NETTUR TECHNICAL TRAINING FOUNDATION

DIPLOMA IN COMPUTER ENGINEERING & CLOUD COMPUTING

SEMESTER – III

WORK INSTRUCTION

FOR

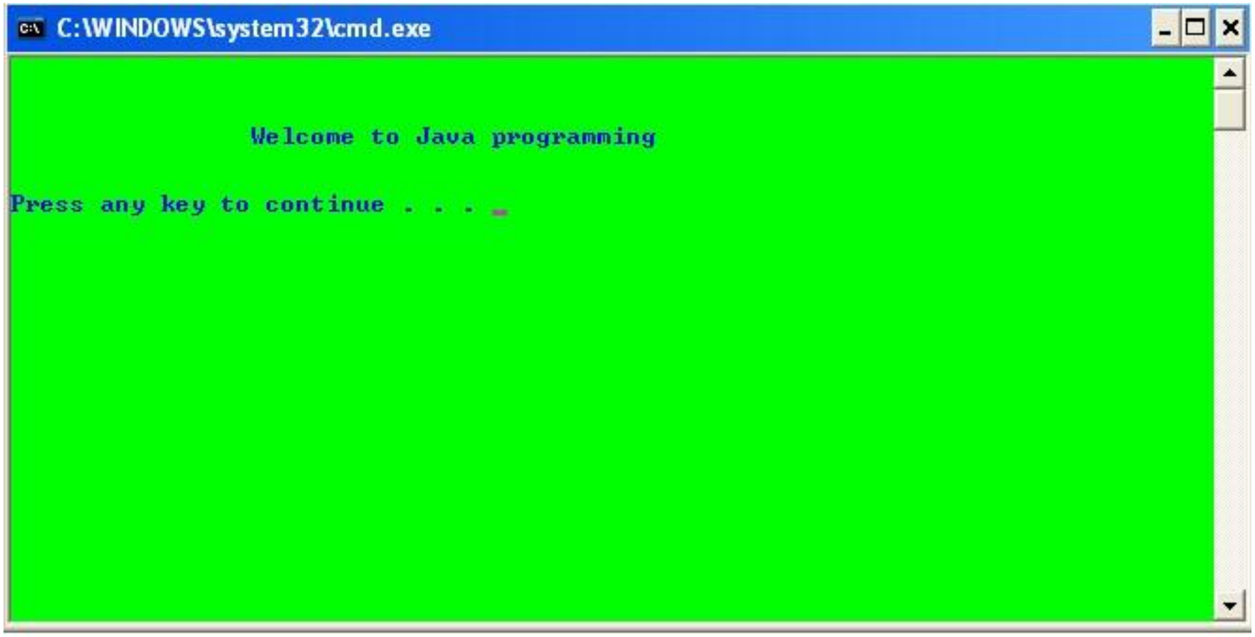
JAVA LAB

CP08 03 07

1. Write a program to display your information on the console.

```
class P1
{
        public static void main(String args[])
        {
                System.out.println("Welcome to Java programming");
        }
}
```

OUTPUT:

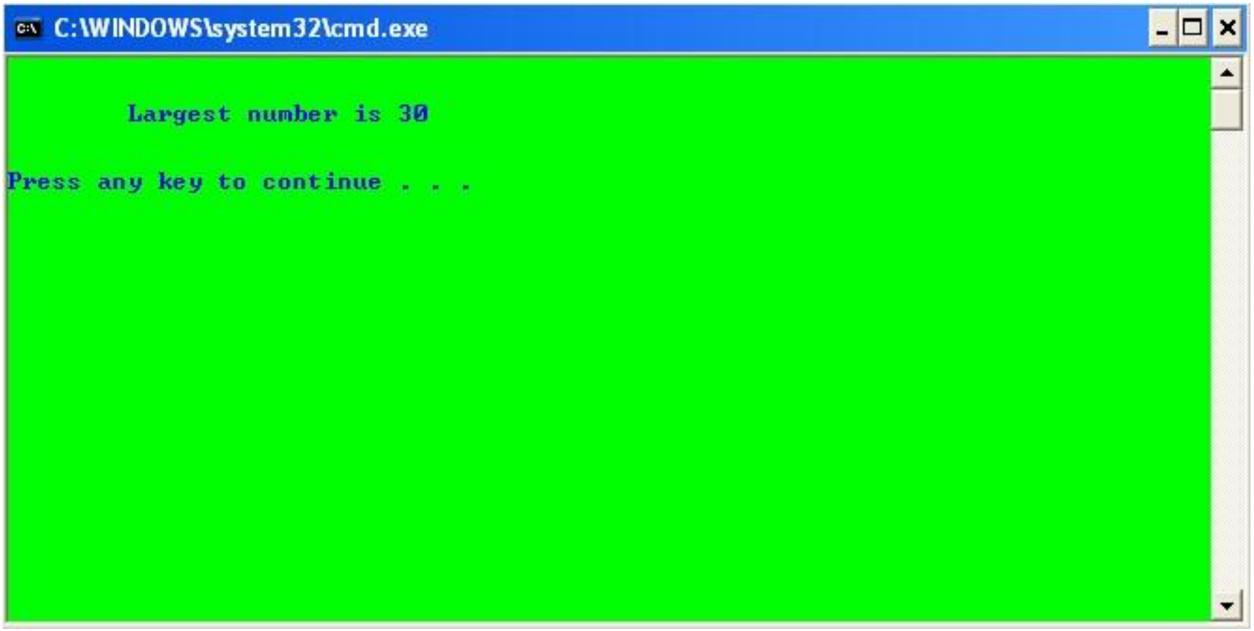


```
                Welcome to Java programming

Press any key to continue . . . _
```

2.Write a program to use the conditional statements.

```
class P2
{
        public static void main(String args[])
        {
                int a=10,b=20,c=30;

                int l = a>b&&a>c?a:b>a&&b>c?b:c;

                System.out.println("Largest number is " +l);

        }
}
```

OUTPUT:

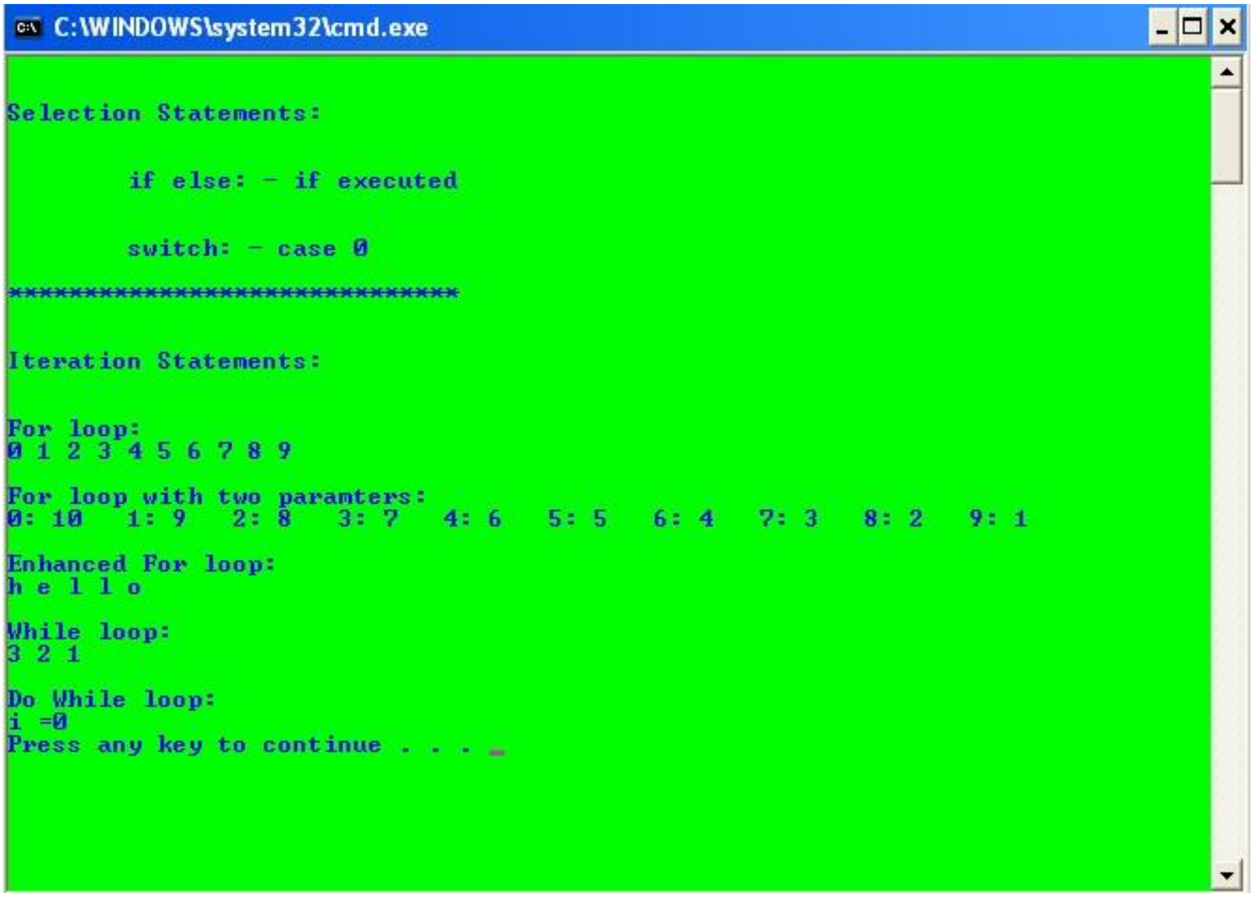


```
        Largest number is 30

Press any key to continue . . .
```

3.Write a program to use the control statements.

```
class P3{
        public static void main(String args[]) {

                System.out.println("\n\nSelection Statements: ");

                int a=0;

                if(a==0)System.out.println("\n\n\tif else: - if executed ");
                else System.out.println("\n\n\tif else: - else executed ");

                switch(a++)             {
                        case 0:System.out.println("\n\n\tswitch: - case 0 ");break;
        case 1:System.out.println("\n\n\tswitch: -  case 1 ");break;
                        default:System.exit(0);                 }

                System.out.println("\n\nIteration Statements: \n\n");

                System.out.println("For loop: ");
for(int i=0;i<10;i++) System.out.print(i+" ");

  System.out.println("\n\nFor loop with two paramters: ");   for(int i=0,
j=10;i<10;i++,j--) System.out.print(i+": "+j+"   ");

  System.out.println("\n\nEnhanced   For   loop:   ");        char
b[]={'h','e','l','l','o'}; for(int i:b) System.out.print((char)i+" ");

                System.out.println("\n\nWhile loop: ");
                int i=3;
while(i!=0)     {
                        System.out.print(i+" ");        i--;            }

                System.out.println("\n\nDo While loop: ");
                do{
                         System.out.println("i ="+i); }while(i!=0);
        }
}
```

OUTPUT:

