Loops

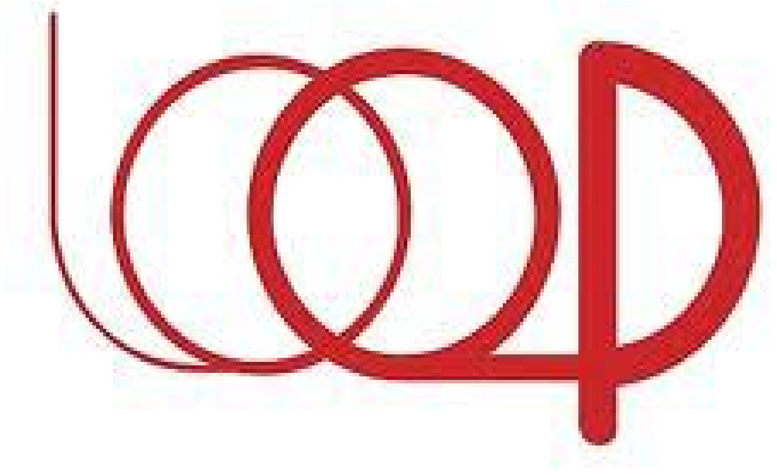
- Loops are the way of repeating lines of codes until loop condition is met.!!!!
- Loops in java :

➢ While loop

➢ Do-while loop

➢ For loop

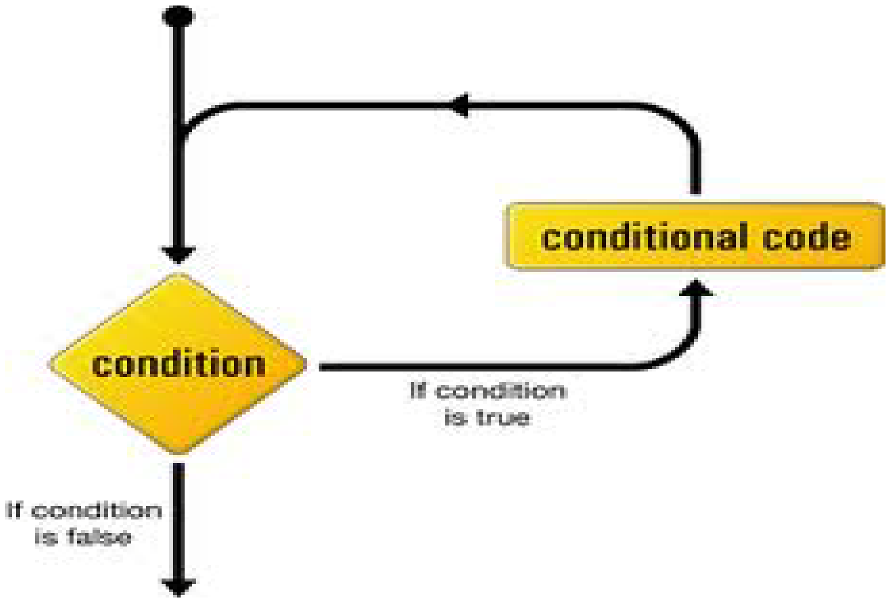
➢ Enhanced for loop



While loop

- While loop repeats a block of code until the condition is true

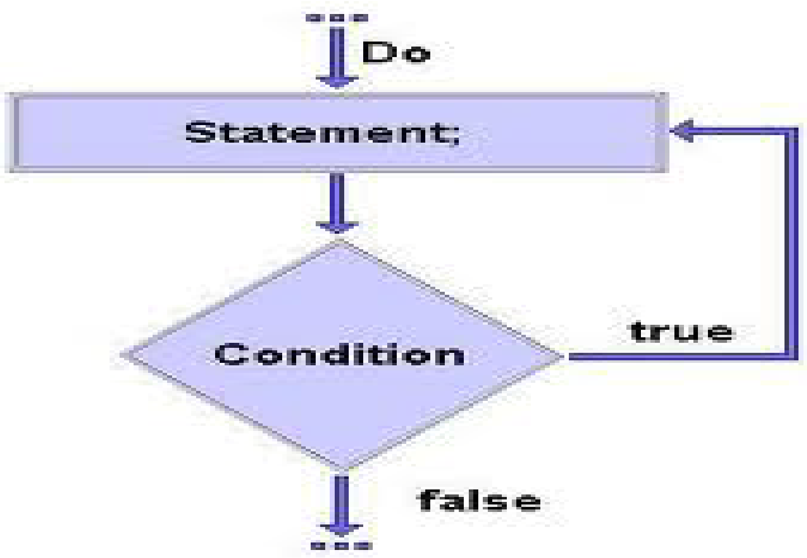
```
While (condition)
{
// block of codes
}
 //where condition is nothing but the Boolean expression
```



