Switch (Case)

- Equivale a um conjunto de *if 's* encadeados, porém mais estruturado.

```
switch(expressao_ordinal)
{
    ordinal_1: bloco1;
               break;
    ordinal_2: bloco2;
               break;
    default: diretiva_default;
}
```

Exemplo - *switch*

```
public class SwitchExemplo
{
      public static void main(String[] args)
      {
            switch (args[0].charAt(0))
            {
                  case 'a':
                  case 'A':System.ou.println("Vogal A");
                           break;
                  case 'e':
                  case 'E':System.ou.println("Vogal E");
                           break;
                  case 'i':
                  case 'I':System.ou.println("Vogal I");
                           break;
                  case 'o':
                  case 'O':System.ou.println("Vogal O");
                           break;
                   case 'u':
                   case 'U':System.out.println("Vogal U");
                           break;
                default:System.out.println("A letra não é vogal.");
            }
      }
}
```

Observações do exemplo

- O ponto de início de execução é sempre um *case,* e a execução pára apenas quando um *break* é encontrado.
- Se o valor da expressão ordinal não for encontrado nos *cases* a diretiva default é executada.
- A diretiva default é opcional.

- *Obs:* Um resultado ordinal é aquele que pertence a um conjunto que se sabe exatamente quem é o elemento anterior e o posterior. Como no conjunto dos inteiros, caracteres...

Estruturas de Repetição (loop)

Estruturas de Repetição

- Essas estruturas fazem com que uma instrução, ou bloco execute repetidamente, enquanto uma expressão seja verdadeira.

Existem 2 tipos:

- Estruturas de repetição incondicional (simples): Repete um número específico de vezes. Estrutura *for*
- Estruturas de repetição condicional: São estruturas de repetição que o controle é feito pela avaliação de expressões condicionais. Ou seja, o número de repetições é indeterminado na fase de programação, será conhecido durante a execução. Estruturas ***while*** e ***do – while***

O que uma estrutura de repetição deve ter ?

- Critério de parada. Através de uma expressão que será avaliada como sendo *true* ou *false*.
- Inicialização das variáveis de controle. Inicializar as variáveis que tem relação com a expressão do critério de parada.
- Atualização das variáveis de controle dentro do laço. Se as variáveis de controle não forem atualizadas dentro do laço, a expressão pode acabar nunca sendo avaliada como false, tornando assim o laço infinito.
- E as instruções que serão executadas repetidas vezes.

Estrutura *for*

- Formato:

```
for (inicialização; condição de execução;
     Incremento/decremento) Bloco de Instruções;
```

- Inicialização: É usado para dar valor inicial a variável de controle (contador).
- Condição de execução: É uma expressão lógica que determina a execução associada ao for, geralmente utilizando a variável de controle.
- Incremento/Decremento: Determina como a variável de controle (ou outras variáveis também) será alterada a cada iteração do for.
- A inicialização é feita apenas antes da primeira iteração.
- A execução é encerrada quando a condição de execução for avaliada como falsa

Exemplo *for*

- Programa que soma os **n** primeiros números que entraram como argumento.

```
public class ExemploFor
{
   public static void main(String[] args)
   {
      int n=Integer.parseInt(args[0]);
      int i, soma;
      //(inicialização; condição; incremento)
      for(i=0, soma=0;    i<=n;      i++)
      {
         System.out.print("Soma = "+soma+" + "+i);
         soma+=i;
         System.out.println(" = "+soma+"\n");
       }
   }
}
```

Na chamada da função no console:

```
>java ExemploFor 3
Soma = 0 + 1 = 1
Soma = 1 + 2 = 3
Soma = 3 + 3 = 6
```

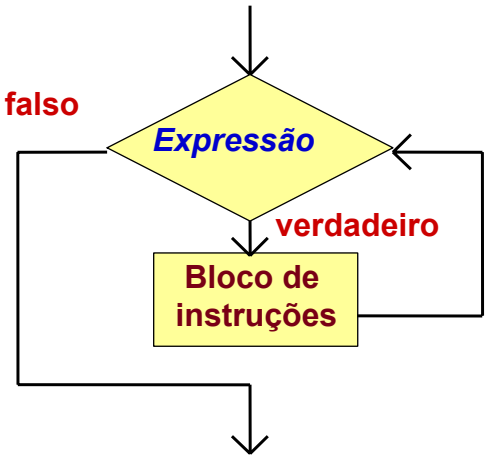
Estrutura *while*

- O Corpo do loop pode ser uma simples instrução, um bloco de instruções ou nenhuma.
- Na estrutura while, a expressão é avaliada no início (antes) da execução do laço. Logo, se a expressão for falsa ao ser avaliada pela primeira vez, o laço não será executado.

```
while( Expressão )
{
    .
    .          // Corpo do loop
    .
}
```

Estrtura *while*

- Quando a expressão é testada, e avaliada como falsa , o laço é encerrado e o controle passa para a instrução que se segue ao corpo do laço.



Exemplo: *while*

```
int contador;    // Declara variável de controle
contador = 1;    // Inicializa variável de controle

while (contador <= 4)   // Testa a expressão
{
   // Repete instruções
   System.out.println("Contador = " + contador);
   contador++;       // Atualiza variável de controle
}
System.out.println("Terminado");
```

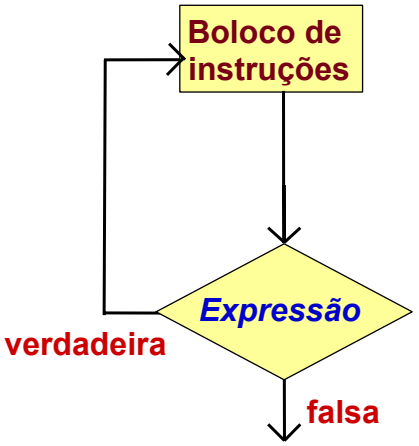
- Critério de Parada é: contador<=4.
- Variável de Controle é: contador.
- Atualização da variável de controle é feita por: contador++;

Estrutura *do-while*

- O Corpo do loop pode ser uma simples instrução, um bloco de instruções ou nenhuma.
- Na estrutura do-while, a expressão é avaliada no final da execução do laço. Logo, o laço será executado pelo menos UMA vez.

```
do
{
    .
    .          // Corpo do loop
    .
}while( Expressão )
```

Estrutura *do-while*



Exemplo: *do-while*

```
import java.io.*;
public class exemploDoWhile
{
      public static void main(String[] args)throws IOException
      {
            BufferedReader in;
            in = new BufferedReader
                              (new InputStreamReader(System.in));

            String aux;
            int contador=0;
            System.out.println("Quantidade de vezes que uma
                        tecla diferente de A foi pressionada:");

            do
            {
                  System.out.println("Contador = "+contador);
                  System.out.print("Digite uma letra e ENTER: ");
                  aux = in.readLine();
                  contador++;
            }while (aux.charAt(0)!='A');//Criterio de Parada
      }
}
```

Exemplo: *do - while*

O exemplo anterior é um programa que conta quantas letras diferentes de A o usuário digitou.

Na chamada da função no console:

```
>Quantidade de vezes que uma tecla diferente de A foi
pressionada:
Contador = 0
Digite uma letra e ENTER: W
Contador = 1
Digite uma letra e ENTER: E
Contador = 2
Digite uma letra e ENTER: R
Contador = 3
Digite uma letra e ENTER: A
```

- O critério de parada é: aux.charAt(0)!='A'.
- A variável de controle é: aux.
- A atualização da variável de controle é feita por:

```
aux = in.readLine();
```