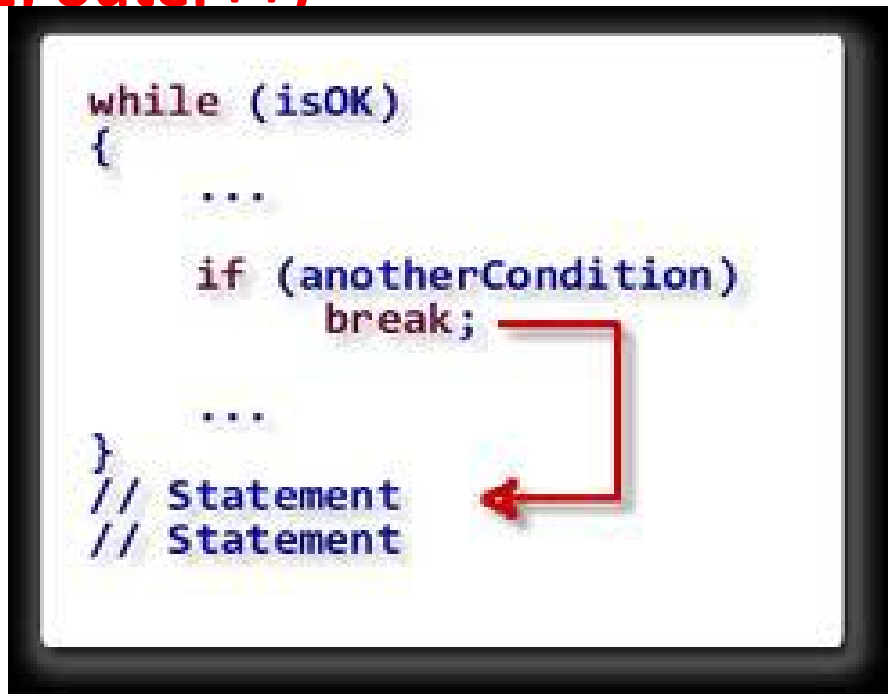
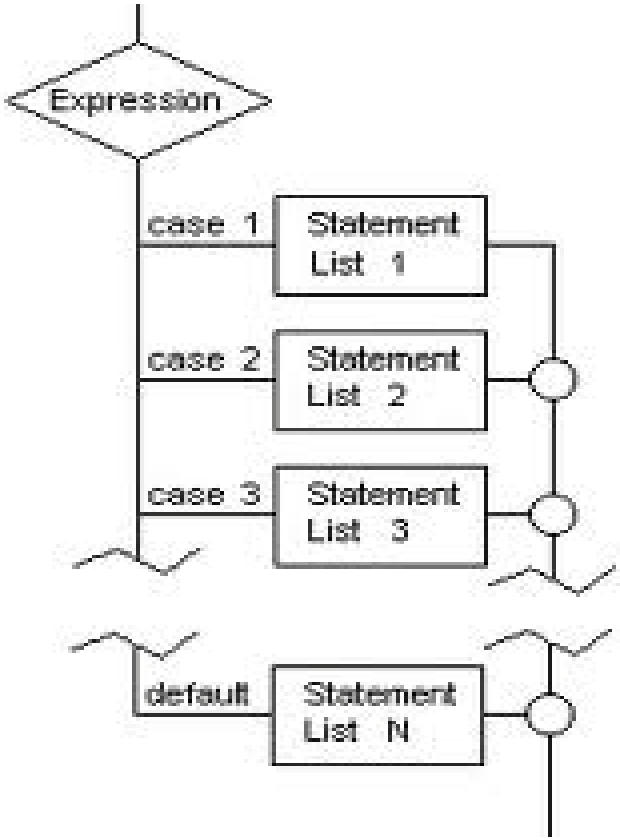
Break and Continue statements

