**1.Types of Objects**:- In Java, we have two types of objects based on referenced variables.

a. Referenced or Rechable object.

b. UnReferenced or UnReachable object.

**1.a) Referenced Object:-** If an object is pointed by at least one referenced variable, It is called Referenced Object.

1**.b)UnReferenced Object:-** If an object is not pointed by at least one referenced variable , it is called Unreferenced variable.

Examples:

a)Employee e1=new Employee(); //Referenced Object

b)Student s1=new Student(); //Referenced Object

c) new Student(); //Un Referenced object

d) new Employee(); //UnReferenced Object.

Note:1 we can access members from referenced object.

Note:2 only one time and only one membet can be accessed from UnReferenced Object.

Example: new Employee().ename;

Q) When do u create referenced object?

A) If we want to access all or single member from object more than one once, we must create referenced object.

Q) when do u create unreferenced object?

A) If we want to access single member that too only one time from object, we must create unreferenced object.

Example:

import java.util.\*;

import java.lang.Object;

class Address{

String dno="13-8";

String city="nlr";

}

class Employee extends Address{

byte eno=1;

String ename="suku";

protected void finalize()throws Exception{

System.out.println("finalized executed");

}}

public class Sample extends Employee implements Cloneable{

public static void main(String args[])throws CloneNotSupportedException {

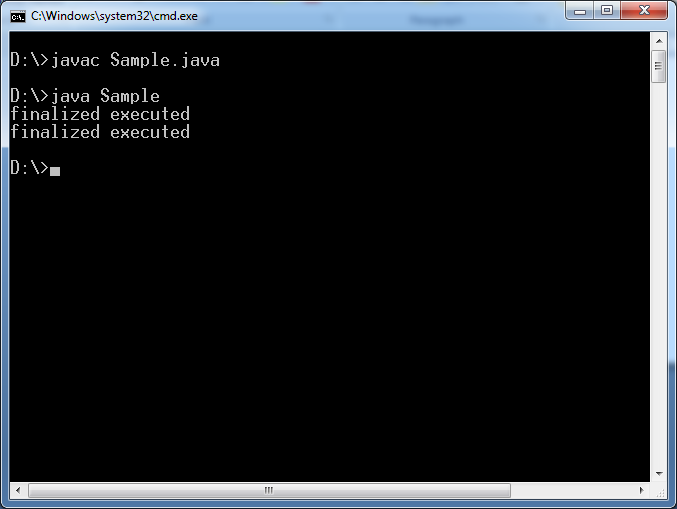
Sample e1=new Sample();

e1=null;

new Sample();

System.gc();

}}



**2.JVM Destroyed Objects:- JVM** destroys only unreferenced objects. It will not destroy the Referenced objects.

We can convert referenced object into unreferenced object in 3 ways:

1. Stroing a “null” in referenced variable.
2. Stroring a another object reference in in referenced variable.
3. Creating islands of isolations.

Example:-

import java.util.\*;

import java.lang.Object;

class Employee {

byte eno=1;

String ename="suku";

protected void finalize()throws Exception{

System.out.println("finalized executed");

}}

public class Sample extends Employee implements Cloneable{

public static void main(String args[])throws CloneNotSupportedException {

Sample e1=new Sample();

Sample e2=new Sample();

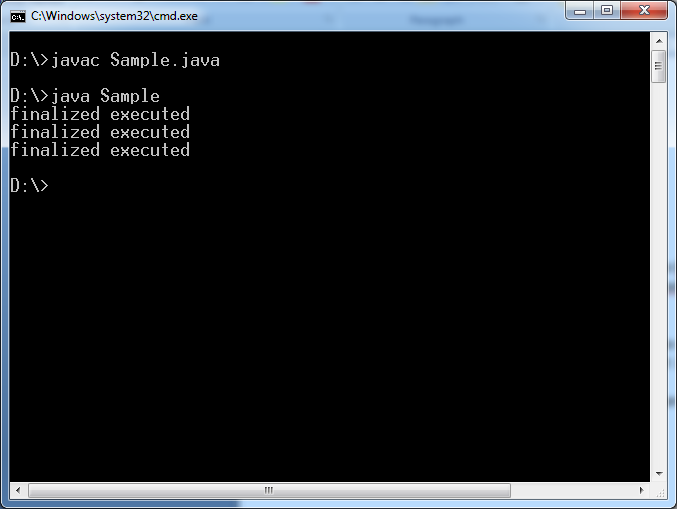
e1=null;//Here referenced object is converted into unreferenced object.

e2=new Sample(); ///here also referenced object is converted into unreferenced object.

new Sample();//here also referenced object is converted into unreferenced object.

System.gc();

}}



**3.Garbage Collection:-** The process of removing unreferenced objects is called Garbage Collection.

Programmer Responsibility:- Java developer responsibility is only creating and unreferencing objects.

JVM Responsibility:- JVM take care of destroying the unreferencing objects.

JVM Internally uses the daemon thread called “garbage collector” to destroy all unreferenced objects. A daemon thread is service thread which runs behind the application. The Garbage collector is started by JVM. The JVM controls garbage collector; It decides when to run Garbage collector. JVM runs Garbage collector when it realizes that the memory is running low.

This daemon thread is low priority thread. Since It is a low priority thread we can not guarantee this thread execution. Therefore we can not guarantee objects destruction even though it is unreferenced.

The programmer request JVM to start garbage collector execution using one of following method.

1. System.gc();

Which is static method of System class.

2. Runtime.getRuntime.gc();

Even programmer request JVM, JVM may start / may not start garbage collector execution.

Note:- If there is no space in Heap area to create new object ,then “ JVM terminates program execution by throwing **“java.lang.OutOfMemory**”.