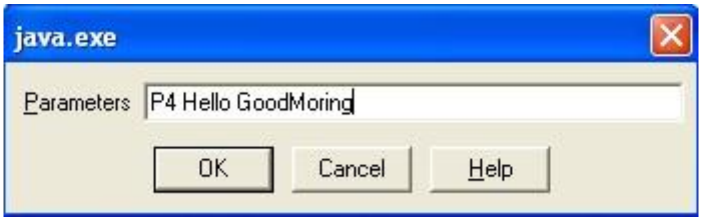


```

Selection Statements:


        if else: - if executed


        switch: - case 0
******************************


Iteration Statements:


For loop:
0 1 2 3 4 5 6 7 8 9

For loop with two paramters:
0: 10  1: 9   2: 8   3: 7   4: 6   5: 5   6: 4   7: 3   8: 2   9: 1

Enhanced For loop:
h e l l o

While loop:
3 2 1

Do While loop:
i =0
Press any key to continue . . .
```

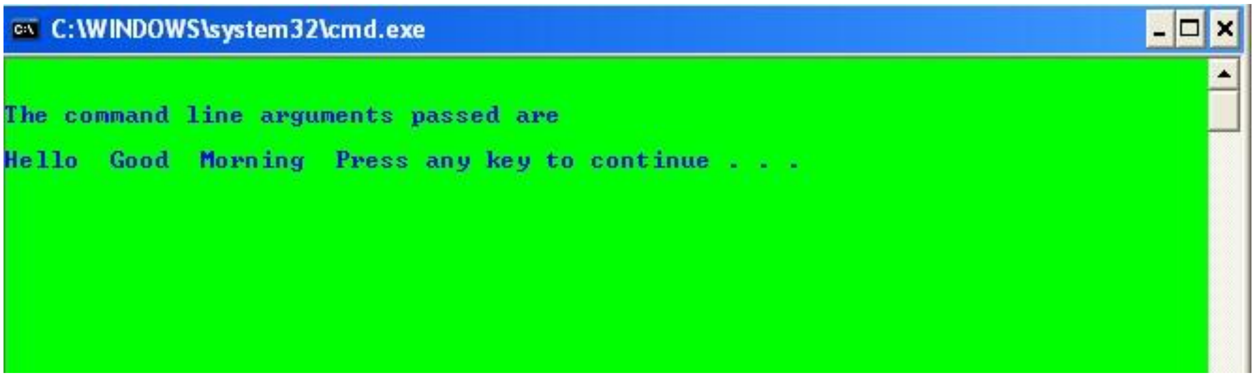
4. Write a program to pass command line arguments and display the same.

```
class P4
{
        public static void main(String args[])
        {
                System.out.println("\n\nThe command line arguments passed are \n" );

  for(int i=0;i<args.length;i++)    System.out.print(args[i]+"  " );

        }
}
```

OUTPUT:



Parameters P4 Hello GoodMoring

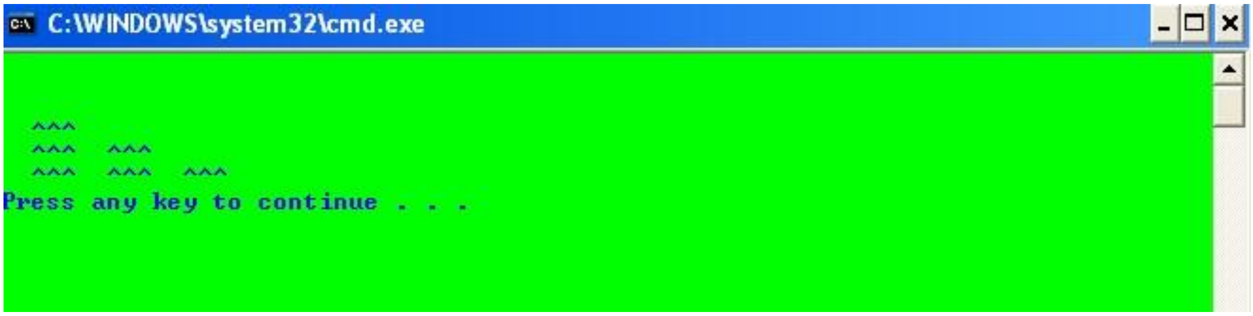


```
The command line arguments passed are
Hello  Good  Morning  Press any key to continue . . .
```

5. Write a program to declare a string array of with varying width and display the content of the array in the same manner in which the data is stored.

```
class P5
{
        public static void main(String args[])
        {
                String s[][]=new String[3][];

                s[0]=new String[1];    s[1]=new String[2];            s[2]=new String[3];

                for(int i=0;i<s.length;i++)
                for(int j=0;j<i+1;j++)
                        s[i][j]="^^^";

                for(int i=0;i<s.length;i++)      {
                        for(int j=0;j<i+1;j++)          System.out.print("  "+s[i][j]);
                         System.out.println(); }

        }
}
```

OUTPUT:

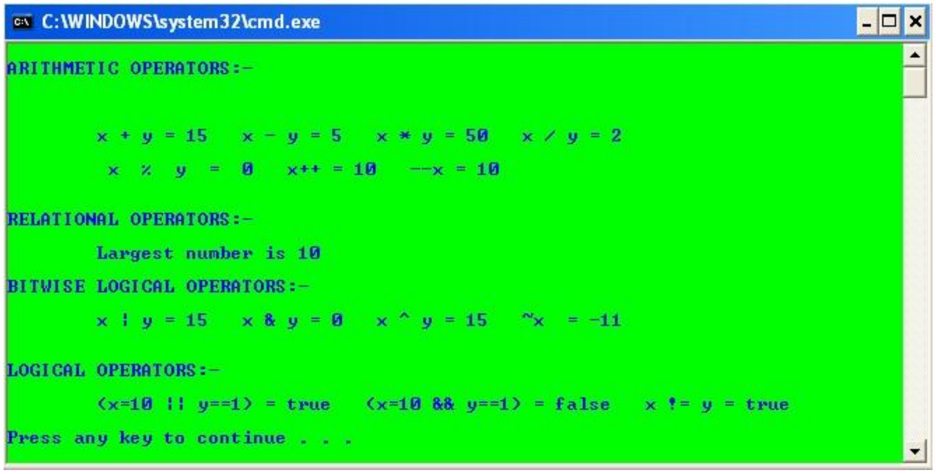


```
  ^^^
  ^^^   ^^^
  ^^^   ^^^   ^^^
Press any key to continue . . .
```

6. Write a program to demonstrate the use of operators.

```
class P66{
        public static void main(String args[]) {
                        int
                x=10,y=5,z=1;

                System.out.println("\nARITHMETIC OPERATORS:-\n\n");
                System.out.println("\n\tx + y = "+(x+y)+"   x - y = "+(x-y)+"   x * y =
                "+(x*y)+"   x / y = "+(x/y)+"\n\n\t x  %  y  =  "+(x%y)+"   x++ =
                "+(x++)+"   --x = "+(--x)+"\n");

                System.out.println("\nRELATIONAL OPERATORS:-\n\t");
if(x>y && x>z)          System.out.println("\tLargest number is " +x);
else if(y>z && y>z) System.out.println("\tLargest number is " +y);
else                        System.out.println("\tLargest number is " +z);

                System.out.println("\nBITWISE LOGICAL OPERATORS:-\n\t");
                System.out.print("\tx | y = " +(x|y)+ “x & y = " +(x&y)+
        "   x ^ y = " +(x^y))+"   ~x  = " +(~x)+"\n");

                System.out.println("\nLOGICAL OPERATORS:-\n\t");
                System.out.print("\t(x=10 || y==1) = " +((x==10) || (y==1)));
                System.out.print("   (x=10 && y==1) = " +((x==10) && (y==1)));
                System.out.print("   x != y = " +(x!=y)+"\n\n");

        }
}
```

OUTPUT:



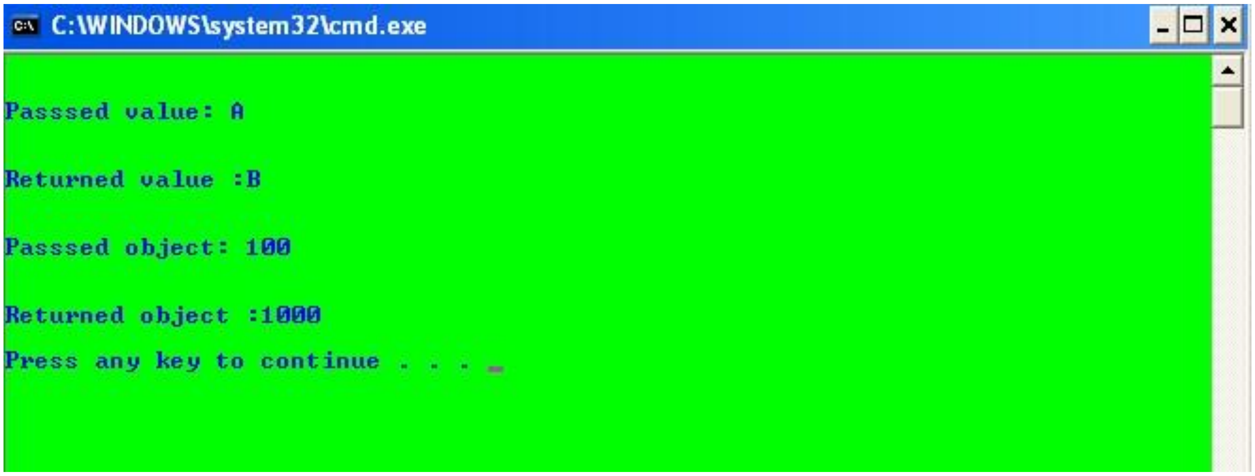
```
ARITHMETIC OPERATORS:-


        x + y = 15   x - y = 5   x * y = 50   x / y = 2
         x  %  y  =  0   x++ = 10   --x = 10

RELATIONAL OPERATORS:-
        Largest number is 10
BITWISE LOGICAL OPERATORS:-
        x | y = 15   x & y = 0   x ^ y = 15   ~x  = -11

LOGICAL OPERATORS:-
        (x=10 || y==1) = true   (x=10 && y==1) = false   x != y = true
Press any key to continue . . .
```

7. Write a program to demonstrate pass by value and passing the objects as parameters.

```
class P77
{
        int x=100;
        public static void main(String args[])
        {
                P77 ob = new P77();             P77 ob1 = new P77();

                char a=ob.retrn('A');
                System.out.println("\n\nReturned value :"+a);

                P77 p7 =ob.pass(ob1);
```

```
                    System.out.println("\n\nReturned object :"+p7.x+"\n");

          }

          P77 pass(P77 obj)
          {
                    System.out.println("\n\nPasssed object: "+obj.x);
                    obj.x=1000;               return obj;
          }

          char retrn(char c)
          {
                              System.out.println("\n\nPasssed value: "+c);
                              c++;                          return(c);
          }
}
```

OUTPUT:

