

OOP-LAB-Assignment-4

Name :- Sagar Gupta
Roll No. :- 21MCF1R47
Reg. No. :- MC21107
Course :- MCA 1st year 2nd sem

Submitting Date :- 25-03-2022

Q-1. Create a class 'BOX' having parameters weight, height, length. Create 3 objects of varying dimension and find its volume i.e weight*height*length.

Ans-1

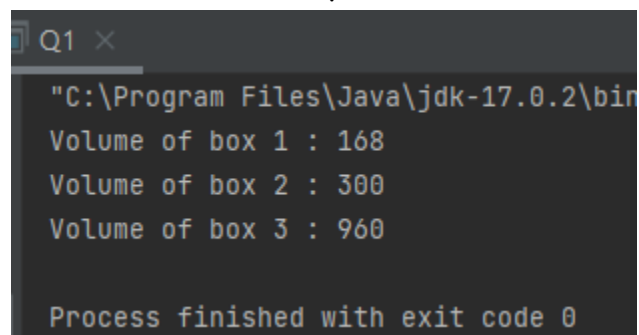
CODE

```
class Box{
    int length,width,height;
    Box(int l, int w, int h) {
        this.length = l;
        this.width = w;
        this.height = h;
    }
    int volume() {
        return length*width*height;
    }
}
public class Q1 {
    public static void main(String args[]) {
        Box b1 = new Box(4,6,7);
        System.out.println("Volume of box 1 : " + b1.volume());

        Box b2 = new Box(5,6,10);
        System.out.println("Volume of box 2 : " + b2.volume());

        Box b3 = new Box(8,10,12);
        System.out.println("Volume of box 3 : " + b3.volume());
    }
}
```

Output



```
Q1 x
"C:\Program Files\Java\jdk-17.0.2\bin
Volume of box 1 : 168
Volume of box 2 : 300
Volume of box 3 : 960
Process finished with exit code 0
```

Q-2 Using constructor overloading, find the volume for a cubical box and cuboidal box.

Ans-2

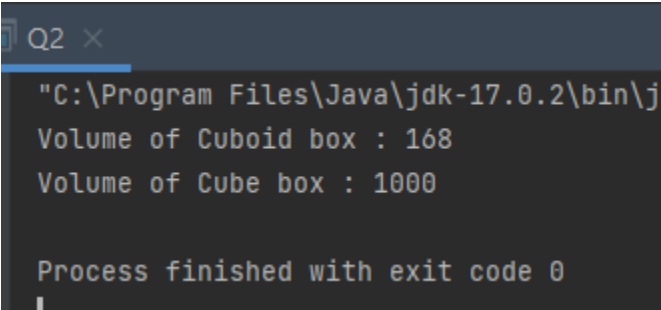
CODE

```
class CBox{
    int length,width,height;
    CBox(int l, int w, int h) {
        this.length = l;
        this.width = w;
        this.height = h;
    }
    CBox(int l){
        this.length=l;
        this.width=l;
        this.height=l;
    }
    int volume() {
        return length*width*height;
    }
}

public class Q2 {
    public static void main(String[] args) {
        CBox b1 = new CBox(4,6,7);
        System.out.println("Volume of Cuboid box : " + b1.volume());

        CBox b2 = new CBox(10);
        System.out.println("Volume of Cube box : " + b2.volume());
    }
}
```

Output



```
Q2 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe"
Volume of Cuboid box : 168
Volume of Cube box : 1000

Process finished with exit code 0
```

Q-3 Given a number, find its cube, square and square root by creating separate methods for each task. (square root can be a double type).

Ans-3

CODE

```
import java.util.Scanner;

public class Q3 {
```

```

public static int cube(int n) {
    return n*n*n;
}
public static int square(int n) {
    return n*n;
}
public static double sqrt(int n) {
    return Math.sqrt(n);
}
public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    System.out.print("Enter the number : ");
    int n = in.nextInt();
    System.out.println("Cube of Given number is : " + cube(n));
    System.out.println("Square of Given number is : " + square(n));
    System.out.println("Square root of Given number is : " + sqrt(n));
}
}

```

Output

```

Q3 x
"C:\Program Files\Java\jdk-17.0.2\bin\j
Enter the number : 4
Cube of Given number is : 64
Square of Given number is : 16
Square root of Given number is : 2.0
Process finished with exit code 0

```

Q-4 Print the sum, difference and product of two complex numbers by creating a class named 'Complex' with separate methods for each operation whose real and imaginary parts are entered by the user.

