- **Functions and Methods :-** Functiones are nothing but the block of code, which only runs when it is called.

Syntax :-

```
return_type name (arguments)
{
    // body
    return statement;
}
```

- **Display the function :-**

```
import java.util.*;
public class Main
{
    public static void main (String[] args)
    {
        sum();  // calling the function
    }

    static void sum()
    {
        Scanner sc = new Scanner (System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        int c = a+b;
        sout (c);
    }
}
```

- Return the value :-

```
import java.util.*;
public class Main
{
    public static void main (String[] args)
    {
        int ans = sum2 (40,60);
        sout (ans);
    }
// Returning the value :-
    Static int sum ()
    {
        int a = 10;
        int b = 20;
        int c = a+b;
        return c;
    }

    Static int sum2 (int a, int b);
    {
        int sum = a+b;
        return sum;
    }

// Returning the String

    Static String name (String name)
    {
        String greet = "Hello" + name;
        return greet;
    }
```

Q→ Swap two numbers using function.

```
import java.util.*;
public class Main ()
{
    public static void main (String [] args)
    {
        Scanner sc = new Scanner (System.in);
        int a = sc.nextInt ();
        int b = sc.nextInt ();

        swap (a, b);
    }

    public static void swap (int a, int b)
    {
        int temp = a;
        a = b;
        b = temp;
        Sout (a+" " + b);
    }
}
```

Q→ Change the value of an array :-

```
import java.util.*;
public class main (
{
    public static void main (String [] args)
    {
```

```
int [] arr = {1, 4, 3, 10};

    Changes (arr);
    Sout (Arrays.toString (arr));
}

public static void changes (int [] nums)
{
    nums [0] = 90;
}
}
```

- **Scoping :-**

In java, variables are only accessible inside the region they are created, This is called scoping.

- **Block Scope :-**

A block of code refers to all of the code between curly braces {}.

↳

```
public class Main
{
    public static void main (String [] args)
    {
        int a = 10;
        int b = 20;
```

```
int c = 40;
{
    c = 60;  // Changing the value. but we
             can't intialised the value of c.
```

→ Anything that is initialized outside the block can be used in inside the block but anything that is initialized the block can't be used outside of the block.

- **Shadowing :-**

Shadowing occur when a variable declared in a certain scope has the same name as a variable in a outer scope.

Ex :-

```
public class main
{
    public static void main (String [] args)
    {
        Static int x = 90;  // Can we this
                               In entire code

        public static void main (String [] args)
        {
            Sout (x);
            int x = 40;  // Shadowing
            Sout (x);
        }
    }
}
```

• Variable arguments :-

A variable length argument is specified by three periods of ~~time~~ dots.

Syntax :-

```
static void fun (int... a)
{
// Method body
}
```

It tells that fun() can be called with 0 or more arguments.

Ex :-

```
import java.util.Arrays;
public class Main
{
public static void main (String [] args)
{
Show (90, 10, 70, 65, 40, 20)
}

public static void show (int... a)
{
Sout (Arrays.toString (a));
}
}
```

- **Function Overloading :-**

There can be two functions with same name but their return type should be different.

Ex :-

```
public class main
{
    public static void main (String [] args)
    {
        fun (2,4);
        fun ("Aditya");
    }

    Static void fun (int a, b)
    {
        Sout (a+b);
    }

    Static void fun (String s)
    {
        Sout (s);
    }
}
```