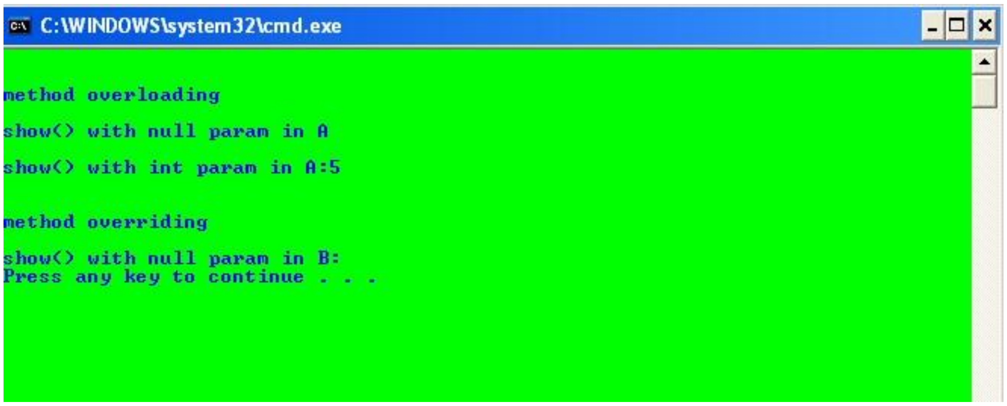


```
Passsed value: A

Returned value :B

Passsed object: 100

Returned object :1000
Press any key to continue . . .
```

8. Write a program to demonstrate method overloading and method overriding.

```
class P8{

          public static void main(String args[]) {

                    System.out.println("method overloading");
A a = new A();
                    a.show();                   a.show(5);
                    System.out.println("method overriding");
B b = new B();                    b.show();
          }
}

class A{

          void show()     {
                    System.out.println("show() with null param in A"); }

          void show(int x){
                    System.out.println("show() with int param in A:"+x);          }
}

class B extends A{

          void show()     {
                    System.out.println("show() with null param in B:"); }
}
```

OUTPUT:

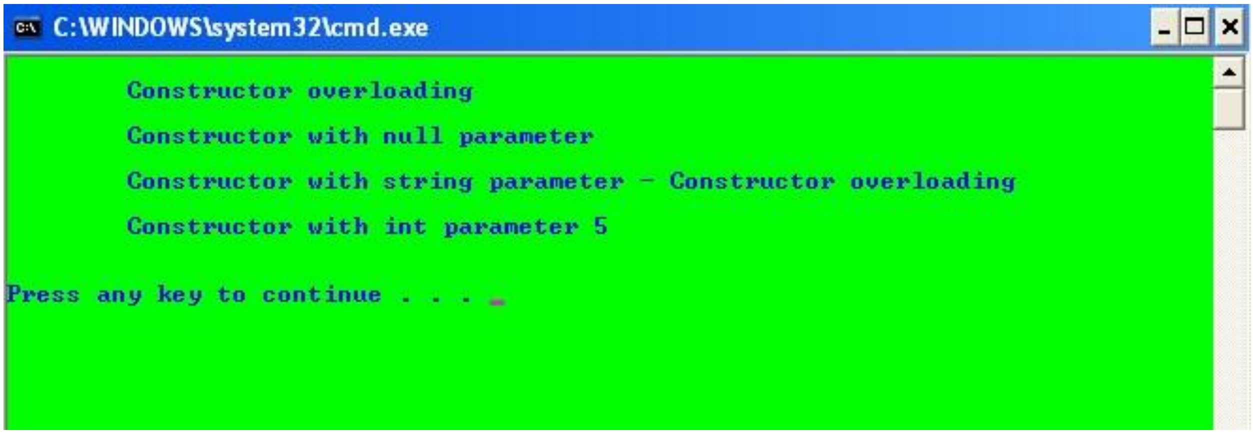


```
method overloading

show() with null param in A

show() with int param in A:5

method overriding

show() with null param in B:
Press any key to continue . . .
```

9. Write a program to demonstrate constructor overloading.

```
class P9
{
        public static void main(String args[])
        {
                System.out.println("\n\tConstructor overloading");
                P9 o1= new P9();
                P9 o2 = new P9(5);
        }

        P9(){
                 System.out.println("\n\tConstructor with null parameter"); }

        P9(int x) {   this("Constructor
overloading");
                System.out.println("\n\tConstructor with int parameter "+x+"\n\n");}

        P9(String x)    {
                System.out.println("\n\tConstructor with string parameter - "+x);    }
}
```

OUTPUT:

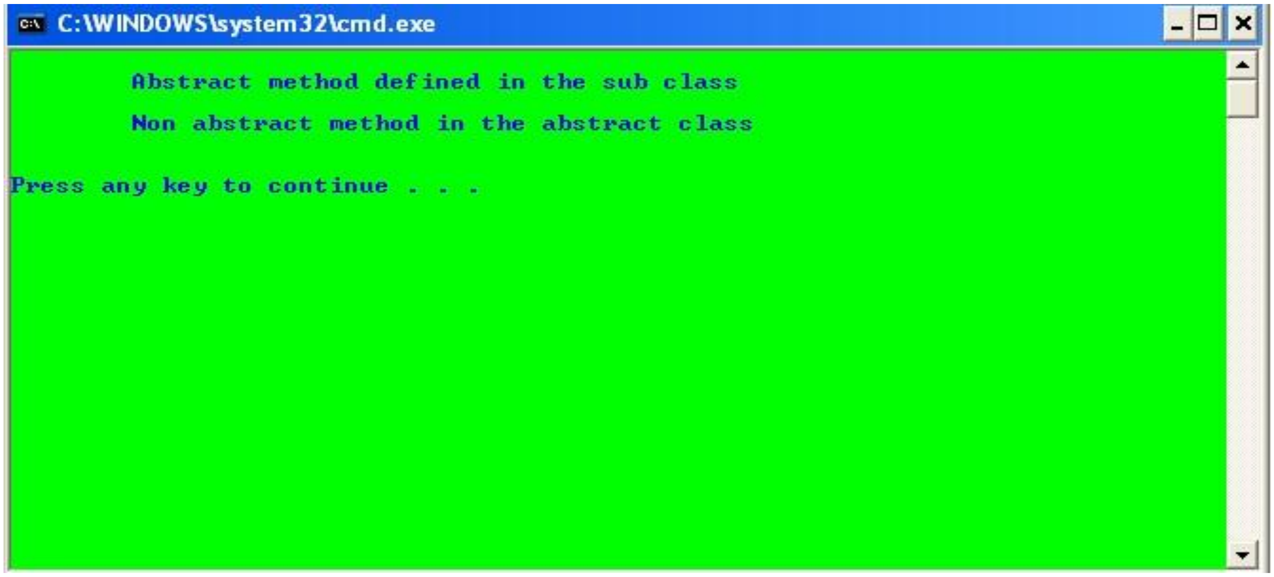


```
        Constructor overloading

        Constructor with null parameter

        Constructor with string parameter - Constructor overloading

        Constructor with int parameter 5

Press any key to continue . . .
```

10. Write a program to use abstract classes.

```
class P10
{
            public static void main(String args[])
            {
                        B b =  new B();
b.show();
                        b.display();
            }
}

abstract class A            {
abstract void show();
                         void display()             {
                                    System.out.println("\Non abstract method in abstract class ");
                         }
}

class B extends A        {
                        void show()                {
                                   System.out.println("\n\tAbstract method defined in the sub class");
                         }
}
```

OUTPUT:

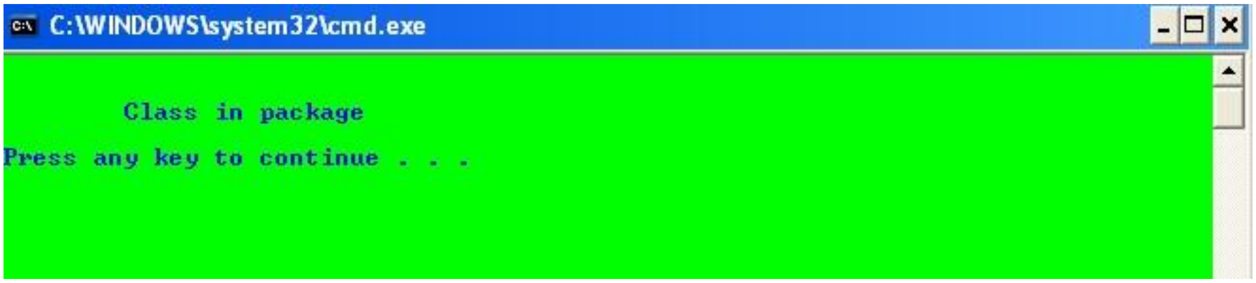


```
        Abstract method defined in the sub class
        Non abstract method in the abstract class

Press any key to continue . . .
```

11. Create a package and import the same in the other class.

```
package pack;

public class packA{

                    public void call(){
                              System.out.println("\n\n\tClass in package\n");                    }
}

import pack.packA;

class P11{

                    public static void main(String args[]) {

                              packA a = new packA();                    a.call();          }
}
```

OUTPUT:

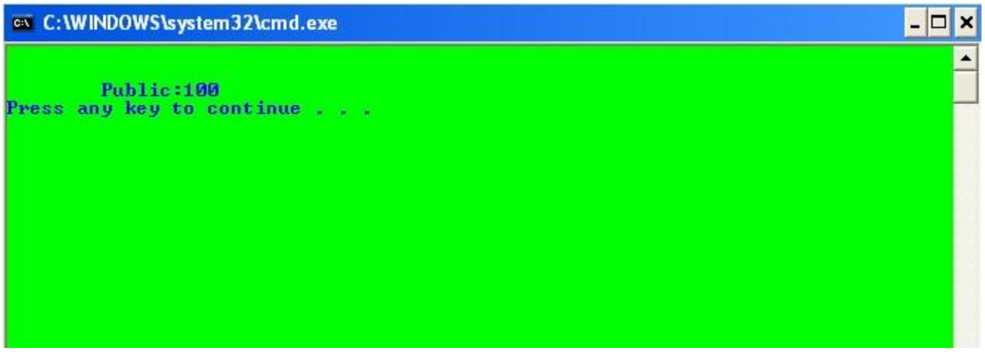


```
        Class in package

Press any key to continue . . .
```

12. Demonstrate the use of access modifiers with the help of packages.

```
package pack; public

class packB {

        void show()     {
                 System.out.println("No modifier:"+y);         }

public void pub()       {

                 System.out.println("Public:"+y);              }

private void priv()     {

               System.out.println("Private :"+y);              }
       protected void prot()    {
               System.out.println("Protected"+y);              }
} import pack.packB; class P12 extends

pack.packB     {        public static void

main(String args[])     {

                packB ob = new packB();       ob.pub();
         }
}
```

OUPUT:

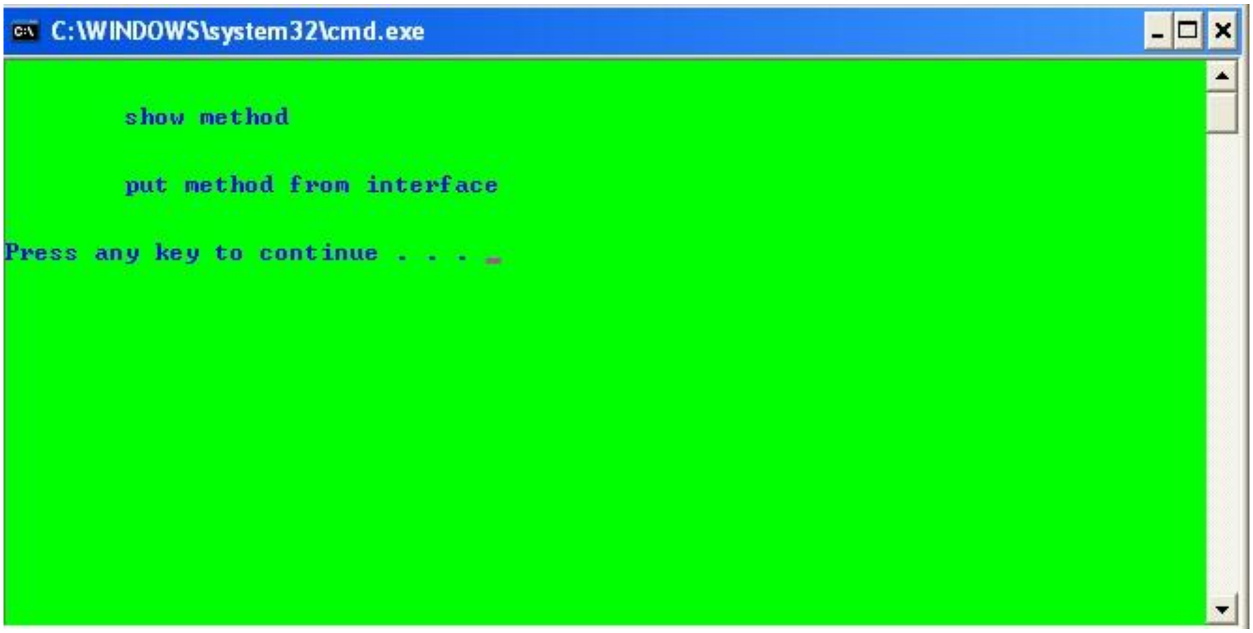


```
        Public:100
Press any key to continue . . .
```

13. Write a program to demonstrate multiple inheritance.

```
interface in      {
          void put();       }
class A{
          void show()     {
                  System.out.println("\n\n\tshow method ");       }
}  class multiple extends A

implements in {

          public void put()          {
                    System.out.println("\n\n\tput method from interface\n\n");   }

public static void main(String args[])           {                   multiple a= new

multiple();

                    a.show();        a.put();           }
}
```

OUPUT:

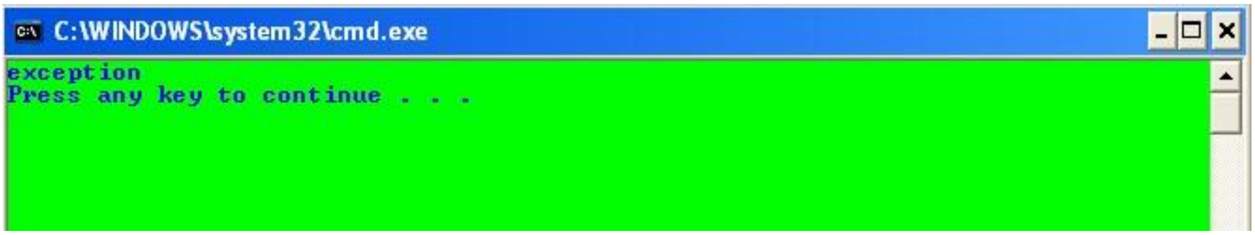


```
        show method

        put method from interface

Press any key to continue . . . _
```

14.Write a program to handle the Exceptions.

```
import java.io.*;

class tryy{

        public static void main(String args[])   {
                BufferedReader br =new BufferedReader(new InputStreamReader(System.in));
                    try       {
                              String s=br.readLine();           }
                   catch(Exception e )      {        System.out.println(e);  }
          }
}
```

OUPUT:

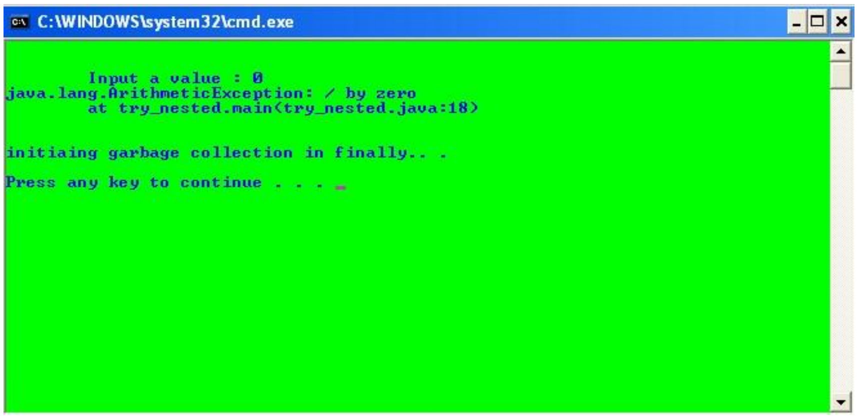


```
exception
Press any key to continue . . .
```

15.Write a program to use finally and nested try block.

```
import java.io.*; class try_nested{   public

static void main(String args[]){

                         int a =10,b;
  BufferedReader br =new BufferedReader(new InputStreamReader(System.in));

System.out.print("\n\n\tInput a value : ");

                         try{
                                    b=Integer.parseInt(br.readLine());
                                      try{
                                                 System.out.println(a/b);            }
                                    catch(ArithmeticException e)             {
                                                 e.printStackTrace();                 }
                         }catch(IOException n) {
                                     n.printStackTrace();                 }

                          finally {
                                  System.out.println("\n\ninitiaing garbage collection in finally.. .\n");
                                      System.gc();                          }
             }}
```

OUPUT:



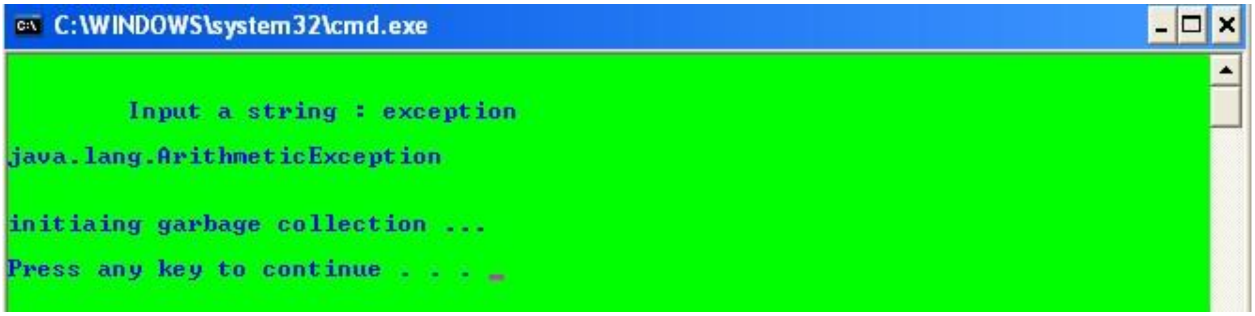
C:\WINDOWS\system32\cmd.exe

```

        Input a value : 0
java.lang.ArithmeticException: / by zero
        at try_nested.main(try_nested.java:18)

initiaing garbage collection in finally.. .

Press any key to continue . . . _
```

16. Demonstrate try, catch, finally, throw and throws in exception handling.

```
import java.io.*;

class try11{

        public static void main(String args[])throws IOException        {
  BufferedReader br =new BufferedReader(new InputStreamReader(System.in));

System.out.print("\n\n\tInput a string : ");

                          try{
   String s=br.readLine();    throw new ArithmeticException();  }

                          catch(ArithmeticException ie){System.out.println("\n"+ie);}

catch(NullPointerException ie){System.out.println(ie);}

                          finally {
                                  System.out.println("\n\ninitiaing garbage collection ...\n");

                                     System.gc();                }
             }
}
```

OUPUT:

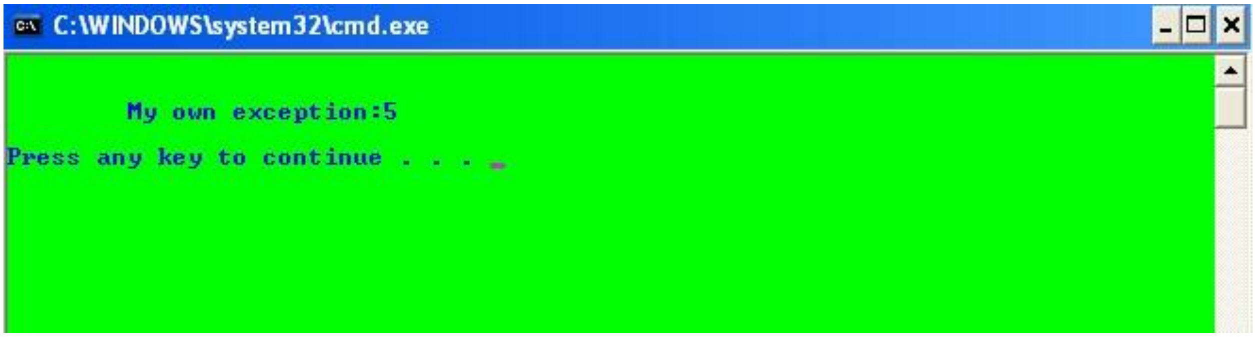


```
        Input a string : exception
java.lang.ArithmeticException

initiaing garbage collection ...
Press any key to continue . . .
```

17. Create your own exception class and throw the same in the other class.

```
class myownexcep1 extends Exception           {
           myownexcep1(int c)                        {
                   System.out.println("\n\n\tMy own exception:"+c);                 }
          public static void main(String args[])             {
                      try{
                                throw new myownexcep1(5);           }

  catch(Exception e)     {         System.out.println();   }

            }
  }
```

OUPUT:

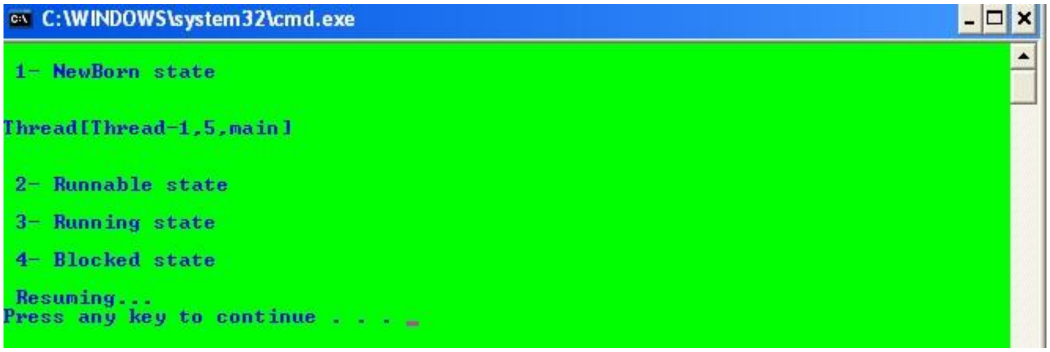


```


        My own exception:5
Press any key to continue . . . _
```