Ans-4

CODE

```

import java.util.Scanner;

class Complex {
    int real1,imaginary1,real2,imaginary2;
    Complex(int r1, int i1 , int r2 ,int i2 ) {
        this.real1 = r1;
        this.imaginary1 = i1;
        this.real2 = r2;
        this.imaginary2 = i2;
    }

    void Sum() {
        int r3 = real1 + real2;
        int i3 = imaginary1 + imaginary2;
        if(i3>=0) {
            System.out.println("Sum is : " + r3 + "+" + i3 + "i");
        }
    }
}

```

```

    }
    else{
        System.out.println("Sum is : " + r3 + "" + i3 + "i");
    }
}

void Difference() {
    int r3 = real1 - real2;
    int i3 = imaginary1 - imaginary2;
    if(i3>=0) {
        System.out.println("Difference is : " + r3 + "+" + i3 + "i");
    }
    else{
        System.out.println("Difference is : " + r3 + "" + i3 + "i");
    }
}

void Product() {
    int r3 = real1*real2 - imaginary1*imaginary2;
    int i3 = real1*imaginary2 + imaginary1*real2;
    if(i3>=0) {
        System.out.println("Product is : " + r3 + "+" + i3 + "i");
    }
    else{
        System.out.println("Product is : " + r3 + "" + i3 + "i");
    }
}
}

public class Q4 {
    public static void main(String args[]) {
        Scanner in = new Scanner(System.in);

        System.out.print("Enter real part of 1st Number : ");
        int r1 = in.nextInt();
        System.out.print("Enter real part of 1st Number : ");
        int i1 = in.nextInt();
        System.out.print("Enter real part of 2nd Number : ");
        int r2 = in.nextInt();
        System.out.print("Enter real part of 2nd Number : ");
        int i2 = in.nextInt();

        Complex c = new Complex(r1,i1,r2,i2);
        c.Sum();
        c.Difference();
        c.Product();
    }
}

```

Output

```
Q4 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.
Enter real part of 1st Number : 10
Enter real part of 1st Number : 12
Enter real part of 2nd Number : 8
Enter real part of 2nd Number : 15
Sum is : 18+27i
Difference is : 2-3i
Product is : -100+246i

Process finished with exit code 0
```

Q-5 Define a private inner class and access the class methods through a method

Ans-5

CODE

```
class OuterClass {
    private class InnerClass {
        public void print() {
            System.out.println("Inner Class");
        }
    }
    void DisplayInner() {
        System.out.println("Outer Class Method");
        InnerClass i = new InnerClass();
        i.print();
    }
}
public class Q5 {
    public static void main(String args[]) {
        OuterClass o = new OuterClass();
        o.DisplayInner();
    }
}
```

Output

```
Q5 x
"C:\Program Files\Java\jdk-17.0.2\bin\
Outer Class Method
Inner Class

Process finished with exit code 0
```

Q-6 Define an anonymous inner class in a program.

Ans-6

CODE

```

abstract class class6{
    abstract void show();
}
public class Q6 {
    public static void main(String[] args) {
        class6 c = new class6() {
            @Override
            void show() {
                System.out.println("Show Method Called");
            }
        };
        c.show();
    }
}

```

Output

```

Q6 x
"C:\Program Files\Java\jdk-17.0.2\bin\
Show Method Called
Process finished with exit code 0

```

Q-7 Show an example of an anonymous inner class passed as an argument.