Do-while loop

- Do-while loop is similar to the "while loop" but the only difference is that, in this the loop block is guaranteed to run at least one time!!!!

```
do
{
//all codes here
}
While (condition)
```



//since condition appears at the end, therefore the code block executes at least one

For loop

- In for loop the initialization, condition checking and updating of loop element is done in a single line.

```
For(initialization; condition; update)
{
// codes
}
```

- The initialization step is executed first, and only once. This step allows you to declare and initialize any loop control variables.
- Next, the Boolean expression is evaluated. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute.
- After the body of the for loop executes, the flow of control jumps back up to the update statement. This statement allows you to update any loop control variables.

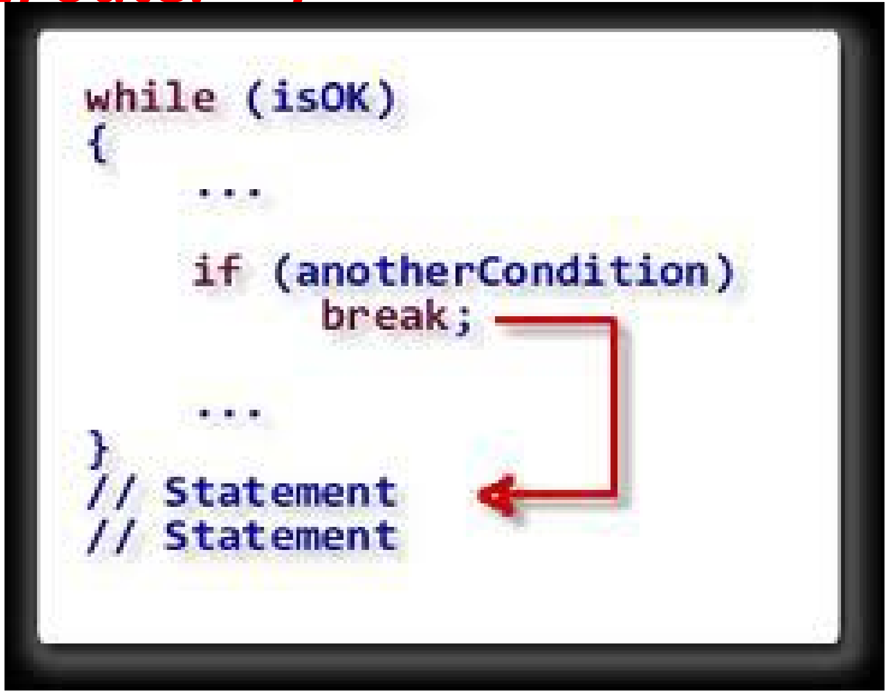
Nesting of loops

- The placing of one loop inside the body of another loop is known as Nesting of loops.
- While working with nesting loops the outer loop will change only when inner loop is completely finished.

```
For (int outer =0; outer<5 ; outer++)
{
For (int inner= 0; inner <3; inner++)
{
System.out.println("outer is " + outer + "inner is" + inner);
}//inner loop ends
}//outer loop ends
```

Break and Continue statements

- Break and Continue statements are used to change the normal flow of compound statement.
- The break statement immediately jumps to the end of the compound statement.
- The continue statement immediately jumps to the next iteration of the compound statement.
- **for (int outer=0; outer< 12; outer++)**
- {
- **if(outer ==3)**
- **continue;**
- System.*out*.*println(outer);*
- **if(outer ==7)**
- **break;**
- }



