Method make the code reusable.

Method has two parts 1. Method definition 2. Method call

Method complete its task control go back to calling Method.

Every Method has its own copy of variable.

Method can return only one value

//Static method can call only other static method directly. It resides in **external memory block**

**Function/Method without return type.**

**import** **static** java.lang.System.***out***;

2

a

2

2000

2004

b

**import** java.util.\*;

**public** **class** Myclass {

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

**int** a=sc.nextInt();

**int** b=sc.nextInt();

Myclass*.calsum*(a,b);//a b is actual parameter

***out***.print(“bye”);

}

**public** **static** **void** calsum(**int** x, **int** y) //formal argument (type,order and no. of the actual and formal argument must always be same)

{

2

x

2

y

4000

4004

4

d

4008

**int** d;

d=x+y;

2

x

2

y

4000

4004

4

d

4008

***out***.print(d);

}

}

Type promotion in parameter

import static java.lang.System.out;

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

float a=sc.nextFloat();//2.5

int b=sc.nextInt();//3

calsum(a,b);//a b is actual parameter

int x, int y🡺err

float x , int y

}

public static void calsum(float x, float y) //formal argument (type,order and no. of the actual and formal argument must always be same)

{

float d;

d=x+y;

out.print(d);//5.5

}

}

Function returning value.

**import** **static** java.lang.System.***out***;

**import** java.util.\*;

**public** **class** Myclass {

2

a

2

2000

2004

b

r

4

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

**int** a=sc.nextInt();//2

**int** b=sc.nextInt();//2

**int** r=*calsum*(a,b);//4

***out***.print(r);

***out***.print(*calsum*(5,6));//11

}

**public** **static** **int** calsum(**int** x, **int** y) //formal argument (type,order and no. of the actual and formal argument must always be same)

return no%2!=0

{

**int** d;

d= x+y;

boolean a= no%2!=0;

**return** (d);

}

}

Task

1. Accept a number and print square of a number
   1. Return type int //input int🡺2 🡺4
   2. Return type float//input int🡺2🡺4.0
   3. Return type int //input float 🡺2.5 🡺typecast and return 6 (6.25)
   4. Return type float//input float🡺2.5🡺6.25

Can we write static method in separate class?

Ans: yes. You have to create separate file with the classname for example have created new file and name of the file is Logicsum. In this file I have to added static method. From Myclass just call method like this Logicsum.calsum(5,7);

**public** **class** Logicsum {

**public** **static** **int** calsum(**int** x, **int** y) //formal argument (type,order and no. of the actual and formal argument must always be same)

{

**int** d;

d=x+y;

**return** (d);

}

}

Just call method with class name.

Lets modify above code. And put it in separate class.

So now you have two class Myclass which I will call as entrypoint class because it has method “main”

Complete example here

File 1. Myclass.java

**mport** **static** java.lang.System.***out***;

**import** java.util.\*;

**public** **class** Myclass {

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

**int** a=sc.nextInt();//2

**int** b=sc.nextInt();//2

**int** r= Logicsum.*calsum*(a,b);//4

***out***.println(r);

***out***.print( Logicsum.*calsum*(5,6));

}

}

File 2. Logicsum.java

**public** **class** Logicsum {

**public** **static** **int** calsum(**int** x, **int** y) //formal argument (type,order and no. of the actual and formal argument must always be same)

{

**int** d;

d=x+y;

**return** (d);

}

}

Now onwards we will be solving all logical program in a separate class with static methods in it.