Ans-7

CODE

```

abstract class class7{
    int a,b;
    class7(int a,int b){
        this.a=a;
        this.b=b;
    }
    abstract int SUM();
    abstract void b7(class7 c);
}

public class Q7 {
    public static void main(String[] args) {
        class7 c = new class7(5,13) {
            @Override
            int SUM() {
                return a+b;
            }
            @Override
            void b7(class7 c) {
                if(c.a==a){
                    System.out.println("Correct");
                }
                else{
                    System.out.println("Incorrect");
                }
            }
        };
    }
}

```

```

    }
}
};
System.out.println(c.SUM());
c.b7(c);
}
}

```

Output

```

Q7 x
"C:\Program Files\Java\jdk-17.0.2\bin
18
Correct
Process finished with exit code 0

```

Q-8 Create a class called vehicle which has the properties as `companyName` and `color`. Create a constructor which initializes these properties. Put a method to get the company name of the vehicle that returns a string which is the `companyName`. Put another method called `introduce()` which introduces the vehicle by displaying its `companyName` and `color`. Create a subclass of this class called `Car` and add properties such as `numberOfPassengers` and `modelName`. The `details()` method of this class must print all the attribute values such as `companyName`, `color`, `numberOfPassengers` and `modelName`.

Ans-8

CODE

```

class Vehicle {
    String companyName,color;
    Vehicle(String n,String c) {
        this.companyName=n;
        this.Color=c;
    }
    public String getCompanyName() {
        return companyName;
    }
    public void introduce() {
        System.out.println("Company Name : "+companyName);
        System.out.println("Color : "+Color+"\n");
    }
}
class Car extends Vehicle
{
    int NP;
    String modelName;
    Car(int n,String m,String cn,String c) {
        super(cn,c);
    }
}

```

```

        this.NP =n;
        this.ModelName=m;
    }
    public void Details() {
        System.out.println("company Name : "+CompanyName);
        System.out.println("Color : "+Color);
        System.out.println("Number of Passengers : "+NP);
        System.out.println("Model Name : "+ModelName+"\n");
    }
}
public class Q8{
    public static void main(String[] args) {
        Car c = new Car(4,"Super X","BMW","Red");
        System.out.println("Company Name(getCompanyname()) :
"+c.getCompanyName()+"\n");
        c.introduce();
        c.Details();
    }
}

```

Output

```

Q8 x
"C:\Program Files\Java\jdk-17.0.2\bin\java
Company Name(getCompanyname()) : BMW

Company Name : BMW
Color : Red

company Name : BMW
Color : Red
Number of Passengers : 4
Model Name : Super X

Process finished with exit code 0

```

Q-9 Implement the above question Q1 with using the super keyword and instead of the details() method in the Car, introduce everything inside the introduce() method using the super keyword and display each attribute's value.

Ans-9

CODE

```

class VEHICLE {
    String CompanyName,Color;
    VEHICLE(String n,String c) {
        this.CompanyName=n;

```



```

        this.Color=c;
    }
    public String getCompanyName() {
        return CompanyName;
    }
    public void introduce() {
        System.out.println("Company Name : "+CompanyName);
        System.out.println("Color : "+Color);
    }
}
class CAR extends VEHICLE
{
    int NP;
    String ModelName;
    CAR(int n,String m,String cn,String c) {
        super(cn,c);
        this.NP =n;
        this.ModelName=m;
    }
    public void introduce() {
        super.introduce();
        System.out.println("Number of Passengers : "+NP);
        System.out.println("Model Name : "+ModelName+"\n");
    }
}
public class Q9{
    public static void main(String[] args) {
        CAR c = new CAR(4,"Super X","BMW","Red");
        c.introduce();
    }
}

```

Output

```

Q9 x
"C:\Program Files\Java\jdk-17.0.2\bin
Company Name : BMW
Color : Red
Number of Passengers : 4
Model Name : Super X

Process finished with exit code 0

```

Q-10 Implement multilevel inheritance and show the instantiation of the objects at each level.

Ans-10

CODE

```

class class10a{
    class10a(){
        System.out.println("Class A Called!");
    }
}

```

```

    }
}
class class10b extends class10a{
    class10b(){
        System.out.println("Class B Called!");
    }
}
class class10c extends class10b{
    class10c(){
        System.out.println("Class C Called!");
    }
}

public class Q10 {
    public static void main(String[] args) {
        class10c c = new class10c();
    }
}

```

Output

```

Q10 x
"C:\Program Files\Java\jdk-17.0.2\b
Class A Called!
Class B Called!
Class C Called!

Process finished with exit code 0

```

Q-11 Create two different base classes and extend both of them by a subclass simultaneously and observe the results. Explain the reason for such an outcome.