18.WAP to demonstrate the life cycle of Thread.

```
class thrd_lc extends Thread{
        thrd_lc t1;     public void

run()   {

              System.out.println("\n 3- Running state");
               for(int i=0;i<5;i++)     {
                       if(i==1)
                               try{  System.out.println("\n 4- Blocked state");
                                       t1.sleep(10); System.out.println("\n Resuming...");
                              } catch(Exception e){}
                       }
               }
       void show()     {
 System.out.println("\n 1- NewBorn state");    t1 = new thrd_lc();

System.out.println("\n\n"+t1+"\n");

              System.out.println("\n 2- Runnable state");     t1.start();
        }
public static void main(String args[])  {  thrd_lc ob= new

thrd_lc(); ob.show(); }

}
```

OUPUT:

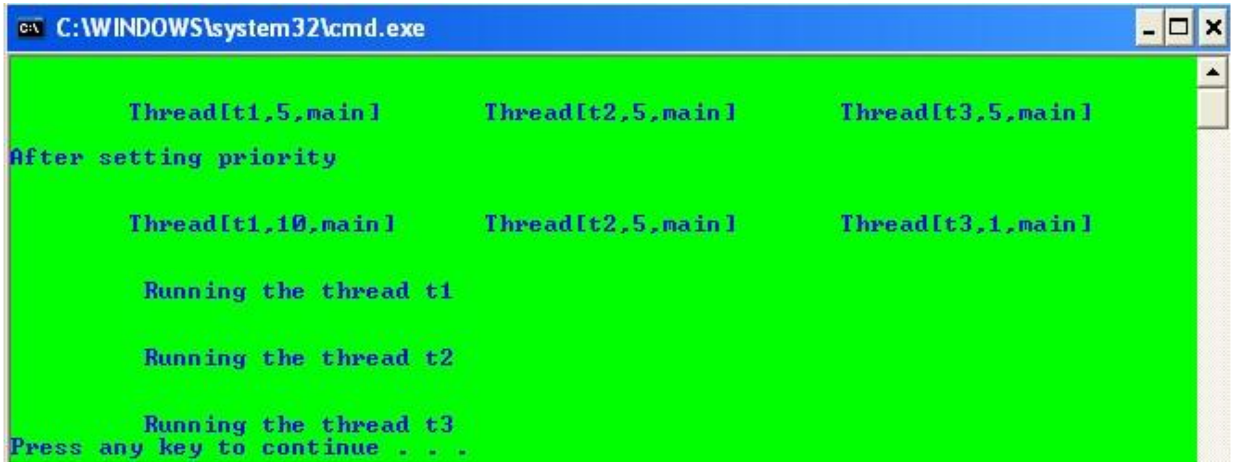


```
 1- NewBorn state

Thread[Thread-1,5,main]

 2- Runnable state

 3- Running state

 4- Blocked state

 Resuming...
Press any key to continue . . .
```

19.WAP to create multiple threads and set priorities.

```
public   class   thrdpriority   extends   Thread{

String th;   thrdpriority(String s)  {

              super(s);
             System.out.print(this+"\t");
              th=s;               }
  public static void main(String[] args)            {

thrdpriority t1 = new thrdpriority("t1");         thrdpriority t2 =

new thrdpriority("t2");             thrdpriority t3 = new

thrdpriority("t3");

t1.setPriority(10);t2.setPriority(5);t3.setPriority(1);

System.out.println("\nAfter setting priority");

System.out.println("\n\n\t"+t1+"\t"+t2+"\t"+t3);        t1.start();

t2.start();t3.start();                     }   public void run()    {

              System.out.println("\n\n\t Running the thread "+th);  } }
```

OUPUT:



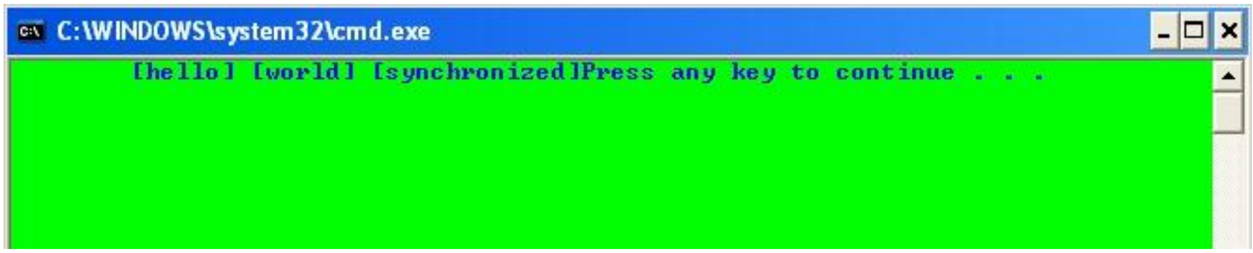
```
        Thread[t1,5,main]       Thread[t2,5,main]       Thread[t3,5,main]
After setting priority

        Thread[t1,10,main]      Thread[t2,5,main]       Thread[t3,1,main]

         Running the thread t1

         Running the thread t2

         Running the thread t3
Press any key to continue . . .
```

20.WAP to demonstrate thread synchronization.

```
class A{
        synchronized void call(String msg)          {

System.out.print("\t["+msg);

                try{    Thread.sleep(1000);         }

catch(Exception e){}

               System.out.print("]");                 }       }
class B implements Runnable                   {
        String msg; Thread t; A obj;   B(String s, A obj1) {

                obj =obj1;      msg=s;
 t= new Thread(this); t.start();  }  public void run()   { obj.call(msg);

} } }

class synceg    {       public static void main(String

args[]) {

A obj = new A();

B ob1=new B("hello",obj);
```

```
                  B ob2=new B("synchronized",obj);
                   B ob3=new B("world",obj);    }         }
```

OUPUT:

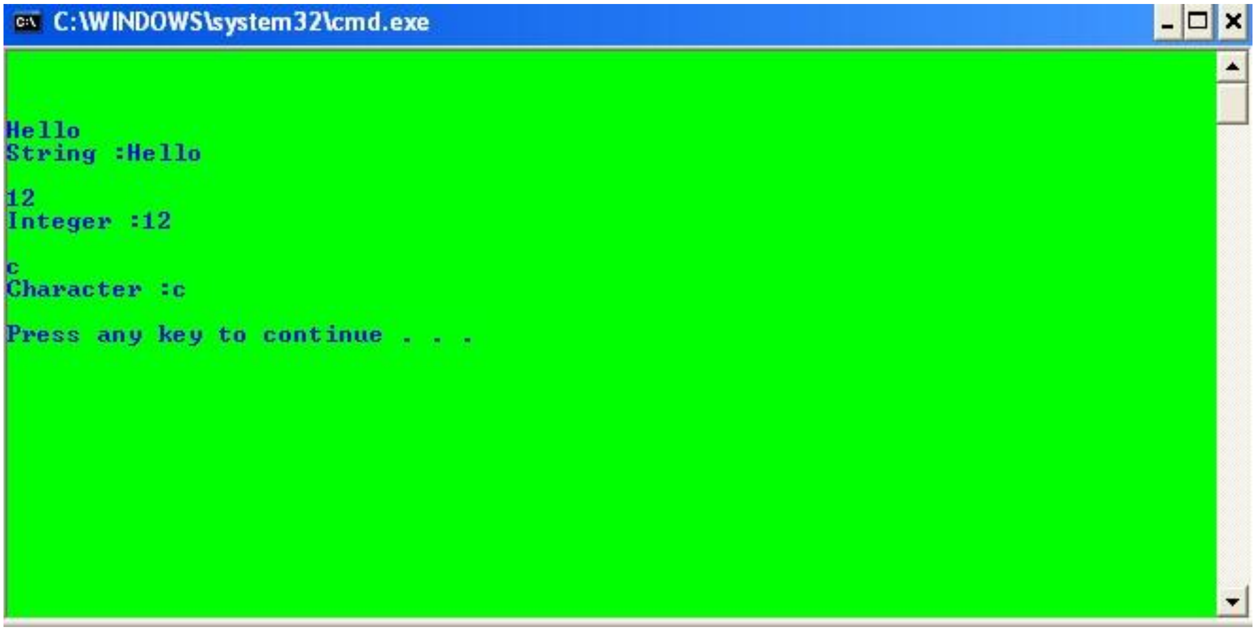


```
[hello] [world] [synchronized]Press any key to continue . . .
```

21.WAP to take input from the user using basic IO and display the output.

```
import java.io.*; class printWriter{

public static void main(String kl[])throws IOException {

  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

 PrintWriter pw = new PrintWriter(System.out,true);   pw.println("\n\n\t");

                    String s = br.readLine();        pw.println("String :"+s+"\n\t");

int i = Integer.parseInt(br.readLine());           pw.println("Integer :"+i+"\n\t");

char c = (char)(br.read());                  pw.println("Character :"+c+"\n\t");

                    }
}
```

OUPUT:

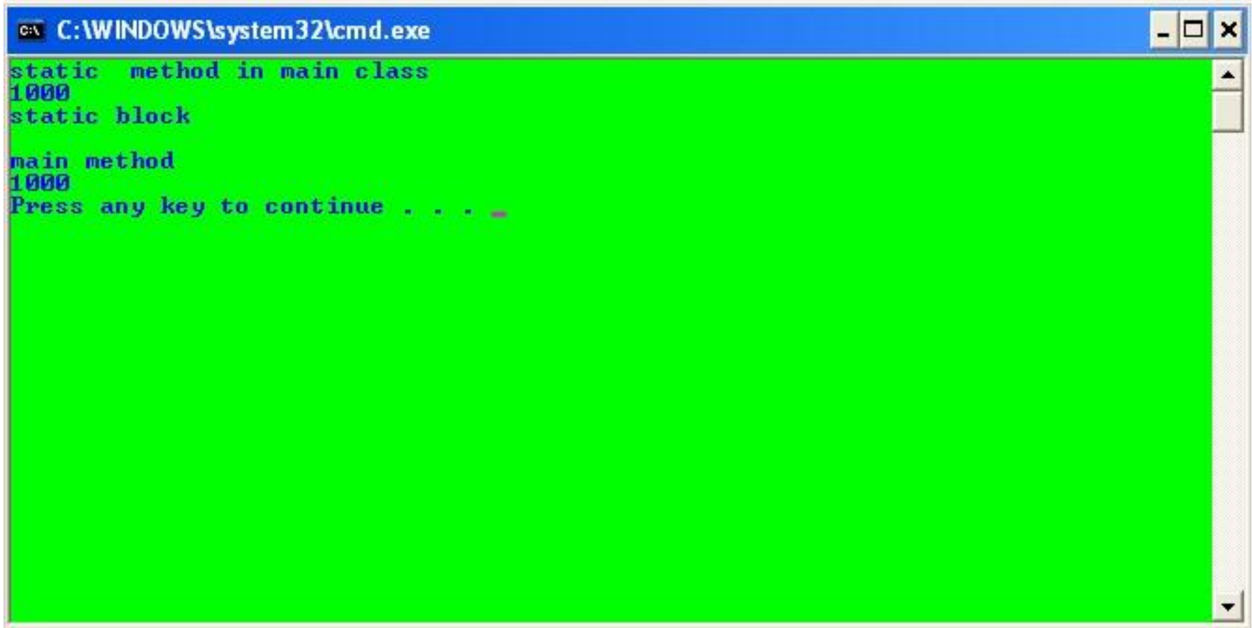


```
Hello
String :Hello

12
Integer :12

c
Character :c

Press any key to continue . . .
```

