Arrays -> An array is a group of contiguous or related data items that share a common name. For example, salary[10] represents the salary of 10^{th} employee.

One-dimensional array

A list of items can be given one variable name using only one subscript and such a variable is called a single- subscripted variable or a one-dimensional array.

For ex,

```
int number[] = new int[5];
```

```
class array1
 {
 public static void main(String args[])
   {
   String name[]={"Rohit","Amit","Niraj"};
   int roll[]={1,2,3};
   int age[]={25,26,27};
   int i;
   for(i=0;i<3;i++)
     {
     System.out.println("name="+name[i]);
     System.out.println("roll="+roll[i]);
     System.out.println("age="+age[i]);
     }
   }
 }
```

WAP to enter no. and find largest

```
import java.io.*;
class larger
  {
  public static void main(String args[]) throws IOException
    {
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    int n,i,large;
    int num[]=new int[20];
    System.out.println("How many no. u want to enter :");
    n=Integer.parseInt(br.readLine());
    for(i=0;i<n;i++)
      {
      System.out.println("Enter number :");
      num[i]=Integer.parseInt(br.readLine());
      }
    large=num[0];
    for(i=0;i<n;i++)
      {
      if(large<num[i])
```

```
      large=num[i];
     }
   System.out.println("large=" +large);
   }
  }
```

WAP to enter number and find smallest

```
import java.io.*;
class smaller
 {
 public static void main(String args[]) throws IOException
  {
  BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
  int n,i,small;
  int num[]=new int[20];
  System.out.println("How many no. u want to enter :");
  n=Integer.parseInt(br.readLine());
  for(i=0;i<n;i++)
    {
    System.out.println("Enter number :");
    num[i]=Integer.parseInt(br.readLine());
    }
  small=num[0];
  for(i=0;i<n;i++)
    {
    if(small>num[i])
      small=num[i];
    }
  System.out.println("smaller=" +small);
  }
 }
```

Addition of two matrix

```
import java.io.*;
class sumofmatrix
 {
 public static void main(String args[]) throws IOException
  {
  BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
  int mat1[][]=new int[3][3];
  int mat2[][]=new int[3][3];
  int add[][]=new int[3][3];
  int i,j;
  System.out.println("Enter the value of first matrix :");
  for(i=0;i<3;i++)
    {
    for(j=0;j<3;j++)
     {
```

```
      mat1[i][j]=Integer.parseInt(br.readLine());
      }
     }
   System.out.println("Enter the value of first matrix :");
   for(i=0;i<3;i++)
    {
    for(j=0;j<3;j++)
     {
     mat2[i][j]=Integer.parseInt(br.readLine());
     }
    }
   for(i=0;i<3;i++)
    {
    for(j=0;j<3;j++)
     {
     add[i][j]=mat1[i][j]+mat2[i][j];
     }
    }
   for(i=0;i<3;i++)
    {
    for(j=0;j<3;j++)
     {
     System.out.print("\t" +add[i][j]);
     }
     System.out.println();
    }
   }
}
```

Class -> A class is a way of binding data and methods into a single unit. It provides a convenient method for packing together a group of logically related data items.

Object-> Instance of class is known as objects.

Syntax :

```
class class_name extends super_class_name
   {
   fields declaration;
   methods declaration;
   }
```

Here class_name and super_class_name are any valid java identifiers. The keyword extends indicates that the properties of the super_class_name class are extend to the class_name class.

Fields declaration

Data is encapsulated in a class by placing data fields inside the body of the class definition. These variables are called instance variables because they are created whenever an object of the class is instantiated.

```
class Rectangle
  {
  int length;
  int width;
  }
```

Methods declaration

It is a self-contained block of structure. Whatever method we write in the class are known as member methods. Whatever methods which does not come under scope of the class are known as non member methods. In java there are no non-member methods.

Syntax :

```
type method_name(parameter-list)
  {
  methods body;
  }
```

creating objects

An object is java is essentially a block of memory that contains space to store all instance variables. Creating an object is also referred to as instantiating an object.

Objects is java are created using the new operator. The new operator creates an object of the specified class and returns a reference to that object.

```
Rectangle rect1;                //declare the object
rect1=new Rectangle();          // instantiate the object
```

The first statement declares a variable to hold the object reference and the second one actually assigns the object reference to the variable. The variable rect1 is now an object of the rectangle.

Both statement can be combined into as :

```
Rectangle rect1=new Rectangle();
```

The method Rectangle() is the default constructor of the class. We can create any number of objects of rectangle.

Example :

```
Rectangle rect1=new Rectangle();
Rectangle rect2=new Rectangle();
```

And so on.

Note1 : Each object has its own copy of the instance variables of its class. This means that any changes to the variables of one object have no effect on the variables of another. It is also possible to create two or more references to the same object.

```
Rectangle r1=new Rectangle();
Rectangle r2=r1;
```

Accessing class members

We can't access the instance variables and the methods directly. To do this, we must use the concerned object and the dot operator.

```
object_name.variablename=value;
```