Ans-11

CODE

```

class class11a{
    class11a(){
        System.out.println("Class A");
    }
}
class class11b{
    class11b(){
        System.out.println("Class B");
    }
}
class class11c extends class11a,class11b{
    class11c(){
        System.out.println("Class C");
    }
}

```

```

public class Q11 {
    public static void main(String[] args) {
        class11c c = new class11c();
    }
}

```

Output
ERROR

Reason

Java does not support Multiple Inheritance with classes.

Q-12 Create a class for Person and initialize the name of the person in its constructor. Extend this class for employees and use the parent class's constructor to initialize the name of the employee and assign id and salary to each employee in the constructor of the employee class which uses the constructor of the person class.

Ans-12

CODE

```

class Person{
    String name;
    Person(String n){
        this.name = n;
    }
}
class Emp extends Person {
    int id;
    double salary;
    Emp(int id, double salary, String name){
        super(name);
        this.id=id;
        this.salary=salary;
    }
    public void show(){
        System.out.println("Employee ID : "+id);
        System.out.println("Employee Salary : "+salary);
        System.out.println("Employee Name : "+name);
    }
}
public class Q12 {
    public static void main(String[] args) {
        Emp e1 = new Emp(1,10000.0,"Paras");
        e1.show();

        Emp e2 = new Emp(2,999999.0,"Sagar");
        e2.show();

        Emp e3 = new Emp(3,12000.0,"Someone");
        e3.show();
    }
}

```

}

Output

```
Q12 x
"C:\Program Files\Java\jdk-17.0.2\bin\jav
Employee ID : 1
Employee Salary : 10000.0
Employee Name : Paras
Employee ID : 2
Employee Salary : 999999.0
Employee Name : Sagar
Employee ID : 3
Employee Salary : 12000.0
Employee Name : Someone

Process finished with exit code 0
```

Q-13 Use parent class method using super keyword in the extended class and show the results of it by taking an appropriate example.

Ans-13

CODE

```
class class13a{
    int x,y;
    class13a(int a, int b){
        this.x=a;
        this.y=b;
    }
    int productsum(){
        return x*y;
    }
}
class class13b extends class13a{
    int z;
    class13b(int a, int b, int c){
        super(a,b);
        this.z = c;
    }
    int productsum(){
        return super.productsum()+z;
    }
}
public class Q13 {
    public static void main(String[] args) {
        class13b c = new class13b(10,5,50);
        System.out.println("a*b+c = "+c.productsum());
    }
}
```

Output

```
Q13 x
"C:\Program Files\Java\jdk-17.0.2\bin
a*b+c = 100

Process finished with exit code 0
```

Q-14 Implement static polymorphism/ compile time polymorphism through method overloading. Implement a class for shapes. In that, put a method to calculate the area of circle, triangle, square, sphere, hemisphere and cone each with the same name area but parameters as required.

Ans-14

CODE

```
class shapes {
    void area(Double radius) {
        double area = 3.14*radius*radius;
        System.out.println ("Area of circle : " +area);
    }
    void area(int base,int height) {
        double area = 0.5*base*height;
        System.out.println ("Area of triangle : " +area);
    }
    void area(int side) {
        int area = side*side;
        System.out.println ("Area of square : " +area);
    }
    void area(Float r) {
        double area = 4*3.14*r*r;
        System.out.println ("Area of sphere : " +area);
    }
    void area(String radius) {
        int r=Integer.parseInt(radius);
        double area=3*r*r;
        System.out.println("Area of hemisphere : "+area);
    }
    void area(Double r,Double l) {
        Double area=3.14*r*l;
        System.out.println("Area of cone : "+area);
    }
}
public class Q14 {
    public static void main(String[] args) {
        shapes s=new shapes();
        s.area(5.35);
        s.area(10,12);
        s.area(10);
        s.area(7.0f);
        s.area("9");
        s.area(5.5d, 10.2d);
    }
}
```

}

Output

```
Q14 ×
"C:\Program Files\Java\jdk-17.0.2\bin
Area of circle : 89.87464999999999
Area of triangle : 60.0
Area of square : 100
Area of sphere : 615.44
Area of hemisphere : 243.0
Area of cone : 176.154

Process finished with exit code 0
```

Q-15 Implement run time polymorphism through method overriding by writing a class called Employee. This class should contain a method called showSalary() which displays the salary range of an employee. Put a generic statement in this method for employee class. Make classes such as Manager, Intern etc. Which extends the Employee class and overrides the salary method to show the specific salary for the respective category of the employee.

