### 1: Write your own program and execute adding a method that takes parameters

class Box

{

double width;

double height;

double depth;

void volume()

{

System.out.print("Volume is ");

System.out.println(width \* height \* depth);

}

}

class Exercise3

{

public static void main (String args[])

{

Box mybox1 = new Box();

Box mybox2 = new Box();

mybox1.width = 20;

mybox1.height = 20;

mybox1.depth = 25;

mybox2.width = 3;

mybox2.height = 7;

mybox2.depth = 9;

mybox1.volume();

mybox2.volume();

} }

**OUTPUT:**

Volume is 10000.0

Volume is 189.0

### 2: Write your own program and execute for demonstrating static variables and methods.

class StaticDemo

{

static int a = 52;

static int b = 89;

static void callme()

{

System.out.println("a = " + a);

}

}

class StaticByName

{

public static void main(String args[])

{

StaticDemo.callme();

System.out.println("b = " + StaticDemo.b);

}

}

**OUTPUT**:

a=52

b= 89

### 3:Write your own program and excecute for demonstrating Default Constructor (No-argument constructor)

class Box

{

double width;

double height;

double depth;

Box()

{

System.out.println("Constructing Box");

width = 10;

height = 10;

depth = 10;

}

double volume()

{

return width \* height \* depth;

}

}

**OUTPUT:**

Constructing Box

1000

### 4: Write your own program and execute for demonstrating parameterizes constructor.

### class Box

### {

double width; double height; double depth;

Box(double w, double h, double d)

{

width = w;

height = h;

depth = d;

}

double volume()

{

return width \* height \* depth;

}

}

class BoxDemo1

{

public static void main(String args[])

{

Box mybox1 = new Box(10,20.30);

Box mybox2 = new Box(7,6,5);

double vol;

vol = mybox1.volume();

System.out.println("Volume is " + vol);

vol = mybox2.volume();

System.out.println("Volume is " + vol); }

}

**OUTPUT:**

Volume is 6000

Volume is 210.0

## 5.Write your own program and execute for demonstrating of finalize()method.

public **class** shashank

{

public static **void** main(String[] args)

{

shashank obj = **new** shashank();

obj = **null**;

System.gc();

System.out.println("Done");

}

@Override

protected **void** finalize()

{

System.out.println("finilize() method called ");

}

}

**OUTPUT:**

Done

finilize() method called

## 6: Write your own program and execute for Demonstrate Switch statement

**import** java.util.Scanner;

**public** **class** java{

**public** **static** **void** main(String args[])

{

Scanner in = **new** Scanner(System.***in***); System.***out***.print("Input sem: ");

**int** sem = in.nextInt();

**switch** (sem)

{

**case** (1):

System.***out***.println("You are 1st sem");

**break**;

**case** (2):

System.***out***.println("You are 2nd sem");

**break**;

**case** (3):

System.***out***.println("You are 3rd sem");

**break**;

**case** (4):

System.***out***.println("You are 4th sem");

**break**;

**case** (5):

System.***out***.println("You are 5th sem");

**break**;

**case** (6):

System.***out***.println("You are 6th sem");

**break**; **default**:

System.***out***.println("Please give the valid sem.");

**break**;

}

}

**OUTPUT:**

Input sem:5

You are in 5th sem

## 7: Write your own program and execute Demonstrate for statement

**public** **class** shashank

{

**public** **static** **void** main(String args[])

{

**int** a[][]={{1,4,4},{2,4,3},{8,4,5}};

**int** b[][]={{1,3,4},{2,4,8 },{1,2,4}};

**int** c[][]=**new** **int**[3][3];

**for**(**int** i=0;i<3;i++)

{

**for**(**int** j=0;j<3;j++){

c[i][j]=a[i][j]+b[i][j];

System.***out***.print(c[i][j]+"\t");

}

System.***out***.println();

