Methods In Java

```
class Test
{
int a=10;
int b=20;
System.out.println(a+b);
//Directly business logic is not allowed inside class

void add()
{
System.out.println(a+b);
//Method is used to write business logic inside the mehtod
}



}
```

2 Types of Methods:

1.instance methdod

```
void m1()
{
        //logic here
}
```

2.static method

```
static void m1()
{
        //logic here
}
```

Syntax:

```
Modifiers_List ReturnType Method_Name(Parameters_List) throws Exception
{
}
```

Example:

```
public void m1(int a,int b)
{
}


private int m2();



protected int m3(int a) throws Exception
{
}
```

Method Signature:

Method_Name(Parameters_List)

Example:
m1(int a,int b)

m2()

m3(int a)

Every method contain 3 parts

```
void m1()//1 Method declaration
{
        //logic  //2.method implementation

}

t1.m1();   //3.Method calling
```

Ex1:How to access instance methods and static methods

create object for a class and using that object to access instance method

in the case of static method access directly using class name

```
class TestEx
{
void m1()//instance methods
        {
                System.out.println("Instance Method");

        }

static void m2()//static method
        {
System.out.println("Static Method");
        }


        public static void main(String args[])
        {
    TestEx ex1=new TestEx();
         ex1.m1();

         TestEx.m2();
        }



}
```

==========

Ex2:Method with Parameters Call by value or Pass by Value

```
class TestEx
{
void m1(int a,char ch)//formal parameters
        {
```

```
        System.out.println("m1() Method");
        System.out.println(a);
        System.out.println(ch);

        }

static void m2(String str,double d)
        {
        System.out.println("m2() Method");
        System.out.println(str);
        System.out.println(d);

        }
public static void main(String args[])
        {
TestEx ex1=new TestEx();
ex1.m1(10,'a');//actual parameters

TestEx.m2("Ashok",10.5);

//ex1.m1("Ashok",10.5);
        }



}
```

Ex3:Method With Expecting Objects Or Call by Reference or Pass by Reference

```
class X
{
}

class Y
{
}

class Emp
{
}
```

```
class Student
{
}


class MyTest
{
void m1(X x,Emp e)
{
System.out.println("m1 method");
}

static void m2(Y y,Student s)
{
System.out.println("m2 method");
}

public static void main(String... args)
        {
        MyTest t=new MyTest();


        //X x1=new X();


        //Emp e1=new Emp();

        //t.m1(x1,e1);

        t.m1(new X(),new Emp());


        //Y y1=new Y();

        //Student s1=new Student();

        //MyTest.m2(y1,s1);

        MyTest.m2(new Y(),new Student());
```

```
	}
}

==================================
//Method calling
class Test2
{
 int x=100;
 int y=200;

 void add(int a,int b)
	{
	 System.out.println(x+y);
	 System.out.println(a+b);

	}
	public static void main(String... args)
	{
		Test2 t1=new Test2();
		t1.add(1000,2000);//method calling
	}
}

output:
300
3000
--------------------
//To represent instance variables use this keyword

class Test2
{
 int x=100;
 int y=200;

 void add(int x,int y)
	{
	 System.out.println(x+y);
```

```
        System.out.println(this.x+this.y);

        }
        public static void main(String... args)
        {
                Test2 t1=new Test2();
                t1.add(1000,2000);//method calling
        }
}
--------------------
//Inside the static method this keyword is not allowed

class Test2
{
 int x=100;
 int y=200;

 static void add(int x,int y)
        {
         System.out.println(x+y);

         System.out.println(this.x+this.y);

        }
        public static void main(String... args)
        {
                Test2 t1=new Test2();
                t1.add(1000,2000);//method calling
        }
}
```