Ans-15

CODE

```
class Employee{
    int sal;
    public void salary(){
        sal=1200;
        showSalary(sal);
    }
    // Generic method
    public static<T> void showSalary(T sal){
        System.out.println(" Salary is : "+sal);
    }
}
class Manager extends Employee {
    int sal=1000;
    public void salary(){
        System.out.print("Manager");
        showSalary(sal);
    }
}
class Intern extends Employee {
    int sal=1500;
    public void salary(){
        System.out.print("Intern");
        showSalary(sal);
    }
}
```

```

public class Q15 {
    public static void main(String[] args) {
        Employee m = new Manager();//upcasting
        m.salary();
        Employee i = new Intern();//upcasting
        i.salary();
    }
}

```

Output

```

Q15 x
"C:\Program Files\Java\jdk-17.0.2\bin\j...
Manager Salary is : 1000
Intern Salary is : 1500

Process finished with exit code 0

```

Q-16 Create a class Book which has attributes such as title of the book, author, genre, number of pages and price of the book. Create another class called StoryBook which is a subclass of the previous class and show an increment of 30% in the rates of this book compared to the previous one using method overriding (polymorphism). Put appropriate methods in the respective classes for getting and setting the values of attributes. The use of super may be required here.

Ans-16

CODE

```

class Book{
    String title,genre;
    int pages;
    double price;
    public Book(String t, String g, int pg,double pr ){
        this.title=t;
        this.genre=g;
        this.pages=pg;
        this.price=pr;
    }

    public void getData(){
        System.out.println("Book Title : "+title);
        System.out.println("Book Genre : "+genre);
        System.out.println("No. of Pages in Book : "+pages);
        System.out.println("Price of Book : "+price+" Rs");
    }
}

class StoryBook extends Book{
    double newPrice;

```

```

StoryBook(String t, String g, int pg,double pr){
    super(t,g,pg,pr);
    newPrice = price+(price*0.3);
}
public void getData(){
    System.out.println("Book Title : "+title);
    System.out.println("Book Genre : "+genre);
    System.out.println("No. of Pages in Book : "+pages);
    System.out.println("Price of Book : "+newPrice+" Rs");
}
}

public class Q16 {
    public static void main(String[] args) {
        Book b = new StoryBook("Steve jobs","Biography",100,200);
        b.getData();
    }
}

```

Output

```

Q16 x
"C:\Program Files\Java\jdk-17.0.2\bin
Book Title : Steve jobs
Book Genre : Biography
No. of Pages in Book : 100
Price of Book : 260.0 Rs

Process finished with exit code 0

```

Q-17 . Create an abstract class Bank which has a method showing the rate of interest of different banks. Extend this class with different Banks and implement the method specific to each sub class.

Ans-17

CODE

```

abstract class Bank{
    public abstract int getRateOfInterest();
}
class SBI extends Bank{
    public int getRateOfInterest(){
        return 7;
    }
}
class PNB extends Bank{
    public int getRateOfInterest(){
        return 8;
    }
}

```



```

public class Q17 {
    public static void main(String[] args) {
        Bank s=new SBI();
        Bank p=new PNB();
        System.out.println("Rate of Interest in SBI is: "+s.getRateOfInterest()+"%");
        System.out.println("Rate of Interest in PNB is: "+p.getRateOfInterest()+"%");
    }
}

```

Output

```

Q17 x
"C:\Program Files\Java\jdk-17.0.2\bin\
Rate of Interest in SBI is: 7%
Rate of Interest in PNB is: 8%

Process finished with exit code 0

```

Q-18a Create an abstract class with constructor, data member and methods such that the class is abstract and another class extends the current class and creates an instance of it.

Q18b Create an abstract class and try to create an object of it. What result do you get? How can it be resolved? Put more than one abstract method in base class and see what happens if you don't override all the abstract methods in it and how to resolve that.