}

}

}

**OUTPUT:**

2 7 8

4 8 11

9 6 9

## 8:Write your own program and execute for Demonstrate encapsulation concept

**class** Encapsulate{

**private** String name;

**private** **int** roll;

**private** **int** age;

**public** **int** getAge() { **return** age;}

**public** String getName() { **return** name;}

**public** **int** getRoll() { **return** roll; }

**public** **void** setAge(**int** newAge) { age = newAge; }

**public** **void** setName(String newName){ name = newName; }

**public** **void** setRoll(**int** newRoll){ roll = newRoll; }

}

### public class EncapDemojava{

**public** **static** **void** main (String[] args)

{

Encapsulate obj = **new** Encapsulate();

obj.setName("Shashank");

obj.setAge(18);

obj.setRoll(44);

System.***out***.println("name: " + obj.getName());

System.***out***.println("age: " + obj.getAge());

System.***out***.println("roll: " + obj.getRoll());

}

}

**OUTPUT:**

name: Shashank

age: 18

roll: 44

**9:** **Implement the concept of importing classes from user defined package and creating packages**

*/\* CREATE A NEW FOLDER WITH NAME pkg1 INSIDE THE pkg1 FOLDER SAVE A.java\*/*

package pkg1;

public class A

{

public void displayA()

{

System.out.println("class A");

}

}

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

*/\* CREATE A NEW FOLDER WITH NAME pkg2 INSIDE THE pkg2 FOLDER SAVE B.java\*/* package pkg2;

public class B

{

protected int m=20;

public void displayB()

{

System.out.println("class B");

System.out.println("m="+m);

}

}

-------------------------------------------------------------------------------- import pkg1.A;

import pkg2.\*;

class shashank

{

public static void main(String args[])

{

A a= new A();

B b= new B();

a.displayA();

b.displayB();

}

}

**OUTPUT:**

class A

class B

m=20

## 10: write the program so that the class should have only one responsibility and should satisfy SRP [Single Responsibility Principle]

**class** Calculator1 {

**public** **static** **int** add(**int** x, **int** y) { **return** x + y; }

**public** **static** **int** sub(**int** x, **int** y) { **return** x - y; }

**public** **static** **int** mul(**int** x, **int** y) { **return** x \* y; }

**public** **static** **int** div(**int** x, **int** y) { **return** x / y; }

}

**class** ResultPrinter {

**public** **static** **void** printResult(**int** value)

{

System.***out***.println("The value is="+value);

}

}

**public** **class** CalcDemoshashank

{

**public** **static** **void** main(String args[])

{

**int** a = Calculator.*add*(20, 50);

ResultPrinter.*printResult*(a);

**int** b = Calculator.*sub*(20, 40);

ResultPrinter.*printResult*(b);

**int** c = Calculator.*mul*(20, 30);

ResultPrinter.*printResult*(c);

**int** d = Calculator.*div*(20, 30);

ResultPrinter.*printResult*(d);

}

}

**OUTPUT:**

The value is=70

The value is=20

The value is=600

The value is=0

## 11: Write your own program and executeIllustrate creating an array of objects

**class** Student

{

**public** **int** roll\_no;

**public** String name;

Student(**int** roll\_no, String name)

{

**this**.roll\_no = roll\_no;

**this**.name = name;

}

}

**public** **class** Shashank

{

**public** **static** **void** main (String[] args)

{

Student[] arr;

arr = **new** Student[5];

arr[0] = **new** Student(1,"Shashidhar");

arr[1] = **new** Student(2,"Nandish");

arr[2] = **new** Student(3,"Nithin");

arr[3] = **new** Student(4,"Chethan ");

arr[4] = **new** Student(5,"Risheek");

**for** (**int** i = 0; i < arr.length; i++)

System.***out***.println("Element at " + i + " : " +

arr[i].roll\_no +" "+ arr[i].name);

}

}

**OUTPUT:**

Eelement at 0: 1 Shashidhar

Eelement at 1: 2 Nandish

Eelement at 2: 3 Nithin

Eelement at 3: 4 Chethan

Eelement at 4: 5 Risheek

## 12:Write your own program and execute for Demonstration of String Class and its Methods

Class java

**public** **static** **void** main(String args[])

{

System.***out***.println("-------------STRING DEMO-------");