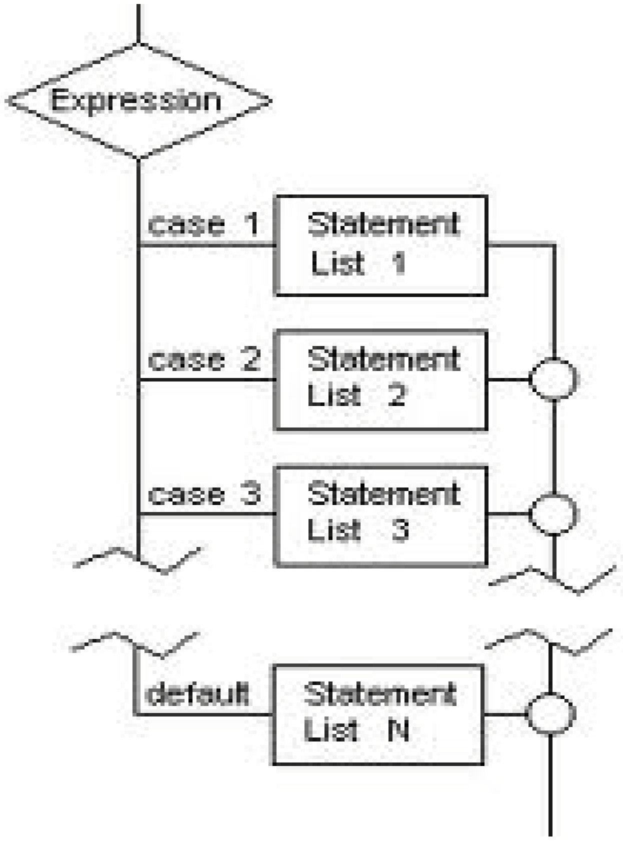
```
while (isOK)
{
        ...

        if (anotherCondition)
              break;

        ...
}
// Statement
// Statement
```

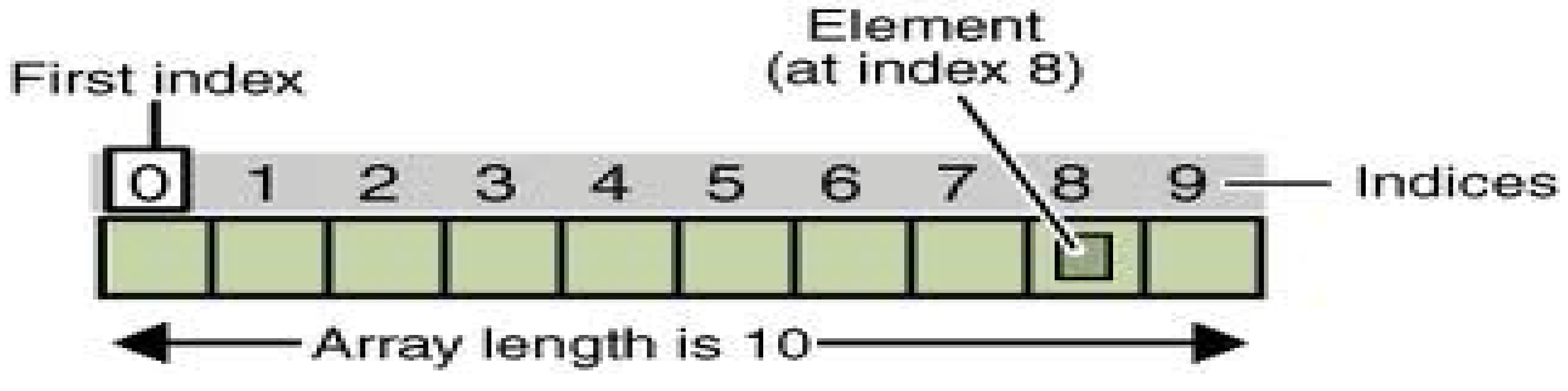
Switch statement

- Switch statement is the shorthand for multiple 'if-else' statement, which allow us to choose a single path from a number of execution path.
- Switch statement works with char, short, byte, int and String.
- **switch(x)**
- {
- **case 1:**
- System.*out*.println(*"case1");*
- **break;**
- **case 2:**
- System.*out*.println(*"case2");*
- **break;**
- **case 3:**
- System.*out*.println(*"case 3");*
- **break;**
- **Default**
- **System.out.println("default case ,**
- }



Arrays

- Arrays are the collection of similar datatypes.
- Each variable in an array is known as ‘array element.
- Each variable of array is referenced by a particular integer number which is known as ‘array index’.
- The total number variables in array decide the length of the array.



Declaration and initialization of array

- In java array is an object, therefore it is declared and initializes like an object.
- Declaration of array variable:

int[] array;

- Constructing the array:

new int[(length of the array)];

- Assigning array to array variable:

array = new int[(length of the array)];

- Initialization of array:

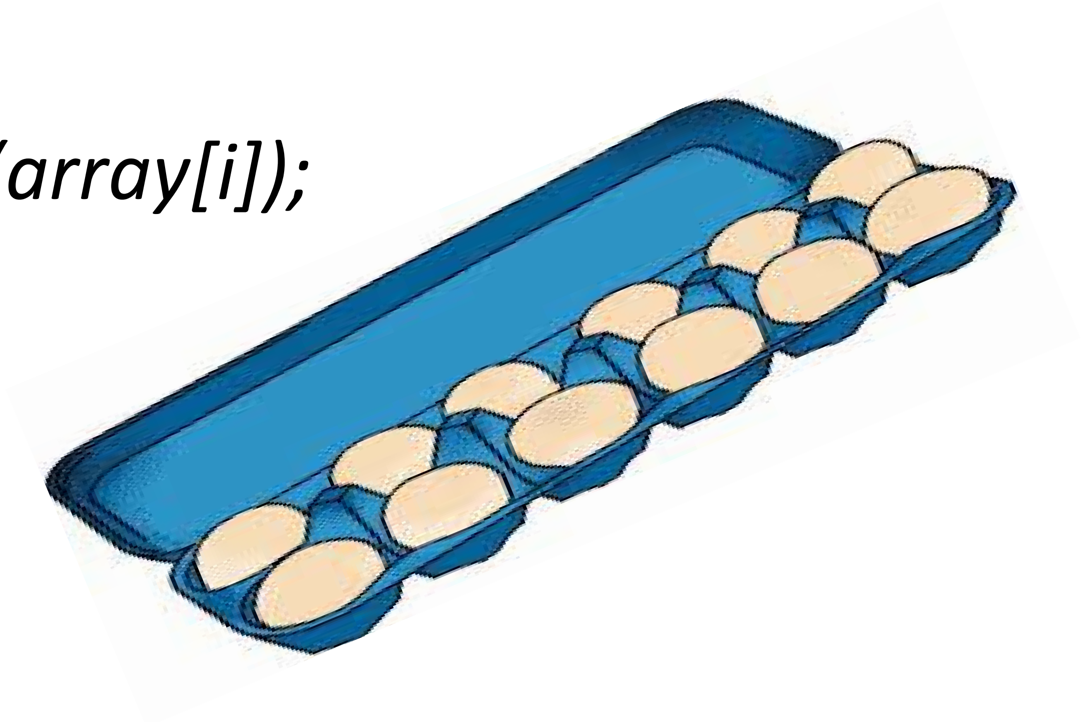
Array[0] = 34;

- Declaration and initialization in single line:

Int[] array = { 34, 56, 7, 23, 34,};

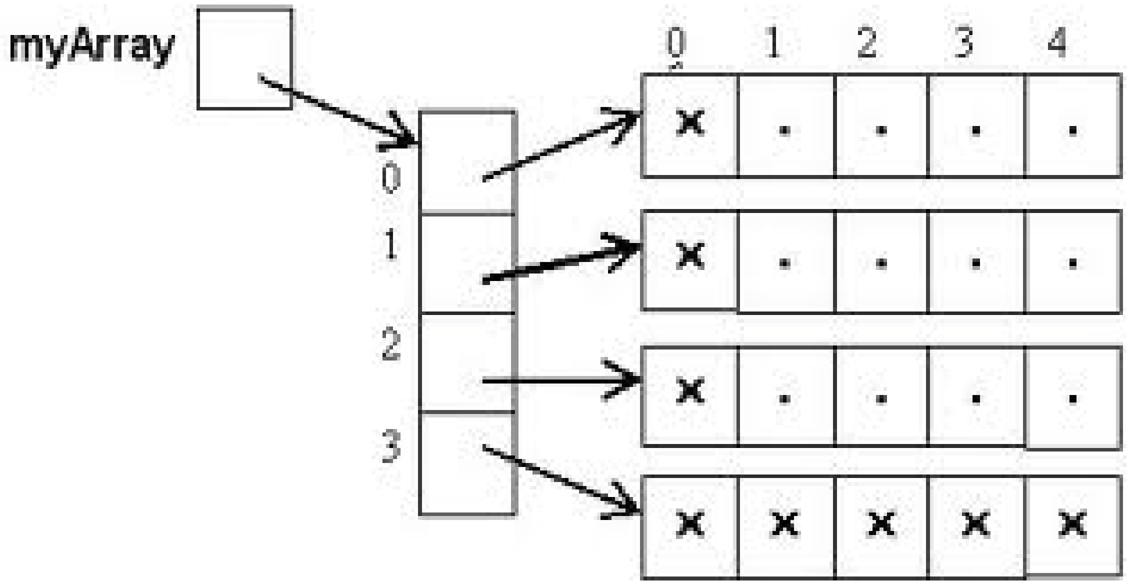
Initialization of array using loop

- Arrays can be initialized by using loops.
- **int [] array = new int[34];**
- **for (int i=0;i<array.length;i++)**
- {
- array[i]=i;
- System.*out.println(array[i]);*
- }



