Function

A function is defined as a set of instructions given to the computer in order to perform a specific task. Each function has a unique name/identity with which it is called within a program.

Functions are used to perform certain actions and they are also known as methods.

A function is a block of code which only runs when it is called.

Programmers use methods to reuse code: define the code once, and use it many times.

Java functions are divided in two categories:

1. Built in/Library/ Predefined Functions: Java provides some pre defined methods, such as : System.out.print( ), System.out.println( ), Math.pow( ), Math.sqrt( ), substring( ), valueOf( ), length( ), charAt( ), parseInt( ) etc.

2. Use defined Functions: The methods defined by user is called user defined functions, such as : main( )

Parts of Functions:- A function has two parts:

1. **Function Header:-** It specifies the access specifier, return type, function name and the parameter list of the of the function. It's also known as function signature.
2. **Function Body:-** The function body is the remaining part of the function after the header enclosed by a pair of braces.
   Any set of declarations and statements between a pair of braces is known as a block.

accessspecifier returntype functionname (parameter list/argument list)//Function header

```
        {                       // starts of fumction body
            variables;
            statements;
Function    …….
Body        …….
            return statement;
        }                       // end of function body
```

Example:

```
public void main(int x)                 //Function header
        {                       // starts of fumction body
                int x,y,z;
                x=10;
                y=6;
                z=x+y;
        }                       // end of function body
```

1. Access Specifier: - It is public, private, protected or default. If programmers don't use any access specifier (public, private, protected) then function work in default scope.

2. Return Type: - It specifies the type of data returned by the method. It can be any valid data type. If a function doesn't returns a value, its return type must be void.

3. Function Name: - Function name shouldn't be same as the class name.

4. Parameter List: - Parameter lists are variables that receives the value of the arguments passed to the function when it's called. It can be empty if there are no parameters.

Function Declaration: - Function declaration is function header with semicolon and parameters are only with data type excluding variable name.

Function Calling: - Function calling is call the function to perform task. Function call by name and parameters of functions. If function is static then it's called by class name otherwise called by object.

Function Definition: - Function Definition is to define the function with function

header and body of the function.

General Form of class: -

```
class  ClassName
{   // start of class
        instance variables;
        static variables;
        member methods()
        {
          body of the method
        }
}// end of class
```

1. class is a keyword that identifies class declaration
2. ClassName is name of class. It is user defined name (identifier).
3. The method and variables defined within a class are called members of the class.

Types of Function: - Functions can be divided into two types:

1. Pure Functions or Accessor Methods

2. Impure Functions or Mutator Methods

1. Pure Functions or Accessor Methods:-

Pure Functions return information to the caller about the state of an object. An accessor

usually contains a return statement or prints information about an object's state.

2. Impure Functions or Mutator Methods:-

Impure Functions that change the state of an object. Mutators as requests for an object to

change its state.

Example:

public class Example

```
{
double a; // instance variable of double type
int b,c; // instance variable of int type


void input(int x,int y)          //impure method change value of instance
variables
{
b=x;
c=y;
}
void sum()                       //impure method change value of instance
variables
{
a=b+c;
}

void display()                   // pure method print value of instance variable
{
System.out.print("sum of two numbers="+a);
}
int valueSum()                   // pure method return value of instance variable

{
return a;
}
}
```

Argument in Function: - An argument is a value that is passed to a function when it is

invoked or called

Parameter in Function: - A parameter is a variable defined by a method that receives a

value when the method is called or invoked.

There are two types of parameters: ***Actual parameters*** and ***Formal parameters***.

Difference between Actual and Formal parameters:

Actual parameters	Formal parameters
1. The parameters of the method when the method is called.	1. The parameters of the method when the method is actually defined.
2. They are not preceded by datatype.	2. They are preceded by datatype.

Classification of a Function:

1. Function which does not have parameter or return statement.
2. Function which has parameter but no return statement.
3. Function which has no parameter but has return statement.
4. Function which has both parameter and return statement.

Example:

```
class Example
{
  void function1( ) // Function which does not have parameter or return
statement.
  {
        System.out.print(“Welcome”);
  }
  void function2( int k)    // Function which has parameter but no return
statement.
  {
        System.out.print(“value”+k);
```

```
    }
    int  function3( )  // Function which has no parameter but return statement.
    {
         return 5;
     }
     int  function4(int a )     // Function which has parameter and return statement.
     {
         return a*a;
      }
}
```

Recursive Function or Recursion: - Recursive function is function that call itself until

the given condition is true.

Example:

```
class Example
{
        void display(int n)
        {
                System.out.print(n);
                if(n>=1)
                display(n-1);
        }
}
```

Function Overloading:

Same name functions with different parameters in same class is called as function

overloading.

Example:

```
class Example
{
        void display( )
```

```
    {
        System.out.print("welcome");
    }
    void display(int n)
    {
        System.out.print("value= " + n);
    }
}
```

Need for Function Overloading: - Objects have certain characteristics and associated

behavior, but an object may behave differently in different situations. To simulate real-

world objects in programming, function overloading is required.

Constructor: Constructor is special function which has same name as class name. It is used to initialize all variables in a class each time an instance of the class is created.

A constructor automatically gets invoked whenever an object of the class is created and is used to initialize the all data members.

Constructor has some rules & prop -

* Constructor name & class name must be same.

* Constructor no return any value that means not void

* every class has one Constructor.

* if user does not define any Constructor in a class then default Constructor created.

* if class has any own Constructor then default Constructor not created.

* if one class has more then one Constructor then that is called Constructor overloading.

* Constructor automatically called (invoked) whenever Object is created.

* Constructor should be declared as public

* Constructor can be private, if Constructor is private then can't called outside the class

body. used for creating factory methods.

factory methods - if class has only private Constructor/s & all static methods of that

class are called factory method.

adv - where no need to create the new resources used to existing resources.

Role of new keyword: The use of new keyword in java is to create instance (objects) of

a class. The syntax to create object can be given as fallows

Difference between constructor and function

Constructor	Function
Constructor has the same name as the class name.	Functions do not have the same name as the class name.
Constructor has no return type.	Functions may have return type.
Gets invoked when object is created.	Gets invoked when object calls the method using dot operator.
Used to initialize instance variables.	Used to perform a specific task.
The access of constructor should always be public.	The access can be public, private or protected.

Types of Constructor: - Three types of constructors

1. Default Constructor
2. Parameterized Constructor
3. Copy Constructor

1. Default Constructor: - When programmer does not create any constructor then compiler create one constructor that is called as default constructor.

Programmer can also define default constructor. The constructor has no

parameter is

called as default constructor.

2. Parameterized Constructor: - The constructor has parameter is called parameterized

constructor. It can be defined only by programmer.

3. Copy Constructor: - It is used to create object as existing ones.