String s1=**new** String("Cssjp");

String s2="sjp ";

String s3="";

System.***out***.println("the string s1="+s1);

System.***out***.println("the string s2="+s2);

System.***out***.println("Char at ="+s2.charAt(2));

System.***out***.println("the length of the string s1="+s1.length());

System.***out***.println("SubString Begin index ="+s1.substring(1));

System.***out***.println("SubString Begin & end index ="+s1.substring(1,4));

System.***out***.println("s1 equals s2 is="+s1.equals(s2));

System.***out***.println("check isempty="+s3.isEmpty());

System.***out***.println("s1 concatination s2 is="+s1.concat(s2));

System.out.println("the length of the string s1="+s1.length());  
 System.***out***.println("Replace="+s1.replace("S", "A"));

System.***out***.println("Lower case is="+s1.toLowerCase());

System.***out***.println("Upper case is="+s1.toUpperCase());

System.***out***.println("s1 equals ignore case s2="+s1.equalsIgnoreCase(s2));

}

**OUTPUT:**

the string s1=Cssjp

the string s2=sjp

Char at=p

the length of the string s1=5

SubString Begin index=ssjp

SubString Begin & index =ss

s1 equals s2 is=false

check isempty=true

s1 concatination s2 is=Cssjpssjp

the length of the string s1=5

Replace=Cssjp

Lower case is=cssjp

Upper case is=CSSJP

s1 equals ignore case s2=false

## 13:Write your own program and execute for Demonstration of Multilevel Inheritance

class GrandParent

{

public void gMethod() { System.out.println("GrandParent method"); }

}

class Parent`extends GrandParent

{

public void pMethod() { gMethod();

System.out.println("Parent method"); }

}

class Child extends Parent

{

public void cMethod() { pMethod();

System.out.println("Child method"); }

}

class MultilevelDemo

{

public static void main(String[] args)

{

Child cobj = new Child();

cobj.cMethod();

cobj.pMethod();

cobj.gMethod();

}

}

**OUTPUT:**

GrandParent method

Parent method

Child method

GrandParent method

Parent method

Child method

## Exercise 14: Demonstration an Open Closed Principle [OCP]

**interface** Shape

{

**public** **double** calculateArea();

}

**class** Rect **implements** Shape

{

**double** length;

**double** width;

**public** **double** calculateArea(){

**return** length \* width;

}

}

**class** Cir **implements** Shape

{

**public** **double** radius;

**public** **double** calculateArea(){

**return** (3.147\*radius\*radius);

}

}

**class** Square **implements** Shape

{

**double** l;

**public** **double** calculateArea() {

**return** (l\*l);

}

}

**class** AreaCal

{

**public** **double** calculateShapeArea(Shape shape){

**return** shape.calculateArea();

}

}

**public** **class** OCPDemoshashank {

**public** **static** **void** main(String args[])

{

AreaCal a1 = **new** AreaCal();

Rect r = **new** Rect();

r.length=30;

r.width=20;

Cir c= **new** Cir();

c.radius=10;

Square s = **new** Square();

s.l=30;

**double** r1= a1.calculateShapeArea(r);

**double** c1= a1.calculateShapeArea(c);

**double** s1=a1.calculateShapeArea(s);

System.***out***.println("----------OUTPUT--------");

System.***out***.println("Area of Rectangle="+r1);

System.***out***.println("Area of Circle="+c1);

System.***out***.println("Area of Square="+s1);

}

}

**OUTPUT:**

----------OUTPUT--------

Area of Rectangle=600.0

Area of circle=314.7

Area of square=900.0

**15: Write your own program and execute forDemonstration of method OverLoading.**

class OverloadDemo

{

void method()

{

System.out.println("No parameters");

}

void method(int a)

{

System.out.println("a: " + a);

}

void method(int a, int b)

