CLASS

A class is a user-defined datatype like structure in C. In the class the data members (datatype) and methods are defined. Once the class is defined we can create the object which are called Instances of class.we can create number of object as we want.

```
Class        Class_name
{
        Variables_declaration;
        Methods_ definition;
}
```

Note: [There is no semi-colon (;) at the end of closing braces.]

CREATING OBJECT :--

Creating an object is the process of allocating block of memory that contains spaces. To store all the variables declare within a class (Instance variable). The **new** keyword can be used to create object for specified class and return references of that object.

Syntax:--

Method1

Class_nameobject_name

Object_name = new class_name();

OR

Method2

Class_nameObject_name = new class_name();

```
e.g.--      Class   Student
             {
                    Int roll;
                    Int age;
                      Void input();
                       {
                          ----------
                          ----------
                       }
                          Void    display()
                      {
                           ----------
                           ----------
                      }
}
```

1) Student s= new student();
   OR
2) Student s;
   S= new Student();

ACCESSING CLASS MEMBER :-

To access the class member we must assign some values before accessing it in the program. The following statement shows how the object are assigned and class members are accessed.

```
e.g.--          Class          Student
                {
                    Int    roll;
                    Int    age;
                      Void input()
                    {
                        ----------
                        ----------
                     }
                        Void display()
                      {
                          ----------
                          ----------
                       }
}
```

Student s= new student ();

s.roll=50;

s.age=20;

s.input();

s.display();

CONSTRUCTOR:-

Constructor is a method that initialize each object when it is created. Constructor function have the same name of the class.

e.g. :- Class Add

{

Int a,b;

Add(int x, int y)

{

A=x;

B=y;

}

Int addition()

{

Return(a+b);

}

```
}
        Class     printdata
    {
        Public static void main(String args[])
        {
            Add ab = new add(100,25);
             Int s = ab.addition();
            System.out.println("sum= "+5);
        }
    }
```

Input Data Through Keyboard :--

```
        Import java.util.Scanner;
Class     Add
{
      Int a,b;
     Add(int  , int y)
    {
        A= x;
        B= y;
    }
```

```
    Int    addition()
    {
       Return  (a+b);
    }
Class   printdata
 {
    Public static void main(string args[])
   {
       Scanner sc= new  Scanner (System.in);
       Int a,b;
        System.out.println(“enter first number = ”);
         a=  sc.nextInt();
         System.out.println(“enter second number = ”);
          b= sc.nextInt();
          Add   ab = new add();
           int   s= ab.addition();
          System.out.println(“Sum of number = ”+s);
  }
}
```

METHOD OVERLOADING :--

1. Method overloading is also called function overloading.
2. It is the process o polymorphism.
3. It means using the same function name for different purpose the appropriate function is executed depending upon the argument passed to it.

Q. WAP. Or demonstrating function (method) overloading.

Solu.

```
Import java.util.Scanner;
Class add
{
Int a, b;
Void addition(int x , int y)
{
a= x;
b= y;
}
int addition ()
{
return(a+b);
}
}
```

```
Class        printdata
    {
            Public static   void main( stringargs[])
            {
              Scanner sc = new Scanner(System.in);
               System.out.println("Enter first number");
                int  a1= sc.nextInt();
                System.out.println("Ente second number= ");
                  int a2 = sc.nextInt();
                  Add ab = new add();
                   ab.addition(a1,a2);
                   System.out.println("sum = " +s);
        }
    }
```

STATIC MEMBERS :--

If we want to access a variable or method in all the classes declared in the program then declare that variable or method as a static. In the class the ordinary variable and methods are called instance variable and instance methods. The static variable and methods in the class are called class variables and class methods. In java method class has number of static methods we can directly call it like method square(2):

e.g.:- Class mathfunction

```
{
  static int square(int a)
  {
      Return(a*a);
  }
  Static int doubles(int b)
  {
      Return(b*2);
  }
}
Class Application
{
    Public static void main(String args[])
    {
        Int a = Mathfunction.square(4);
        Int b = Mathfunction. Double(2);
        System.out.println(“a = ” +a);
        System.out.println(“b = ” +b);
    }
```

}

RESTRICTION OF USING STATIC VARIABLES OR METHODS :-

(1) The static methods can only use static data.
(2) The static method can only call another static method.
(3) They cannot use a super keyword.

NOTE : [In class the ordinary variables and methods are called instance method. If we want to call the static variable or method no need for the object, we can call it directly with the class name. Static variables and methods are called class variables and methods.]

```
e.g. ->  class     X
         {
              Int   a;                    Instance
              Int  b;                     Variable
              Void show ()
                {
                   ----------             Instance
                   ----------               Method
                 }
          }
```

⇨ Class X

```
{
        static  int a,b   --- Class variable
        static  void Show ()   -----   Class methods
        {
            --------
            --------
        }
}
```

NESTING OF METHODS :-

We can call a method by another method of the same class is called nesting methods.

e.g. --

```
Class  addition
{
    Int x,y;
    addition(int a , int b)
    {
        X=a;
        Y=b;
    }
    int add()
    {
        return(x+y);
    }
```

```
                    void      display()
                    {
                        int sum = add();
                       System.out.println("Sum of value=+sum");
                     }
                }
Class          mainin
    {
           Public static void main(String args[])
        {
            addition   a =    new addition (10,20);
            a.display();
          }
    }
```