```
##Example1
public class ex1 {
    public static void main(String[] args) {

  System.out.print("print() print in");
System.out.print("same line");
System.out.println("While");
System.out.println("print() prints the line with carriage return at the end");
System.out.print("we can also have carriage return");
System.out.println("\n by giving \\n");

    }
}
o/p
PS C:\Users\Admin\Desktop\sycs-41\prac1> javac ex1.java
PS C:\Users\Admin\Desktop\sycs-41\prac1> java ex1
print() print insame lineWhile
print() prints the line with carriage return at the end
we can also have carriage return
 by giving \n

--------------------------------------------------------------------------------
------------------------

##example 2
public class ex2 {
    public static void main(String[] args)
{

int i= Integer.parseInt(args[0]);
int j= Integer.parseInt(args[1]);
if(i>j){
System.out.println(i+" is greater than "+j);
}
else
{
System.out.println(j+" is greater than "+i);
}
    }
}

o/p
PS C:\Users\Admin\Desktop\sycs-41\prac1> javac ex2.java
PS C:\Users\Admin\Desktop\sycs-41\prac1> java ex2 3 6
6 is greater than 3
--------------------------------------------------------------------------------
----------

##Practical-1a
class Myclass
 {
private int a;
    public Myclass()
{
System.out.println("default constructor");
           }
public Myclass(int value)
{
a=value;
System.out.println("parameter constructor and value is :
"+a);
}
public Myclass(Myclass other)
{
```

```
a=other.a;
System.out.println("copy constructor and value is : "+a);
}
}
public class pr
{
public static void main(String args[])
{
Myclass bj1=new Myclass();
Myclass bj2=new Myclass(5);
Myclass bj3=new Myclass(bj2);


}
}
```

o/p

```
PS C:\Users\Admin\Desktop\sycs-41> javac pr.java
PS C:\Users\Admin\Desktop\sycs-41> java pr
default constructor
parameter constructor and value is : 5
copy constructor and value is : 5
```

##practical 1b ##overloading

```
class overloaddemo
{
void addition()
{
System.out.println("test function with no parameters");
}
void addition(int a)
{
System.out.println("the value of parameter is a : " +a);
}
void addition(int a,int b)
{
System.out.println("the valueof a is" +a+ "the value of b is : "+b);
}
double addition(double a , double b)
{
System.out.println("the value of a is : " +a);
return a*b;
}
}
class ol
{
public static void main(String[] args)
{
overloaddemo obj=new overloaddemo();
double result;
obj.addition();
obj.addition(10);
obj.addition(10,20);
result=obj.addition(12.3,1.2);
System.out.println("the result is : " + result);
}
}
```

o/p

```
PS C:\Users\Admin\Desktop\sycs-41\prac2> javac ol.java
PS C:\Users\Admin\Desktop\sycs-41\prac2> java ol
test function with no parameters
```

```
the value of parameter is a : 10
the valueof a is10the value of b is : 20
the value of a is : 12.3
the result is : 14.76
--------------------------------------------------------------------------------
---------
## practical 1c ##static method

class student
{
int rollno;
String name;
static String college="sn";
static void change()
{
college="Abhinav";
}
student(int r,String n)
{
 rollno=r;
name=n;
}
void display()
{
System.out.println(rollno+" " +name+ " "+college);
}
}
class teststa
{
 public static void main(String args[])
{
student.change();
student s1=new student(41,"Shubham");
student s2=new student(42,"Nilesh");
student s3=new student(43,"Gaurav");
s1.display();
s2.display();
s3.display();

}
}

o/p
PS C:\Users\Admin\Desktop\sycs-41\prac2> javac teststa.java
PS C:\Users\Admin\Desktop\sycs-41\prac2> java teststa
41 Shubham Abhinav
42 Nilesh Abhinav
43 Gaurav Abhinav

--------------------------------------------------------------------------------
-----------------
##prac-1a
--------------------------------------------------------------------------------
#code
-----

class Example{
int a,b;
public Example()
{
this.a =0;
this.b =0;
System.out.println("Default Constructor : a ="+a+",b="+b);
}
```

```
public Example(int a)
{
this.a=a;
this.b=0;
System.out.println("Single parameter Constructor : a ="+a+",b="+b);

}
public Example(int a, int b)
{
this.a=a;
this.b=b;
System.out.println("two parameter Constructor : a ="+a+",b="+b);

}

public void display(){
System.out.println("Display with no parameters: a ="+a+",b="+b);
}

public void display(int a){
System.out.println("Display with one parameters: a ="+a+",b="+b);
}
public void display(int a, int b){
System.out.println("Display with one parameters: a ="+a+",b="+b);
}

public static void staticMethod(){
System.out.println("this is a static method");
}

public static void main(String[] args)
{
Example obj1 = new Example();
Example obj2 = new Example(5);
Example obj3 = new Example(5,10);

obj1.display();
obj1.display(7);
obj1.display(7,14);
Example.staticMethod();
}
}

o/p
PS C:\Users\Admin\Desktop\sycs-41\prac3> javac Example.java
PS C:\Users\Admin\Desktop\sycs-41\prac3> java Example
Default Constructor : a =0,b=0
Single parameter Constructor : a =5,b=0
two parameter Constructor : a =5,b=10
Display with no parameters: a =0,b=0
Display with one parameters: a =7,b=0
Display with one parameters: a =7,b=14
this is a static method
--------------------------------------------------------------------------------
----------------------------31-07-2024---------
##prac-1b
------------
wap to implement the concept of inheritance and method overriding
--------------------------------------------------------------------
#code
-----
class A
{
 void show()
```

```
{
System.out.println("base class");
}
}

class B extends A
{

void show()
{
System.out.println("Derived Class");
}

}

 class pr1b
{
public static void main(String[] args)
{
B s= new B();
A s1= new A();
s.show();
s1.show();
}

}

o/p
----
PS C:\Users\Admin\Desktop\sycs-41\prac3> javac pr1b.java
PS C:\Users\Admin\Desktop\sycs-41\prac3> java pr1b
Derived Class
base class

----------------------practical-
2------------------------------------------------------------
##prac-2a
------------
wap implement the concept of abstract classes and methods
-----------------------------------------------------------------
#code
------

abstract class shape
{
public abstract double area();
}

class circle extends shape
{
  private double radius;
public circle(double radius)
{
 this.radius=radius;
}
//@override
public double area()
{
return Math.PI*radius*radius;
}
}
class pr2a
{
public static void main(String[] args)
```

```
{
circle c=new circle(10.0);
System.out.println("circle area is : "+c.area());

}

}
```