{

System.out.println("a and b: " + a + " " + b);

}

double method(double a)

{

System.out.println("double a: " + a); return a\*a;

}

}

class OverloadExample

{

public static void main(String args[]) {

OverloadDemo ob = new OverloadDemo();

double result;

ob.method();

ob.method(20);

ob.method(10, 30);

result = ob.method(123.25);

System.out.println("Result of ob.method(123.25): " + result);

}

}

**OUTPUT:**

No parameters

a:20

a and b: 10 30

double a:123.25

result of ob.method(123.25): 15190.5625

**16:Write your own program and execute for Demonstration of method overriding.**

class A

{

int i, j;

A(int a, int b) { i = a; j = b; }

void show() { System.out.println("i and j: " + i + " " + j); }

} class B extends A

{ int k;

B(int a, int b, int c)

{ super(a, b); k = c;

}

void show()

{

System.out.println("k: " + k);

}

}

class Override

{ public static void main(String args[])

{

B subOb = new B(2, 6, 17);

subOb.show();

}

}

**OUTPUT: k=17**

## 

## 17: Write your own program and execute for Demonstration of Abstract class

abstract class A

{

abstract void callme();

}

class B extends A

{

void callme()

{

System.out.println("this is call me inside child.");

}

public static void main(String[] args)

{

B b = new B();

b.callme();

}

}

**OUTPUT:**

**this is me inside child.**

## 18:Write your own program and execute for Demonstration of Interface concept in Java

## interface Area

{

final static float pi=3.14f;

double compute(double x, double y);

}

interface Display extends Area

{

void display\_result(double result);

}

class Rectangle implements Display

{

public double compute(double x, double y)

{

return(pi\*x\*y);

}

public void display\_result(double result)

{

System.out.println("The area is:"+result);

}

}

class InterfaceDemoshashank

{

public static void main(String args[])

{

Rectangle r1=new Rectangle();

double res=r1.compute(20.4,25.4);

r1.display\_result(res);

}

}

**OUTPUT:**

**The area is :1627.0224543571471**

**19:Wite your own program and execute forDemonstration of file writer in java**

Import java.io.\*;

Class FileWriter {

Public static void main(String args[]) throws IOException {

FileWriter fw=new FileWriter(“Shashank.txt”);

String s=”Sri Jaya chamarajendra polytechnic Bangalore 560001”;

Char ch[]=s.toCharArray();

for (int i = 0; i < ch.length; i++)

fw.write(ch[i]);

fw.close();

}

}

**OUTPUT:**

## sri jayachamarajendra polytechnic banglore 560001.

## 

**20:Write your own program and execute for Demonstration of file reader class in java**

Import java.io.\*;

Class MyFileReader {

Public static void main(String args[]) throws IOException {

Int i = 0;

FileReader fr=new FileReader(“Shashank.txt”);

While ((i=fr.read()) != -1)

System.out.println((char) i);

fr.close();

}

}

**OUTPUT:**

## Arithematic exception occurs rest of the code

**21:Write your own program and execute for demonstrating of Multiple Catch Block in Exception Handling.**

**public** **class** MultipleCatchBlock1 {

**public** **static** **void** main(String[] args){

**try**{

**int** a[]=**new** **int**[5];

a[5]=30/0;

}

**catch**(ArithmeticException e)

{

System.out.println("Arithmetic Exception occurs");

}

**catch**(ArrayIndexOutOfBoundsException e)

{

System.out.println("ArrayIndexOutOfBounds Exception occurs");

}

**catch**(Exception e)

{

System.out.println("Parent Exception occurs");

}

System.out.println("rest of the code");

}

}

**OUTPUT:**

## Arithmetic Exception occurs

## Rest of the code

**22:Write your own program and execute for Demonstration of Throwing Our Own Exceptions.**

**public** **class** TestThrow1{

**static** **void** validate(**int** age)

{

**if**(age<20)

**throw** **new** ArithmeticException("not valid");

**else**

System.out.println("welcome to vote");