22.WAP to demonstrate the use of static methods & blocks.

```
class statc{
  static int y=1000;   static void call() {

                        System.out.println("static  method in main class\n");            }

public static void main(String args[])            {

                      System.out.println("main method\n"+y);          }
            static    {
                        call();
                      System.out.println("static block\n");     }
}
```

OUPUT:

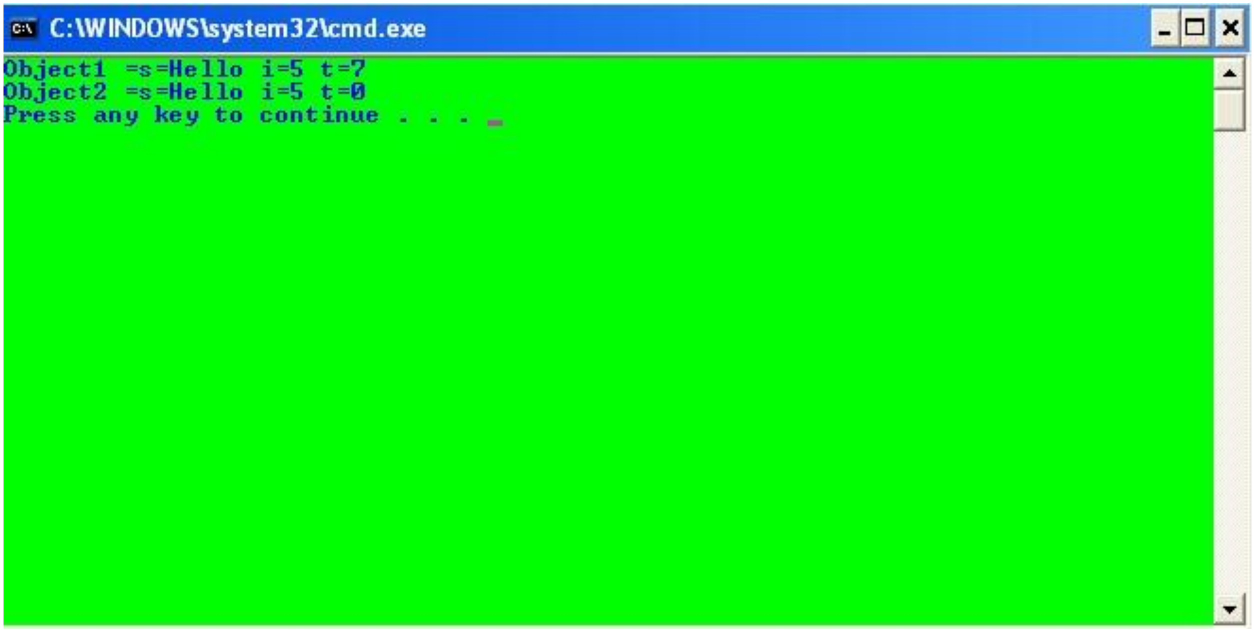


```
static  method in main class
1000
static block

main method
1000
Press any key to continue . . .
```

23. Demonstrate the use of transient in serialization.

```
import java.io.*; public class serialization{

        public static void main(String rags[])   {
                    try       {
                            A obj1 = new A("Hello",5,7);
                            System.out.println("Object1 ="+obj1);
  FileOutputStream         fout         =new         FileOutputStream("Aaa");

ObjectOutputStream         oos=         new         ObjectOutputStream(fout);

oos.writeObject(obj1);    oos.flush();   oos.close();

                   }catch(Exception e)              {
                              System.out.println("Exception in serialization"); }
                    try       {
                              A obj2;
  FileInputStream          fin          =new          FileInputStream("Aaa");

ObjectInputStream   ois=   new   ObjectInputStream(fin);          obj2   =

(A)ois.readObject();

                              ois.close();
                            System.out.println("Object2 ="+obj2);
                   }catch(Exception e)     {
                           System.out.println("Exception in deserialization");      }

          }
}



class A implements Serializable{
          String s;          int i;    transient int t;

public A(String s1, int i1,int t1) {

                   this.s=s1;                    this.i=i1;                    this.t=t1;
          }
         public String toString() {
                  return "s="+s+" i="+i+" t="+t;             }
}
```

OUPUT:

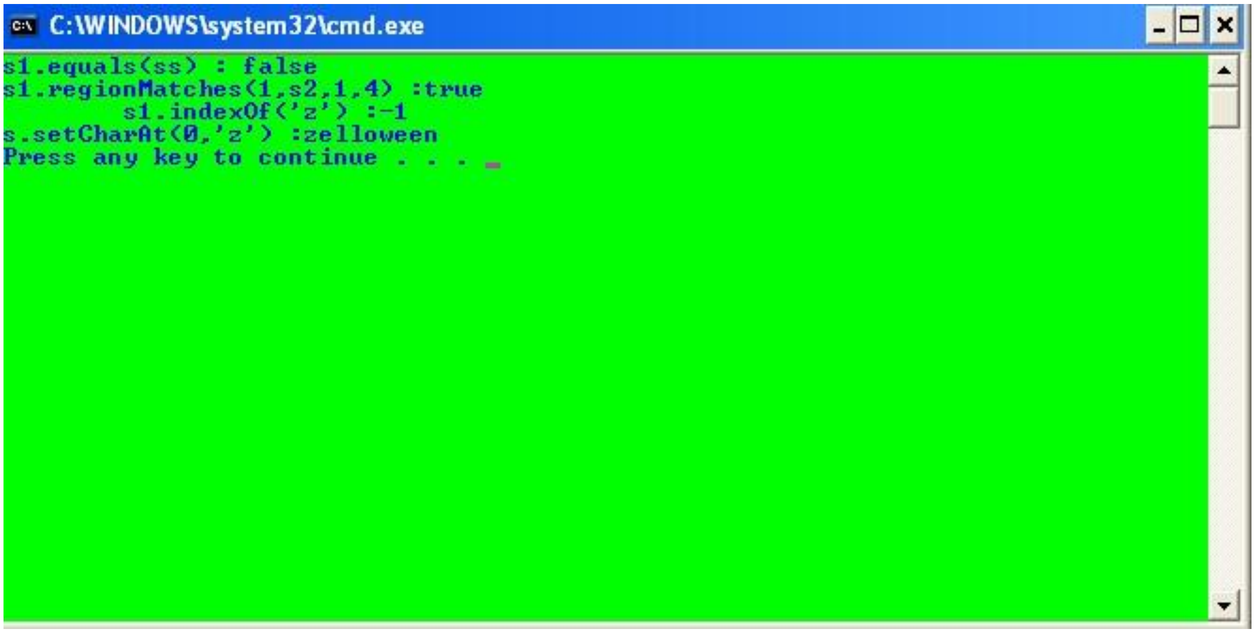


```
Object1  =s=Hello  i=5  t=7
Object2  =s=Hello  i=5  t=0
Press any key to continue . . .
```

24. Demonstrate the usage of string and stringbuffer.

```
class strng      {
        public static void main(String args[])   {
                  String s1 = new String("Hello");
                    String ss="hello";
                    String s2 = new String("Helloween");
                    System.out.println("s1.equals(ss) : "+s1.equals(ss));


                    StringBuffer s = new StringBuffer("Helloween");
                    System.out.println("s1.regionMatches(1,s2,1,4) :"+s1.regionMatches(1,s2,1,4));
                   System.out.println("    s1.indexOf('z') :"+      s1.indexOf('z'));
                    System.out.println("s.setCharAt(0,'z') :"+ s.setCharAt(0,'z'));
} }
```

OUPUT:

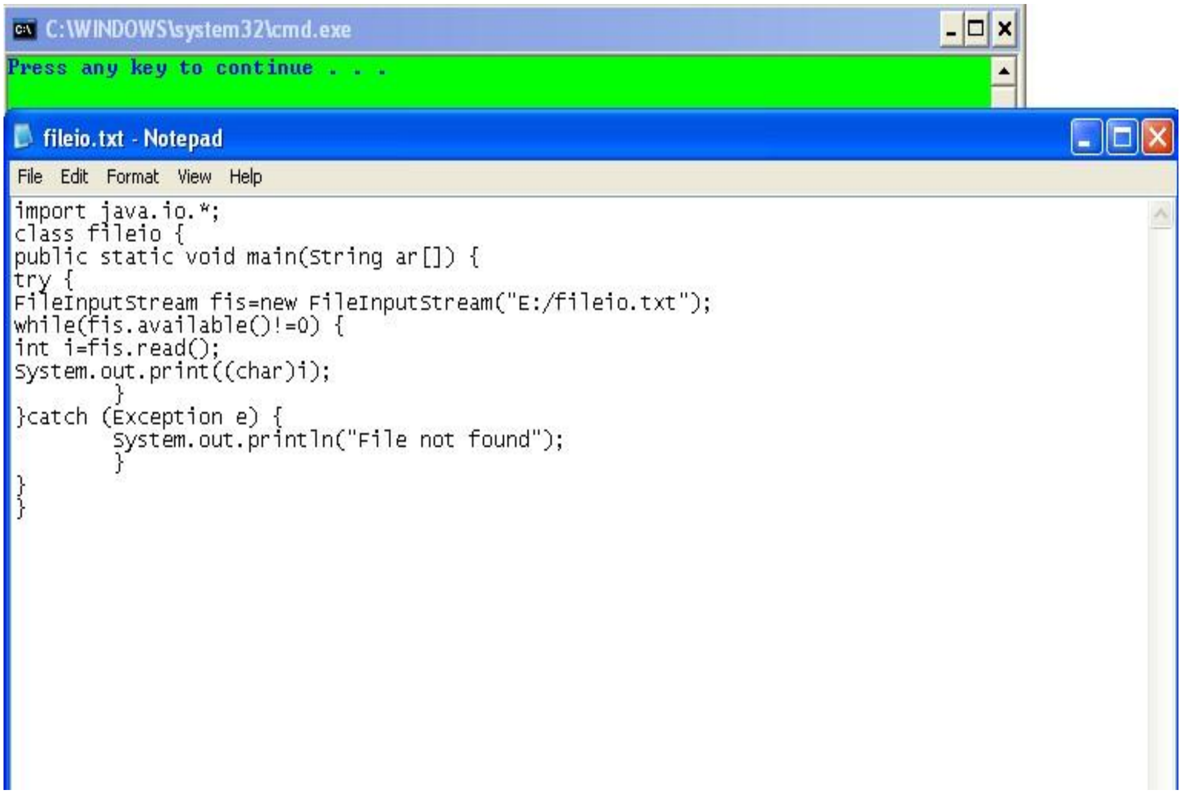


```
s1.equals(ss) : false
s1.regionMatches(1,s2,1,4) :true
        s1.indexOf('z') :-1
s.setCharAt(0,'z') :zelloween
Press any key to continue . . .
```