o/p

```
PS C:\Users\Admin\Desktop\sycs-41\prac3> javac pr2a.java
PS C:\Users\Admin\Desktop\sycs-41\prac3> java pr2a
circle area is : 314.1592653589793
```

##2b

write a program to impliement interface

#code

```
interface shape
{
 double area();
 double perimeter();
}
class circle implements shape
{
private double ra;
public circle(double ra)
{
this.ra=ra;
}
public double area()
{
return Math.PI*ra*ra;
}
public double perimeter()
{
return 2*Math.PI*ra*ra;
}
}
public class pr2b
{
   public static void main(String args[])
   {
    circle c=new circle(10.0);
    System.out.println("Area of circle is "+c.area());
    System.out.println("circle perimeter is"+c.perimeter());
   }
}
```

o/p

```
PS C:\Users\Admin\Desktop\sycs-41\prac4> javac pr2b.java
PS C:\Users\Admin\Desktop\sycs-41\prac4> java pr2b
Area of circle is 314.1592653589793
circle perimeter is628.3185307179587
```

##3A

---

write a program to define userdefine Exception and raise them as per requirement

---

#code

```
import java.util.*;
class CustomException extends Exception
{
 public CustomException(String message)
{
super(message);
}
}
public class pr3
{
 public static void main(String args[])
{
  try
{
int age=-20;
// Scanner sc = new Scanner(System.in);
// System.out.println("Enter the age");
if(age<0)
{
  throw new CustomException("Age cannot be negative");
}
System.out.println("age"+age);
}
catch(CustomException e)
{
  System.out.println(e.getMessage());
}
}

}
```

o/p

```
PS C:\Users\Admin\Desktop\sycs-41\prac4> javac pr3.java
PS C:\Users\Admin\Desktop\sycs-41\prac4> java pr3
Age cannot be negative
```

```
//if age positive if age=20;
```

```
PS C:\Users\Admin\Desktop\sycs-41\prac4> javac pr3.java
PS C:\Users\Admin\Desktop\sycs-41\prac4> java pr3
age20
```

###3B

PREDEFINE EXCEPTION

```
public class pr3b
{
    public static void main(String[] args)
    {
        try
        {
            int result=divide(10,0);
            System.out.println("result is :"+result);
        }
        catch(ArithmeticException e)
```

```
        {
            System.out.println("Error : Division by Zero");
        }
    }
    public static int divide(int a, int b)
    {
        return a/b;
    }
}
```

o/p

```
PS C:\Users\Admin\Desktop\sycs-41\prac4> javac pr3b.java
PS C:\Users\Admin\Desktop\sycs-41\prac4> java pr3b
Error : Division by Zero
```

##prac4a

---

write program to demonstrate the method of
a.list interface
b.set interface
c.map interface

A.list interface

#code

```
import java.util.*;

class ListDemo {
    public static void main(String[] args) {
        List<String> fruits = new ArrayList<>();
        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Cherry");
        fruits.add("Kiwi");
        fruits.add("Banana");
        fruits.add("Mango");
        System.out.println("List of fruits");
        for (String fruit : fruits) {
            System.out.println(fruit);
        }
        System.out.println("\n Element at index 2 :" + fruits.get(2));
        fruits.remove("Banana");
        System.out.println("\n list after removing Banana :" + fruits);
        System.out.println("\n list Contained Mango :" +
fruits.contains("Mango"));
        System.out.println("\n iterating using listiterator");
        ListIterator<String> iterator = fruits.listIterator();
        while (iterator.hasNext()) {
            System.out.println(iterator.next());
        }
    }
}
```

o/p

```
PS C:\Users\Admin\Desktop\sycs-41\prac4> javac ListDemo.java
PS C:\Users\Admin\Desktop\sycs-41\prac4> java ListDemo
List of fruits
Apple
Banana
```

```
Cherry
Kiwi
Banana
Mango

 Element at index 2 :Cherry

 list after removing Banana :[Apple, Cherry, Kiwi, Banana, Mango]

 list Contained Mango :true

 iterating using listiterator
Apple
Cherry
Kiwi
Banana
Mango
```

#B. SET INTERFACE

#code