}

**public** **static** **void** main(String args[])

{

validate(15);

System.out.println("rest of the code...");

}

}

## OUTPUT: Exception in thread “main” java.lang.ArithmeticException:not valid

## at TestThrow1.validate(TestThrowl.java:5)

## at TestThrow1.maqin(TestThrowl.java:11)

## 23: Design an interface & implement it like one that builds different types of toys and check compliance with ISP [ Interface Segregation principle]

**interface** Toy

{ **void** price(**double** p); **void** color(String c);

}

**interface** Move

{ **void** move();

}

**interface** flying

{

**void** fly();

}

**class** car **implements** Toy,Move

{

**double** p; String c;

**public** **void** price(**double** p)

{

**this**.p=p;

System.***out***.println("The car Toy"); System.***out***.println("price =" +p);

}

**public** **void** color(String c)

{

**this**.c=c;

System.***out***.println("color =" +c);

}

**public** **void** move()

{

System.***out***.println("start to move forward and backward ");

}

}

**class** plane **implements** Toy, flying

{

**double** p; String c;

**public** **void** price(**double** p)

{

**this**.p=p;

System.***out***.println("The Plane Toy");

System.***out***.println("price =" +p);

}

**public** **void** color(String c)

**{**

**this .c=c;**

**System.out.println(“color=” +c);**

**}**

**Public void move()**

**{**

**System.out.println(“plane can move forward and backword”);**

**}**

**Public void fly()**

**{**

**System.out.println(“starts flying”);**

**}**

**}**

**Public class Toyb1**

**{**

**Public static void main(Strings args[])**

**{**

**Car c=new car();**

**c.price(24.0);**

**c.color(“green”);**

**c.move();**

**plane p=new plane();**

**p.price(50.0);**

**plane.color(“blue”);**

**p.move();**

**p.fly();**

**}**

**}**

## OUTPUT:

## The car toy

## Price =24.0

## Color=green

## Start to move forward and back ward

## The plane toy

## Price=50.0

## Color=blue

## Plane can move forward and backward

## Starts flying

## 24:Write your own program and execute for demonstrating enum in java,

## class ShshankWorkingdays

## {

**Enum Day  
 {**

**Sunday,Monday,Tuesday,Wednesday,Thursday,Friday,Saturday**

**}**

**Public static void main(Strings args[])**

## {

**For (Day d : Day.values())**

**{**

Weekend(d);

## {

}

Static void weekend (Day d)

## {

If (d.equals(Day.Sunday))

System.out.println(“value=”+ d + “is a Holiday”);

else

System.out.println(“value=”+ d + “is a working day”);

## }}

## OUTPUT:

## Value:Sunday is a holiday

## Value:Monday is a working day

## Value:Tuesday is a working day

## Value:Wednesday is a working day

## Value:Thursday is a working day

## Value:Friday is a working day

## Value :Saturday is a working day.

## 

**this**.c=c;

System.***out***.println("color =" +c);

}

**public** **void** move()

{

System.***out***.println("plane can move forward and backward ");

}

**public** **void** fly()

{

System.***out***.println("starts flying");

}

}

**public** **class** Toyb1

{

**public** **static** **void** main(String args[])

{

car c=**new** car();

c.price(12.0);

c.color("red");

c.move();

plane p=**new** plane();

p.price(15.0);

p.color("yellow");

p.move();

p.fly();

}

}

## 24:Write your own program and execute for Demonstrating Enum in Java

**class** Workingdays

{

**enum** Day

{

***Sunday***, ***Monday***, ***Tuesday***, ***Wednesday***, ***Thursday***, ***Friday***, ***Saturday***

} **public** **static** **void** main(String args[])

{ **for** (Day d : Day.*values*())

{

*weekend*(d);

}

}

**static** **void** weekend (Day d)

{

**if** (d.equals (Day.***Sunday***))

System.***out***.println("value = "+ d + "is a Holiday"); **else**

System.***out***.println("value ="+ d + " is a working day"); }}