25.WAP to create a process and display the output given by that process.

```
class procssbldr
{       public static void main(String args[])throws

Exception

        {
            ProcessBuilder r = new ProcessBuilder("notepad.exe","E:/fileio.txt");
               r.start();
        }
}
```

OUPUT:



```
Press any key to continue . . .
```

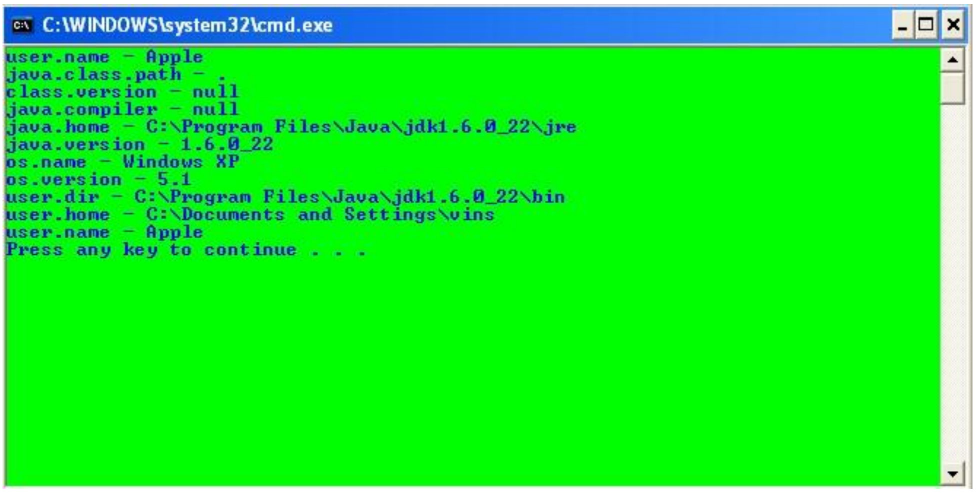
```
import java.io.*;
class fileio {
public static void main(String ar[]) {
try {
FileInputStream fis=new FileInputStream("E:/fileio.txt");
while(fis.available()!=0) {
int i=fis.read();
System.out.print((char)i);
        }
}catch (Exception e) {
        System.out.println("File not found");
        }
}
}
```

26.WAP to set the System properties.

```
class Systm {  public static void main(String args[])throws

Exception {

          System.setProperty("user.name","Apple");
         System.out.println("user.name - "+System.getProperty("user.name"));
        System.out.println("java.class.path - "+System.getProperty("java.class.path"));
        System.out.println("class.version - "+System.getProperty("class.version"));
        System.out.println("java.compiler - "+System.getProperty("java.compiler"));
         System.out.println("java.home - "+System.getProperty("java.home"));
         System.out.println("java.version - "+System.getProperty("java.version"));
         System.out.println("os.name - "+System.getProperty("os.name"));
         System.out.println("os.version - "+System.getProperty("os.version"));
```

```
                    System.out.println("user.dir - "+System.getProperty("user.dir"));
                    System.out.println("user.home - "+System.getProperty("user.home"));
                    System.out.println("user.name - "+System.getProperty("user.name"));
                      }
          }
```

OUPUT:



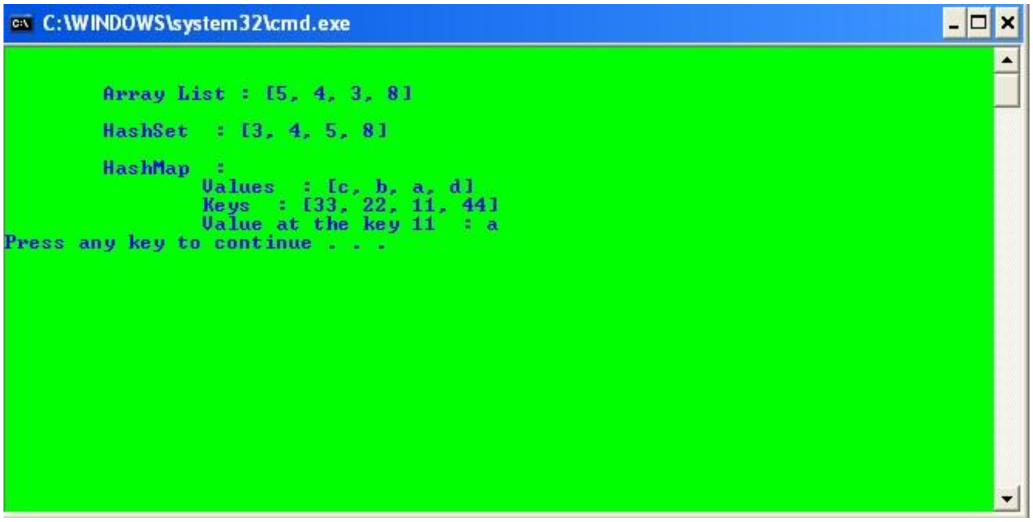
C:\WINDOWS\system32\cmd.exe

```
user.name - Apple
java.class.path - .
class.version - null
java.compiler - null
java.home - C:\Program Files\Java\jdk1.6.0_22\jre
java.version - 1.6.0_22
os.name - Windows XP
os.version - 5.1
user.dir - C:\Program Files\Java\jdk1.6.0_22\bin
user.home - C:\Documents and Settings\vins
user.name - Apple
Press any key to continue . . .
```

27. WAP to use arrayList, sort the elements, retrieve the values and the keys from the map.

```
import java.util.*; class p29{            public static

void main(String args[])        {

ArrayList a = new ArrayList<Integer>();

                    a.add(5);         a.add(4);         a.add(3);         a.add(8);
                   System.out.println("\n\n\tArray List : "+a);
                    HashSet hs= new HashSet(a);
                     System.out.println("\n\tHashSet  : "+hs);

HashMap hm =  new  HashMap();                hm.put(11,"a");

hm.put(22,"b");

                    hm.put(33,"c");          hm.put(44,"d");
                   System.out.println("\n\tHashMap  : ");
                   System.out.println("\t\tValues  : "+hm.values());
                   System.out.println("\t\tKeys  : "+hm.keySet());
                  System.out.println("\t\tValue at the key 11  : "+hm.get(11));
          }}
```

OUPUT:

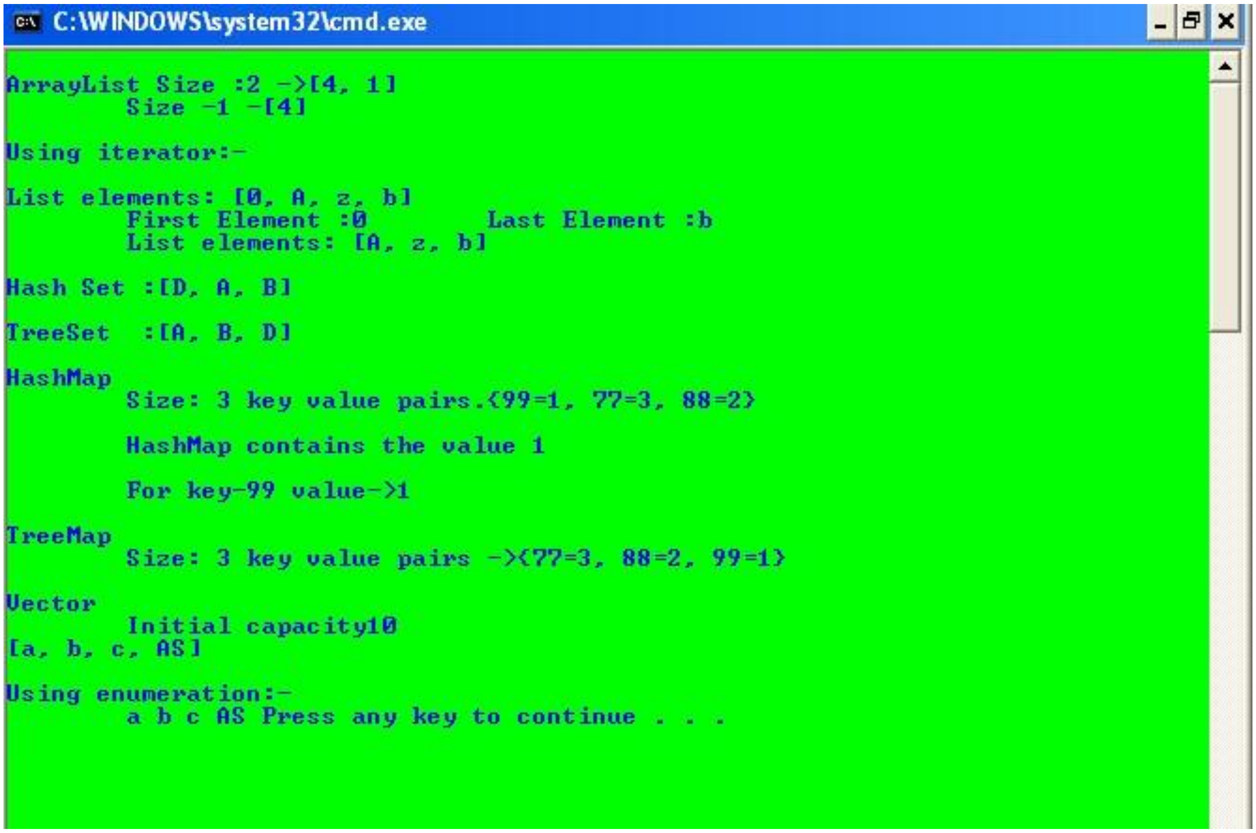


```
        Array List : [5, 4, 3, 8]

        HashSet  : [3, 4, 5, 8]

        HashMap  :
                Values  : [c, b, a, d]
                Keys  : [33, 22, 11, 44]
                Value at the key 11  : a
Press any key to continue . . .
```