Multidimensional arrays

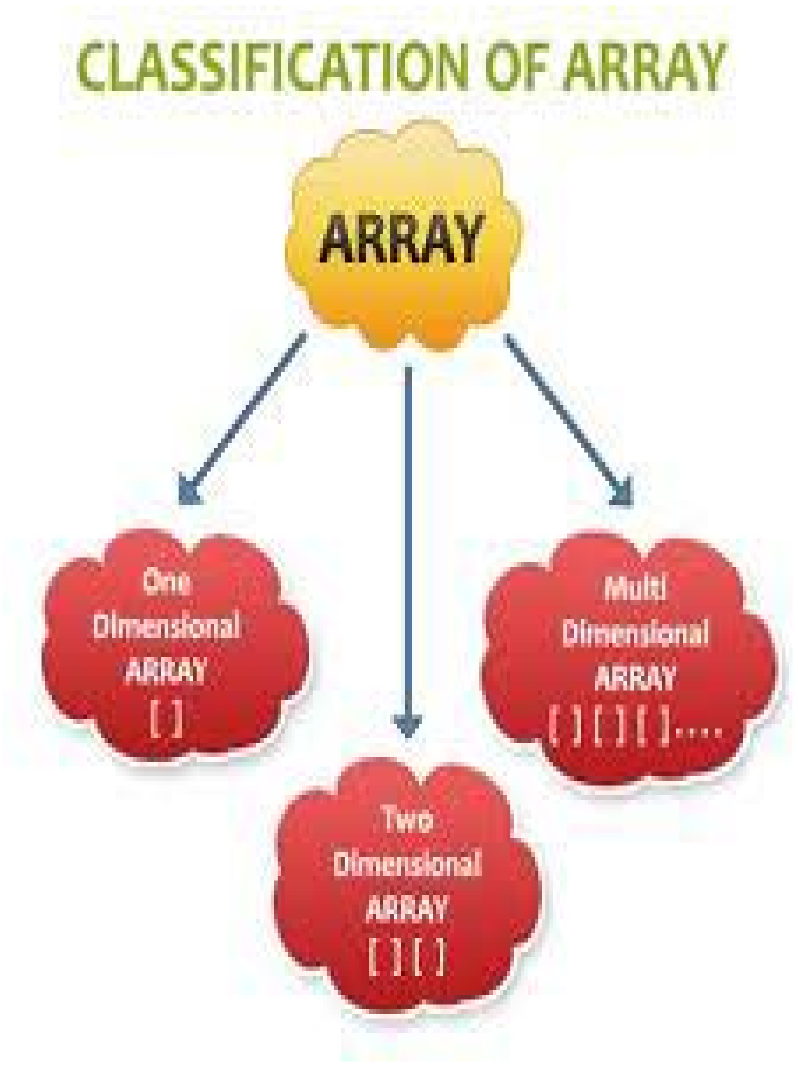
- Multi-dimensional arrays are nothing but the "array of arrays" where each element represents a single dimensional array.



- Here 'myArray' is a 2-d array, whose each element contains a single dimensional array.

Initialization and declaration of 2-d array

- **for(int i=0;i<myArray2.length;i++)**
- {
- **for (int j= 0; j<myArray2[i].length;j++)**
- {
- myArray2[i][j]= j;
- }
- }
- **for (int i=0; i<myArray2.length;i++)**
- {
- **for (int j=0 ; j<myArray2[i].length; j++)**
- {
- System.*out.print(myArray2[i][j] + "\t");*
- //System.out.print("\t");
- }
- System.*out.println();*
- }


CLASSIFICATION OF ARRAY
ARRAY
One Dimensional ARRAY [ ]
Two Dimensional ARRAY [ ] [ ]
Multi Dimensional ARRAY [ ] [ ] [ ]....

Program #1 : pyramid of Stars

- **public class Test {**

- **public static void main(String [] args)**
- {

- **for (int outer =1; outer<=5; outer++)**
- {
- **for (int inner = 0; inner<outer; inner++)**
- {
- System.*out.print(“*”);*
- }
- System.*out.println();*
- }
- }
- }

Enhanced for loop

- “Enhanced for loop” is introduced in java 5, in order to simply the way to iterate a collection or array.
- In this the loop continues till the last element of the collection or array.
- **for (int y : array)**
- {
- System.*out.print(y);*
- }

Enhanced for loop for 2-d array

- **for(int[] x : myArray2)**
- {
- **for (int y : x)**
- {
- System.*out.print(y + "\t");*
- }
- System.*out.println();*
- }