- 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;**
- }



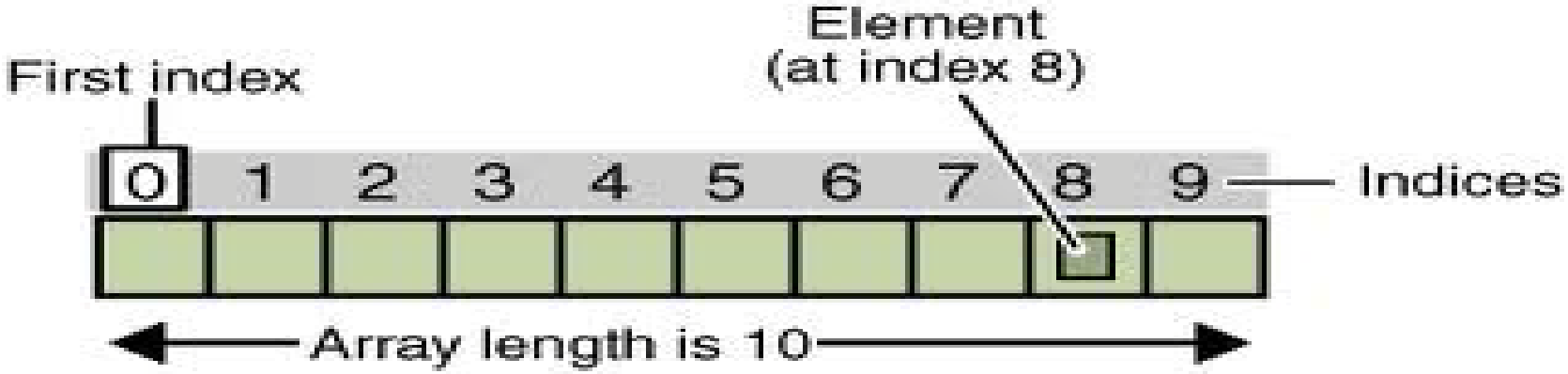
Switch statement

- 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:

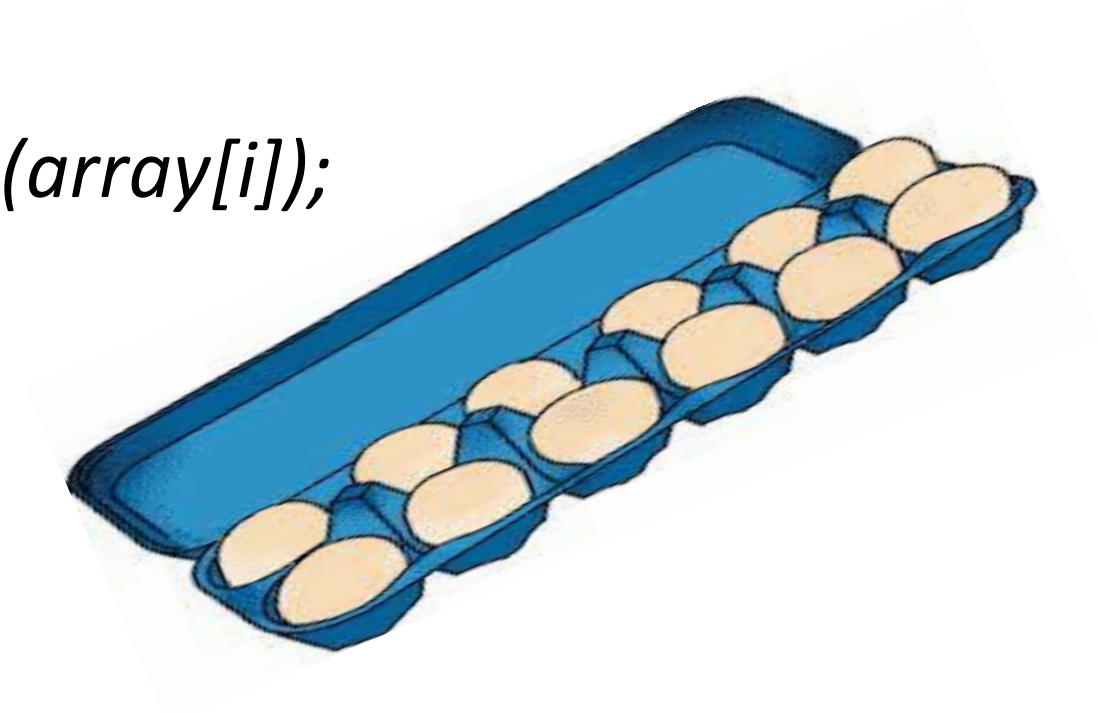
Array[0] = 34;

- Declaration and initialization in single line:

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

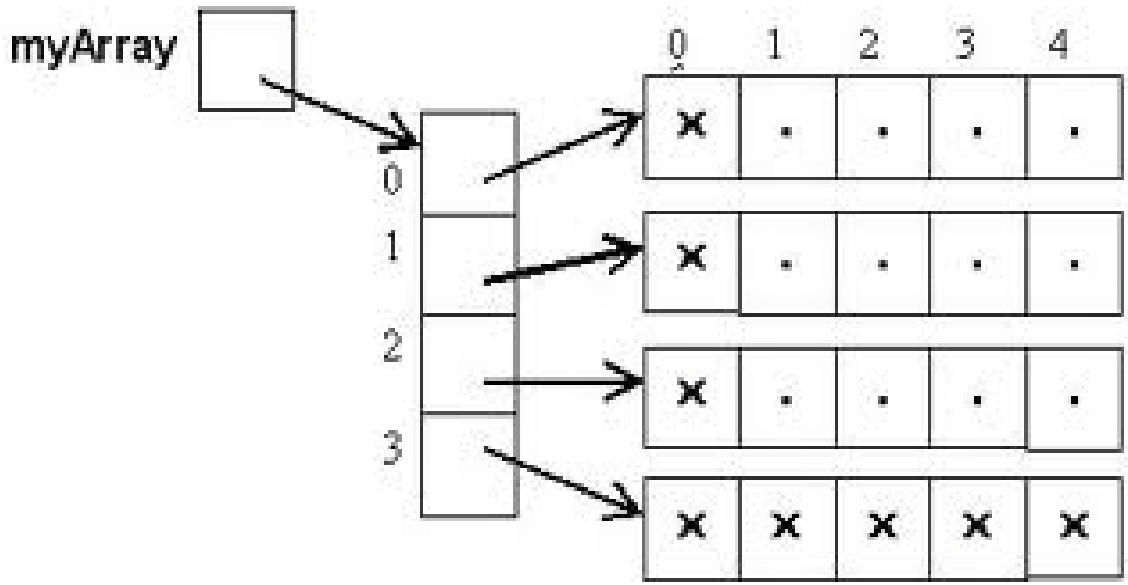
Initialization of array using loop

- 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

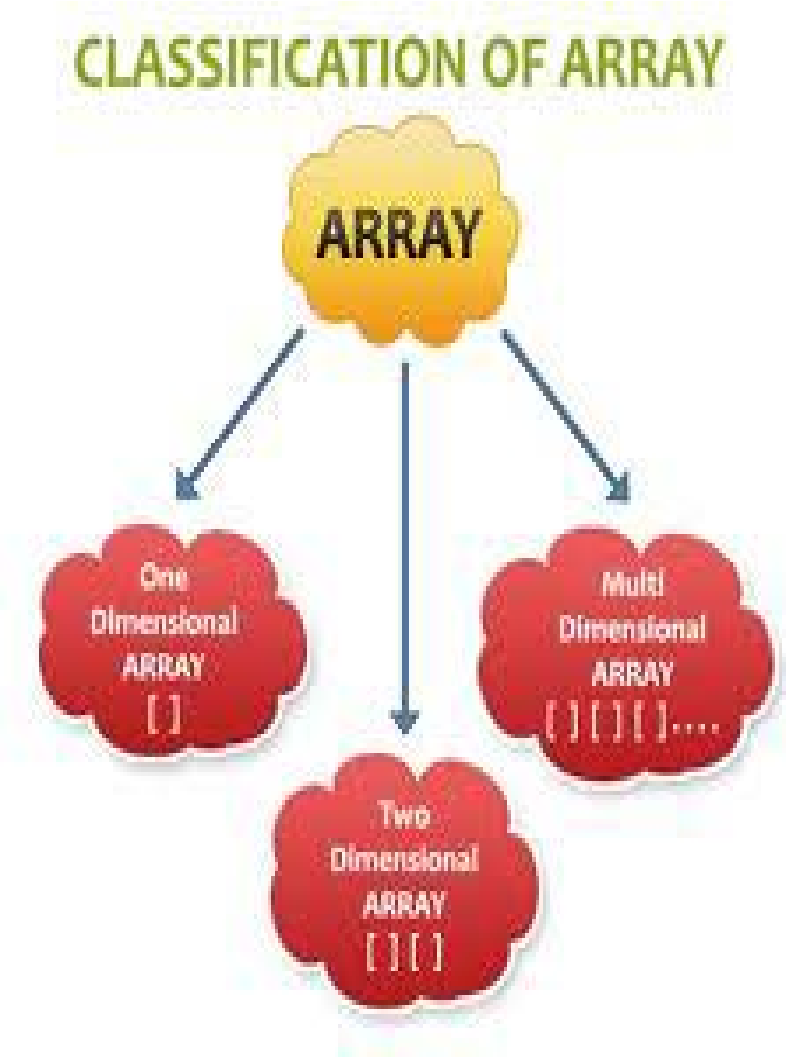
- Multi-dimensional arrays are nothing but the "array of arrays" where each element represents a single dimensional array.