Ans-18a

CODE

```

abstract class class18a{
    int x,y,z;
    class18a(int a, int b, int c){
        this.x = a;
        this.y = b;
        this.z = c;
    }
    int sum(){
        int sum = x + y + z;
        return sum;
    }
}
class class18b extends class18a{
    class18b(int a, int b, int c) {
        super(a, b, c);
    }
}

```

```

public class Q18 {
    public static void main(String[] args) {
        class18a c = new class18b(5,6,7);
    }
}

```

```

    System.out.println("Sum is: "+c.sum());
}
}

```

Output

```

Q18 X
"C:\Program Files\Java\jdk-17.0.2\
Sum is: 18
Process finished with exit code 0

```

Ans-18b

//1st part : error, we can't create object for abstract class

//2nd part : it can be resolved if we create another class which extends this
 // abstract class and then move on to create an instance of the new class. This
 new class
 // also must override all the abstract methods in the abstract cls

//3rd part : error, all the abstract methods in the abstract cls must be overridden
 to avoid unnecessary extra work.

Q-19 Implement Q17 using interface concept and instantiate the subclass
 to show its working.

Ans-19

CODE

```

interface BANK{
    abstract int getRateOfInterest();
}
interface Sbi extends BANK{
    default int getRateOfInterest(){
        return 7;
    }
}
interface Pnb extends BANK{
    default int getRateOfInterest(){
        return 8;
    }
}
class c2 implements Sbi,Pnb{
    void show(){
        System.out.println("Sbi Interest Rate : "+Sbi.super.getRateOfInterest()+"%");
        System.out.println("Pnb Interest Rate : "+Pnb.super.getRateOfInterest()+"%");
    }
    @Override
    public int getRateOfInterest() {
        return Sbi.super.getRateOfInterest();
    }
}

```

```

public class Q19{
    public static void main(String[] args) {
        c2 c = new c2();
        c.show();
    }
}

```

Output

```

Q19 x
"C:\Program Files\Java\jdk-17.0.2\bin
Sbi Interest Rate : 7%
Pnb Interest Rate : 8%

Process finished with exit code 0

```

Q-20 Try to implement multiple inheritance and observe the results. Implement multiple inheritance using interfaces taking some examples and showing the results.

Ans-20

CODE

```

interface A{
    default void show(){
        System.out.println("Interface A");
    }
}
interface B{
    default void show(){
        System.out.println("Interface B");
    }
}
class C implements A,B{
    public void show(){
        A.super.show();
        B.super.show();
    }
}
public class Q20 {
    public static void main(String[] args) {
        C c = new C();
        c.show();
    }
}

```

Output

```
Q20 x
"C:\Program Files\Java\jdk-17.0.2\
Interface A
Interface B

Process finished with exit code 0
```

Q-21 . Implement interface inheritance and show the results by instantiating the class implementing the interface that inherits/extends the base interface.

Ans-21

CODE

```
interface i1{
    static void show(){
        System.out.println("Interface 1");
    }
}
interface i2 extends i1{
    static void show(){
        i1.show();
        System.out.println("Interface 2");
    }
}
class c1 implements i2 {
    public void show(){
        i2.show();
    }
}
public class Q21 {
    public static void main(String[] args) {
        c1 c = new c1();
        c.show();
    }
}
```

Output

```
Q21 x
"C:\Program Files\Java\jdk-17.0.2\bin\
Interface 1
Interface 2

Process finished with exit code 0
```

Q-22 Write a class that keeps a running total of all characters passed to it (one at a time) and throws an exception if passed a non-alphabetic character.

