HEAP and STACK Diagram :

```
public class Test
{
  int x = 100;

  public static void main(String [] args)
  {
     Test t1 = new Test();
     t1 = new Test();
  }
}
```

javac Test.java
java Test

Whenever we execute the progran then JVM will get some memory from the O.S. This memory is divided into two sections

1) HEAP MEMORY (Storing the Object)
2) STACK MEMORY (Method Execution)

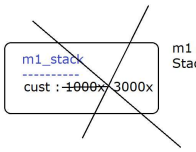
HEAP MEMORY
1000x : TestObject, x : 100
2000x : TestObject, x : 100

STACK MEMORY
main_stack
---
t1 : ~~1000x~~ 2000x

HEAP AND STACK Diagram for CustomerDemo.java

HEAP
1000x : CustomerObject, name : 2000x, id : ~~2~~ 5
2000x : StringObject,Ravi
3000x : CustomerObject, name :4000x, id : ~~7~~ 9
4000x : StringObject, Rahul

STACK
main_stack
---
val : 100
c : 1000x



m1 method execution is over so it will be deleted from the Stack Frame

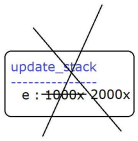
---

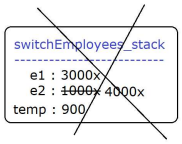
HEAP and STACK Digram for Employee.java

HEAP MEMORY
1000x : EmployeeObject, id : 100 ~~200~~ 500
2000x : EmployeeObject, id : ~~100~~ 400
3000x : EmployeeObject, id : ~~100~~ ~~900~~ 500
4000x : EmployeeObject, id : ~~100~~ 900

Output
400
500
500
500

STACK MEMORY
main_stack
---
val : 200
e1 : 1000x
e2 : 3000x





//HEAP and STACK Diagram for Test.java

HEAP MEMORY
1000x : TestObject, t : ~~null~~ 3000X , val : 100
2000x : TestObject, t : ~~1000x~~ 3000x 2000X , val : 200
3000x : TestObject, t : ~~1000x~~ 4000x , val : 300
4000x : TestObject, t : 2000x, val : 400

t2.t = t3;
t3.t = t4;
t1.t = t2.t; //3000x
t2.t = t4.t; //2000X

```
System.out.println(t1.t.val); //300
System.out.println(t2.t.val); //200
System.out.println(t3.t.val); //400
System.out.println(t4.t.val); //200
```

STACK MEMORY
main_stack
---
t1 : 1000x
t2 : 2000x
t3 : 3000x
t4 : 4000x

HEAP and STACK daigram for Beta.java

HEAP MEMORY
1000x : BetaObject
2000x : AlphaObject, val : ~~9~~ 15
3000x : AlphaObject, val : 2
4000x : AlphaArrayObject[~~3000x~~ 2000x , 2000x]

```
System.out.println(ar[0].val); 15
System.out.println(ar[1].val); 15
```

CLASS_DATA
100x : Alpha.class -> sval : 200 , b : 1000x

STACK MEMORY
main_stack
---
am1 : 2000x
am2 : 3000x
ar : 4000x