OBJECT ORIENTED PROGRAMMING SYSTEM
CHAPTER 1 : CLASS & OBJECTS

OBJECTS : Any entity which has states and behaviour is called as objects. For eg. Pen, account, building.

Pen		**Account**		**Student**	
State	**Behaviour**	**State**	**Behaviour**	**State**	**Behaviour**
Color	Writing	a/c no.	Debit	Name	Studying
Brand	Drawing	Type of ac	Credit	Id	Exam
Shape	Pointing	Bank name	Payment	Gender	Reading
Price	…..	Balance …. etc	…..	Age	writing

CLASS : -

- A class is blueprint of an objects.
- A class contains or defines states and behaviours of an objects.
- The states of the class are called as data members and the behaviours of the objects are called as function members.
- The data members of the class are represented by variables and the functions members of the class are represented by methods.

JAVA NAMING CONVENTION :

Class :

- Any entity in java which starts with upper case declared with the keyword class is called as java class.
- Class names should be nouns, in mixed case with first letter of each internal word capitalize.
  Eg. Account, AccountName;

Interface :

- Any entity in java which starts with upper case and declared with the keyword interface is called java interface.
- Interface name should be capitalize like class name.
- Eg. : interface Storing, interface RunnableInterface

Methods :

- It defines the action which is performed on data members of the class.
- Methods should be in mixed case with the first letter lower case, with the first letter of each internal word capitalize.
- Eg : run(), getData()

Variables :

- Variables should be declared in mixed case with a lowercase first letter. Internal words starts with capital letters.
- Eg: i, c, myWidth;

Constants :

- The constants should be all uppercase with words separated by underscore (“ _”).
- Eg. MIN_WIDTH = 5;

The members of class be classified into two types :

(a) Static Members (b) Non-Static members

Static members

- Any members of the class which is declared using static keyword is called as static members.

Static Members present in Same Class:

- If a static method is trying to access the static members present in the same class then it can refer to them directly with the name of static member.

Program

```
class Demo
{
        static int v1=100;
        public static void test()
        {
                System.out.println("this is test() of demo class");
    }
    public static void main (String ar[])
        {
        System.out.println("V1 is = " + v1);
        test();
        }}
```

```
G:\javaprogs\OOPS\Static>javac Demo.java

G:\javaprogs\OOPS\Static>java Demo
V1 is = 100
this is test() of demo class
```

Program 2

```
class Demo
{
                int v1=100;
        public static void test()
        {
                System.out.println("this is test() of demo class");
    }
    public static void main (String ar[])
        {
        System.out.println("V1 is = " + v1);
        test();
        }}
```

```
G:\javaprogs\OOPS\Static>javac Demo.java
Demo.java:11: error: non-static variable v1 cannot be referenced from a static context
                System.out.println("V1 is = " + v1);
                                                 ^
1 error
```

Note : - A non static variable can not be referenced by a static context.

Note : **

- Within one java program we can write any number of class.
- If a program contains multiple classes then the class which contains main method should be used as filename.

Static Members Present in Different Class :

- We can access static members of different class using the classname with dot(.) operator followed by member name.
- Syntax :

  className.memberName;

  className.memberFunction();

Note : we can not use static member in different class directly. If we use it throw error.

Program

// static member used by static method use in different class directly.

```
class Demo
{
        static int v1=100;
        public static void test()
```

```
        {
                System.out.println("this is test() of demo class");
    }}
class MainClass
{
    public static void main (String ar[])
        {
        System.out.println("V1 is = " + v1);
        test();
        }}
```

```
G:\javaprogs\OOPS\Static>javac MainClass.java
MainClass.java:15: error: cannot find symbol
                System.out.println("V1 is = " + v1);
                                                 ^
  symbol:   variable v1
  location: class MainClass
MainClass.java:16: error: cannot find symbol
                test();
                ^
  symbol:   method test()
  location: class MainClass
2 errors
```

Program :

// static member used by static method use in different class by classname.member

```
class Demo
{
        static int v1=100;
        public static void test()
        {
                System.out.println("this is test() of demo class");
    }}
class MainClass2
{
    public static void main (String ar[])
        {
        System.out.println("V1 is = " + Demo.v1);
        Demo.test();
        }}
```

NON-STATIC MEMBERS :

- Any member of the class which is declared without using static keyword is called as non-static members.
- We can access non-static members of a class only by creating the object for the class.

Object creation

Syntax :

New className()

Eg.

New Sample()

Here New is a keyword which creates a new object and Sample() is a **constructor call** which copy all the non-static member to object.

- A non static method can access non-static data members or non static function members present in the same class without creating any object.

Program : calling non-static member and method from main method in a single class by not creating any object

```
class Abc
{
	int z1=123;
	public  void view()
	{
		System.out.println("this is view() of Abc");
		System.out.println("value of a = " + z1);
	}
	public static void main(String[] args)
	{
		System.out.println("Z1 = " + new Abc().z1);
		new Abc().view();
	}}
```

```
G:\javaprogs\OOPS\abstraction>java Mainclass2
Z1 = 123
this is view() of Abc
value of a = 123
```

Program :

// use of non static members and function in other class

```
class Mainclass
{
        double k1=123.45;
        public  void count()
        {
                System.out.println("this is view() of Demo");
        }}
class Mainclass2
{
        public static void main(String[] args)
        {
                System.out.println("Program Strts ");
                System.out.println("k1 = " + new Mainclass().k1);
                new Mainclass().count();
                System.out.println("Progrm end " );
        }
}
```

```
G:\javaprogs\OOPS\abstraction>java Mainclass2
Program Strts
k1 = 123.45
this is view() of Demo
Progrm end
```

Program // static member used by static method use in same class

```
class Demo
{
                static int v1=100;
                public static void test()
                {
                        System.out.println("this is test() of demo class");
        }
        public static void main (String ar[])
                {
                System.out.println("V1 is = " + v1);
                test();
        }}
```

```
G:\javaprogs\OOPS\abstraction>java Mainclass2
V1 is = 100
this is test() of demo class
```

Program // static member used by static method use in different class.

```
class Demo
{
        static int v1=100;
        public static void test()
        {
                System.out.println("this is test() of demo class");
    }}
class MainClass
{
    public static void main (String ar[])
        {
        System.out.println("V1 is = " + Demo.v1);
        Demo.test();
        }}
```

```
G:\javaprogs\OOPS\abstraction>java Mainclass2
V1 is = 100
this is test() of demo class
```