Ans-22

CODE

```
import java.io.*;

class NotACharException extends Exception{
    public NotACharException(String s) {
        super(s);
    }
    boolean check(char s) {
        boolean isAlpha = Character.isLetter(s);
        return isAlpha;
    }
}

class CHECKSTRING{
    static int cnt=0;
    CHECKSTRING() throws IOException,NotACharException{
        NotACharException c=new NotACharException("");
        BufferedReader b = new BufferedReader(new InputStreamReader(System.in));
        try {
            while(true) {
                char ch = b.readLine().charAt(0);
                System.out.println("\n");
                if (c.check(ch)) {
                    cnt++;
                }
                else {
                    throw new NotACharException("");
                }
            }
        }catch (NotACharException notAChar){
            System.out.println("The no of entered characters is: " + cnt);
            System.out.println(cnt+1+" element is alphanumeric!");
        }
    }
}

public class Q22 {
    public static void main(String[] args) throws NotACharException, IOException {
        System.out.println("Enter Character 1 at a time");
        CHECKSTRING c = new CHECKSTRING();
    }
}
```

Output

```
Q22 x
"C:\Program Files\Java\jdk-17.0.2\bin\ja
Enter Character 1 at a time
a
v
3

The no of entered characters is: 2
3 element is alphanumeric!

Process finished with exit code 0
```

Q-23 Create a user-defined exception named CheckArgument to check the number of arguments passed through the command line. If the number of arguments is less than 5, throw the CheckArgumentexception, else print the addition of all the five numbers.

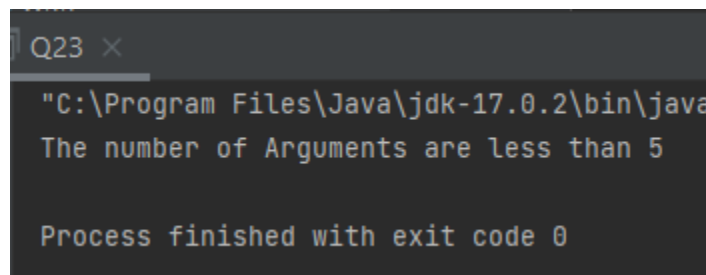
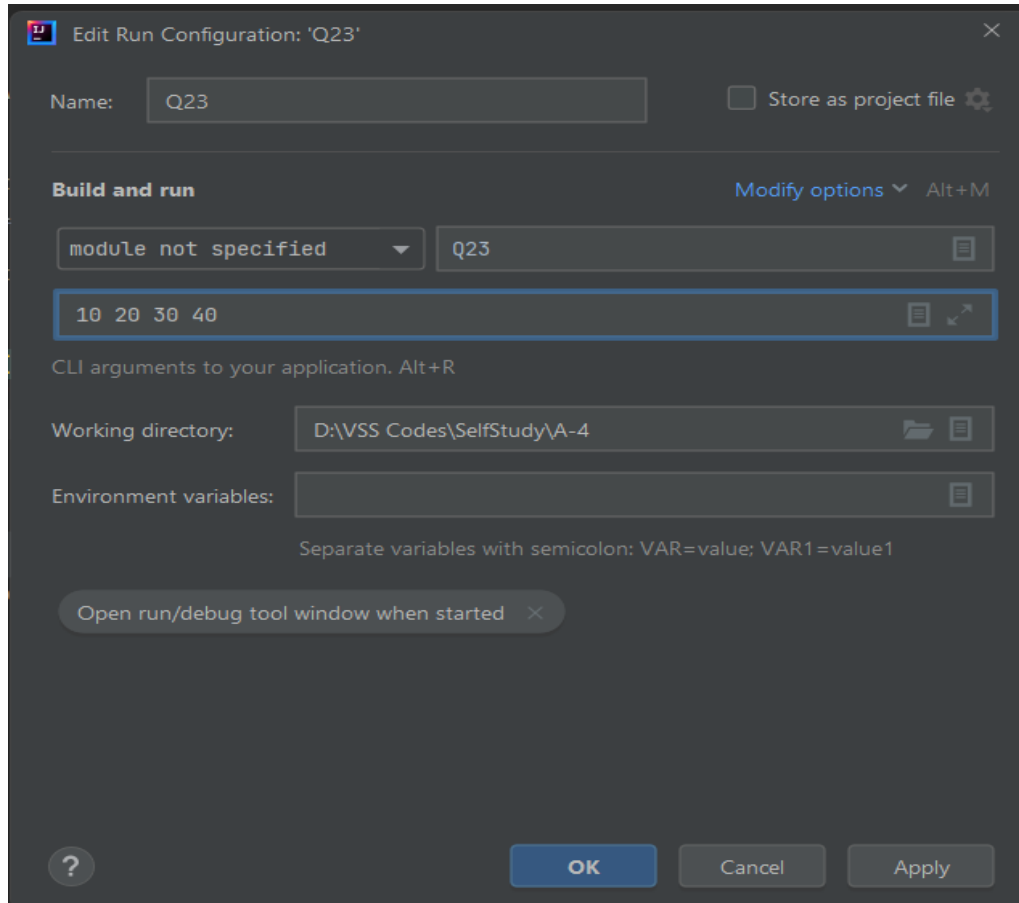
Ans-23