- 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();*
- }



Enhanced for loop

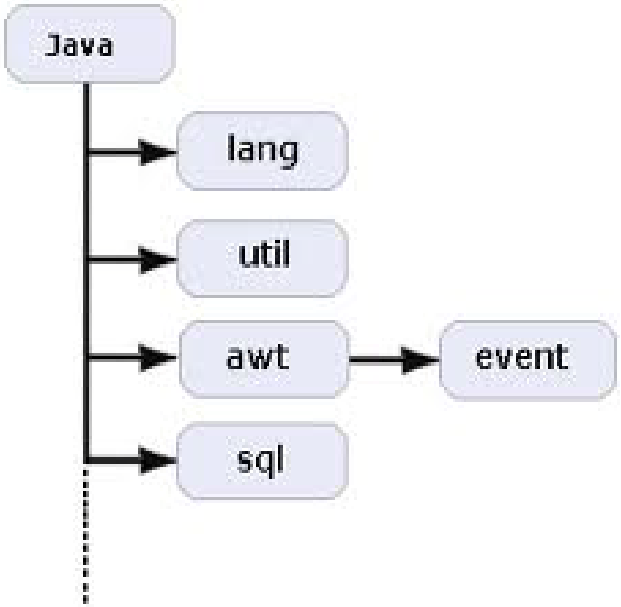
- “Enhanced for loop” is introduced in java 5, in order to simply the way to iterate a collection or array.
- In this the loop continues till the last element of the collection or array.
- **for (int y : array)**
- {
- System.*out.print(y);*
- }

Enhanced for loop for 2-d array

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

Packages in java

- A package is the grouping of related types providing access protection and named space management.
- Packages are created by using the keyword “package” and it should be first line of the source file.
- In order to use classes of other packages we have to use “import” statements.



Using import to access packages

- **package package1;**

- **public class demoTest {**
- **public void go()**
- {
- System.*out.println("in different package");*
- }
- }
- **import package1.*;**
- **public class Test {**
- **int x = 6;**
- **public static void main(String[] args)**
- {

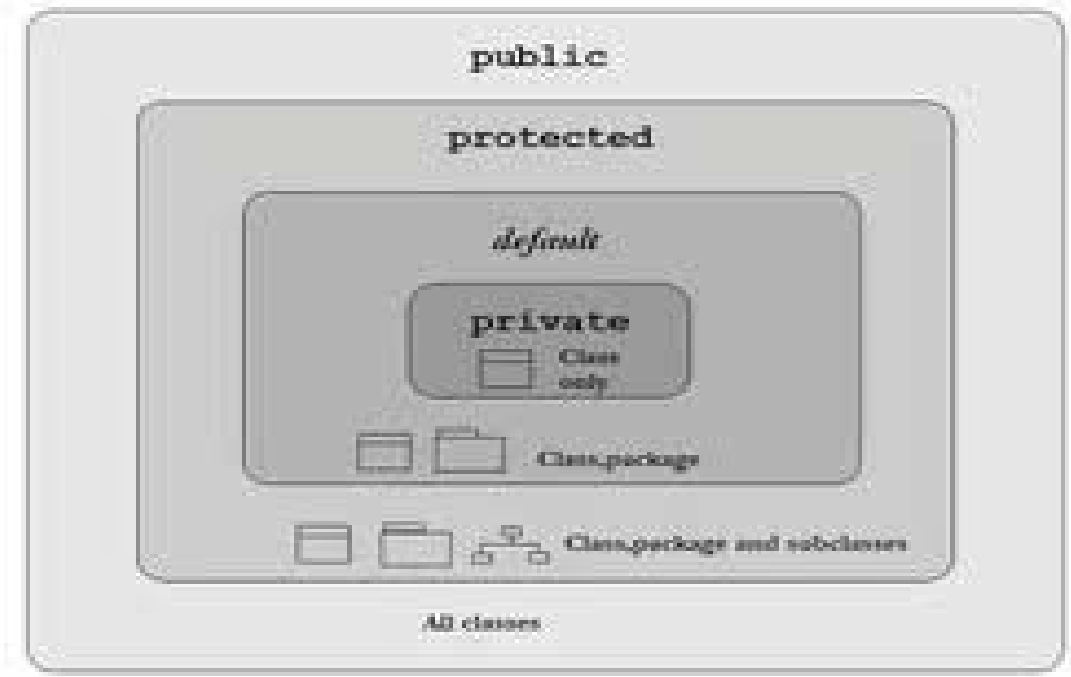
demoTest t = **new demoTest();**

- t.go();
- }
- }

Access Modifiers in java

- Access modifiers specifies access level of a java component.
- Access modifiers can be divided into two categories:

1) Class level
2) Member level



Class level access modifiers

- Public :

If a class is marked as public then it is accessible anywhere in java world.

Public class demo {}

- Default:

If a class have no modifier, then it will be marked as ‘default’ implicitly, then it is accessible in it’s package only

Class demo()

Class Member level access modifier

- Public:

If a member is marked as public then it is accessible in whole java world.

- Default:

If a member have no modifier, then it will be marked as ‘default’ implicitly, and accessible in it’s package only.

- Protected:

If a member is marked as protected then it is accessible in it’s package. It is also accessible outside the package but through “inheritance” only.(????)

- Private:

If a member is marked as private then it is accessible in it’s class only.