28. WAP to work with Collection interfaces & classes.

```
import  java.util.*;  class  p30{    public  static

void main(String args[]) {

                ///////////////////////////////ArrayList/////////////////////////////////////////
                ArrayList<Integer> al = new ArrayList<Integer>();
                al.add(4);                al.add(1);
                System.out.println("\nArrayList Size :"+al.size()+" ->"+al);
                al.remove(1);
                System.out.println("\tSize -"+al.size()+" -"+al);
                /////////////////////////////////Iterator///////////////////////////////////////////////
                System.out.println("\nUsing iterator:-");

        Iterator it =al.iterator();

                while(it.hasNext())     {
                        if(it.next().equals("5"))      System.out.println("found");      }
                //////////////////////////////////LinkedList/////////////////////////////////////////////////////
                LinkedList l = new LinkedList();
                l.add("A");     l.addFirst("0");        l.addLast("z");      l.add("b");
                System.out.println("\nList elements: "+l);
                System.out.print("\tFirst Element :"+l.getFirst());

        System.out.println("\tLast Element :"+l.getLast());

                l.remove();     System.out.println("\tList elements: "+l);
        ////////////////////////////////HashSet///////////////////////////////////////////////////
                Set<String> set = new HashSet<String>();
                set.add("B");           set.add("A");           set.add("D");
                System.out.println("\nHash Set :"+set);
                /////////////////////////////////////TreeSet//////////////////////////////////////////////////
                TreeSet<String> ts = new TreeSet<String>();
                ts.add("B");            ts.add("A");            ts.add("D");
                System.out.println("\nTreeSet  :"+ts);
                /////////////////////////////////////HashMap////////////////////////////////////////
                HashMap hm = new HashMap();
                hm.put( 99, 1 );        hm.put( 88, 2 );        hm.put( 77, 3 );
                System.out.println("\nHashMap \n\tSize: " + hm.size() + " key value
                pairs."+hm);
                if( hm.containsValue(1) )
                        System.out.println("\n\tHashMap contains the value 1");
                if( hm.containsKey(99))
                        System.out.println("\n\tFor key-99 value->"+hm.get(99));
                ////////////////////////////////TreeMap//////////////////////////////////////////////////////////
                TreeMap tm = new TreeMap();
                tm.put( 99, 1 );        tm.put( 88, 2 );        tm.put( 77, 3 );
                System.out.println("\nTreeMap \n\tSize: " + tm.size() + " key value pairs
                >"+tm);
                /////////////////////////////////Vector/////////////////////////////////////////////////////////
                Vector v = new Vector();
                System.out.println("\nVector \n\tInitial capacity"+v.capacity());

                v.addElement("a");          v.addElement("b");
                v.addElement("c");          v.add("AS");
                System.out.println(v);
            /////////////////////////////////Enumeration/////////////////////////////////////////////////////////
                System.out.print("\nUsing enumeration:-\n\t");

        Enumeration e =v.elements();

        while(e.hasMoreElements())

                        System.out.print(e.nextElement()+ " ");
        }
}
```

OUPUT:

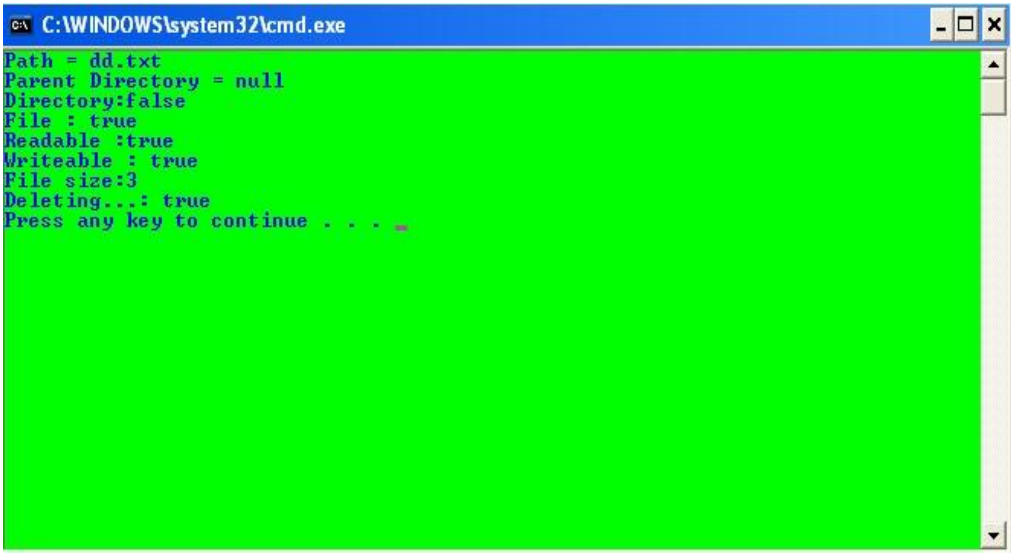


```
ArrayList Size :2 ->[4, 1]
        Size -1 -[4]

Using iterator:-

List elements: [0, A, z, b]
        First Element :0        Last Element :b
        List elements: [A, z, b]

Hash Set :[D, A, B]

TreeSet  :[A, B, D]

HashMap
        Size: 3 key value pairs.{99=1, 77=3, 88=2}

        HashMap contains the value 1

        For key-99 value->1

TreeMap
        Size: 3 key value pairs ->{77=3, 88=2, 99=1}

Vector
        Initial capacity10
[a, b, c, AS]

Using enumeration:-
        a b c AS Press any key to continue . . .
```

29. WAP to retrieve all the file properties and display the same.

```
import java.io.*;
class file{
        public static void main(String kl[])    {
                File f=new File("dd.txt");
                System.out.println("Path = "+f.getPath());
                System.out.println("Parent Directory = "+f.getParent());
                System.out.println("Directory:"+f.isDirectory());
                System.out.println("File : "+f.isFile());
                System.out.println("Readable :"+f.canRead());
                System.out.println("Writeable : "+f.canWrite());
                System.out.println("File size:"+f.length());
                System.out.println("Deleting...: "+f.delete());
}}
```

OUTPUT:



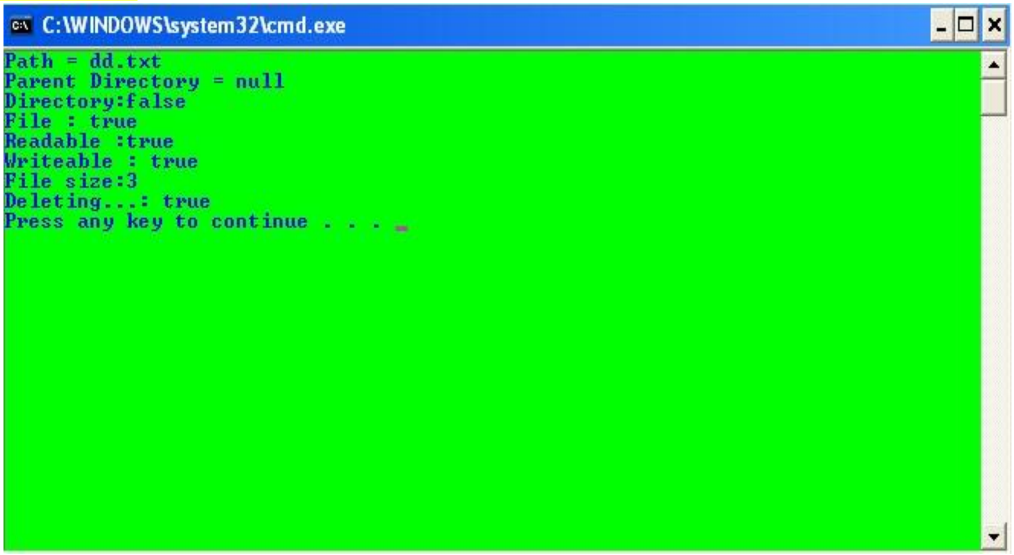
```
Path = dd.txt
Parent Directory = null
Directory:false
File : true
Readable :true
Writeable : true
File size:3
Deleting...: true
Press any key to continue . . .
```

30. WAP to demonstrate the various input and output streams(Character Streams, Byte Streams, Object Streams)

```
import java.io.*;

class file{

        public static void main(String kl[])    {

                File f=new File("dd.txt");

                System.out.println("Path = "+f.getPath());

                System.out.println("Parent Directory = "+f.getParent());

                System.out.println("Directory:"+f.isDirectory());

                System.out.println("File : "+f.isFile());

                System.out.println("Readable :"+f.canRead());

                System.out.println("Writeable : "+f.canWrite());

                System.out.println("File size:"+f.length());
```

```
                        System.out.println("Deleting...: "+f.delete());
        }}
```

OUTPUT:



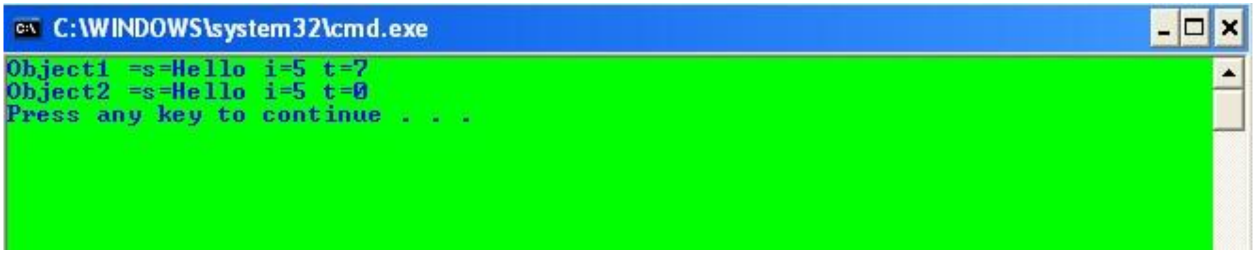
```
Path = dd.txt
Parent Directory = null
Directory:false
File : true
Readable :true
Writeable : true
File size:3
Deleting...: true
Press any key to continue . . .
```

31. WAP for serializing and de serializing the objects.

```
        import java.io.*;
        public class serialization        {
        public static void main(String rags[])            {
                try       {
                        A obj1 = new A("Hello",5,7);
                        System.out.println("Object1 ="+obj1);
                        FileOutputStream fout =new FileOutputStream("Aaa");

ObjectOutputStream oos= new ObjectOutputStream(fout);

                        oos.writeObject(obj1);

                oos.flush();

oos.close();

                }catch(Exception e)    {
                        System.out.println("Exception in serialization");      }
                try       {
                        A obj2;
                        FileInputStream fin =new FileInputStream("Aaa");

ObjectInputStream ois= new ObjectInputStream(fin);

                        obj2 = (A)ois.readObject();
                        ois.close();
                        System.out.println("Object2 ="+obj2);
                }catch(Exception e)    {
                         System.out.println("Exception in deserialization");   }

        }
        }
        class A implements Serializable        {
 String s; int i;  transient int t;   public A(String s1, int i1,int t1) {
```

```
                    this.s=s1;          this.i=i1;          this.t=t1;      }
          public String toString()          {
                    return "s="+s+" i="+i+" t="+t;          }
}
```

OUTPUT:



```
Object1  =s=Hello  i=5  t=7
Object2  =s=Hello  i=5  t=0
Press any key to continue . . .
```