5;
    [[
    achou ← FALSO;
    se x=n3 então
       achou ← VERDADEIRO;
    se x<n3 então
       se x=n1 OU x=n2 então
          achou ← VERDADEIRO;
    se x>n3 então
      se x=n4 OU x=n5 então
         achou ← VERDADEIRO;
    ]];
```

Busca em números ordenados

```
Módulo busca_ordenada7;
    [[
    achou ← VERDADEIRO;
    se x>n4 então
        [[
        se x>n6 E x<>n7 então
            achou ← FALSO;
        se x<n6 E x<>n5 então
            achou ← FALSO;
        ]]
    se x<n4 então
        [[
        se x<n2 E x<>n1 então
            achou ← FALSO;
        se x>n2 E x<>n3 então
            achou ← FALSO;
        ]]
    ]];
```

Busca em números ordenados

```
Módulo busca_ordenada9;
    [[
    achou ← VERDADEIRO;
    se x<n5 então
        [[
        se x<n2 E x<>n1 então
            achou ← FALSO;
        se x>n2 E (x<>n3 E x<>n4) então
            achou ← FALSO;
        ]]
    se x>n5 então
        [[
        se x<n8 E (x<>n6 E x<>n7) então
            achou ← FALSO;
        se x>n8 E x<>n9 então
            achou ← FALSO;
        ]]
    ]];
```

Repetições

- São também chamadas de **loops** ou **laços**;
- É uma estrutura que permite executar um trecho do algoritmo várias vezes;
- Há dois tipos básicos de repetições:
  - Com teste no início (**enquanto**);
  - Com teste no final (**repita**);

- Comando Repita

```
repita
    <comando1>;
    <comando2>;
    .
    .
    .
    <comandoN>;
até <expressão>;
```

- Comando Enquanto

```
enquanto <expressão> faça
    <comando>
```

Problema 15 – Quadrado perfeito

- **Faça um programa que leia do usuário números inteiros e exiba na tela sua raiz. O programa deve terminar quando for digitado um número que não é um quadrado perfeito.**
- **Solução:**

```
Variáveis
    num : inteiro;
    raiz_real : real;
Início
    repita
        ler num;
        imprimir RAIZ(num);
        raiz_real ← RAIZ(num);
        num ← QUAD(num);
    até num<>raizNum;
Fim.
```

Problema 16 – Pares em um intervalo

- **Faça um programa que leia do usuário um limite inferior e um limite superior para um intervalo de números inteiros e exibe os números pares nesse intervalo fechado. IMPORTANTE: assuma que o usuário informa corretamente um limite inferior MENOR OU IGUAL a um limite superior. Solução:**

```
Variáveis
 limite_inferior,limite_superior,i: inteiro;
Início
   ler_limites;
   i ← limite_inferior;
   enquanto i<=limite_superior faça
      [[
      se i%2=0 então escrever i;
      i ← i+1;
      ]]
Fim.
Módulo ler_limites;
[[
   escrever 'Informe limite inferior:';
   ler limite_inferior;
   escrever 'Informe limite superior';
   ler limite_superior;
]];
```

Problema 16 – Números pares em um intervalo

Solução Melhorada:

```
Variáveis
 limite_inferior,limite_superior,i:inteiro;
Início
   ler_limites;
   se limite_inferior % 2 = 0 então
      i ← limite_inferior;
   senão
      i ← limite_inferior+1;
   enquanto i<=limite_superior faça
      [[
      escrever i;
      i ← i +2;
      ]]
Fim.
Módulo ler_limites; // aperfeiçoando...
[[
 escrever 'Informe limite inferior:';
 ler limite_inferior;
 escrever 'Informe limite superior:';
 ler limite_superior;
 se limite_inferior > limite_superior então
    [[
    i ← limite_inferior;
    limite_inferior ← limite_superior;
    limite_superior ← i;
    ]]
]];
```

Problema 17 – Qual o maior? E o menor?

- **Faça um programa que leia do usuário 100 números reais e informe qual o maior e qual o menor deles.**
- **Solução:**

```
Variáveis
    num, menor, maior : real;
    i : inteiro;
Início
    ler_inteiro;
    menor ← num;
    maior ← num;
    i ← 1;
    repita
        ler_inteiro;
        se num>maior então maior←num;
        se num<menor então menor←num;
        i ← i+1;
    até i>100;
Fim.
Módulo ler_inteiro;
[[
    escrever 'Diga um número inteiro';
    ler num;
]];
```

Problema 18 – sinuca

- **Uma variante popular da sinuca envolve o uso de 4 bolas pretas e 4 vermelhas (além da branca). Nesse jogo, ganha a primeira equipe que derrubar as 4 bolas de sua cor.**
- **Faça um programa que lê do usuário um caracter 'V' ou 'P' respectivamente quando uma bola vermelha ou preta for "encaçapada" e que avisa quando uma equipe vencer.**

```
Variáveis
    carac : caractere;
    vermelhas, pretas : inteiro;
Início
    vermelhas ← 4;
    pretas ← 4;
    repita
        ler carac;
        se carac='V' então
            vermelhas ← vermelhas-1;
        se carac='P' então
            pretas ← pretas-1;
    até vermelhas=0 OU pretas=0;
    se vermelhas=0 então
        escrever 'Vermelhas venceram!';
    senão
        escrever 'Pretas venceram!';
Fim.
```