CODE

```
class CheckArgument extends Exception {
    public CheckArgument(String s) {
        super(s);
    }
}
public class Q23{
    public static void main(String[] args) {
        int arr[]=new int[5];
        int sum=0;
        try {
            if(args.length<5)
                throw new CheckArgument("");
            else {
                for (int i = 0; i < 5; i++) {
                    arr[i] = Integer.parseInt(args[i]);
                    sum += arr[i];
                }
                System.out.println("The sum of entered number is: "+sum);
            }
        }
        catch(CheckArgument e) {
            System.out.println("The number of Arguments are less than 5");
        }
    }
}
```

}

Output



Q-24 Write a program for example of try and catch block. In this check whether the given array size is negative or not.

Ans-24

CODE

```
import java.util.Scanner;
public class Q24 {
    public static void main(String[] args) throws Exception{
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter array size: ");
        int size=sc.nextInt();
        try {
            int arr[] = new int[size];
            System.out.println("Size Alloted to array");
        }
        catch(NegativeArraySizeException e) {
```

```

        System.out.println(e);
    }
}
}

```

Output

```

Q24 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe"
Enter array size: -5
java.lang.NegativeArraySizeException: -5

Process finished with exit code 0

```

Q-25 Write a program for example of multiple catch statements occurring in a program.

Ans-25

CODE

```

public class Q25 {
    public static void main(String[] args) {
        int[] a= {1,0};
        try{
            int n=a[0];
            int d=a[1];

            //1st dividing by 0
            int r=n/d;
            //2nd Array out of index
            int l = a[2];
            System.out.println("Result= "+r);
        }
        catch(ArithmeticException ae )
        {
            System.out.println(ae);
        }
        catch(ArrayIndexOutOfBoundsException aie)
        {
            System.out.println(aie);
        }
    }
}
}

```

Output


```
Q25 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe"
java.lang.ArithmeticException: / by zero

Process finished with exit code 0
```

```
Q25 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent:C:\Program Files\Je
java.lang.ArrayIndexOutOfBoundsException: Index 2 out of bounds for length 2

Process finished with exit code 0
```

Q-26 Write a program to illustrate sub class exception precedence over base class.

Ans-26

CODE

```
class Parent{
    int n = 100;
    int d=0;
    public void cal(){
        try {
            int r = n / d;
        }catch (ArithmeticException ae){
            System.out.println("Parent Class Exception Caught");
        }
    }
}
class Child extends Parent{
    //subclass Exception got precedence over Parent class
    public void cal(){
        try {
            int arr[] = {1, 2, 3};
            arr[5] = 100;
        }catch (ArrayIndexOutOfBoundsException a){
            System.out.println("Base Class Exception Caught");
        }
    }
}
public class Q26 {
    public static void main(String[] args) {
        Parent p=new Child();
        p.cal();
    }
}
```

Output

```
Q26 x
"C:\Program Files\Java\jdk-17.0.2\bin\
Base Class Exception Caught

Process finished with exit code 0
```

Q-27 Write a program to illustrate usage of try/catch with the finally clause.

Ans-27

CODE

```
import java.util.Scanner;
class class27{
    void show(int a, int b) {
        try {
            int c=a/b;
            System.out.println("a/b = ");
        } catch (Exception e) {
            System.out.println(e);
        } finally {
            System.out.println("Finally block executed!");
        }
    }
}
public class Q27 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter 2 Numbers : ");
        int a = sc.nextInt();
        int b = sc.nextInt();

        class27 c = new class27();
        c.show(a,b);
    }
}
```

Output

```
Q27 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe
Enter 2 Numbers : 5 0
java.lang.ArithmeticException: / by zero
Finally block executed!

Process finished with exit code 0
```

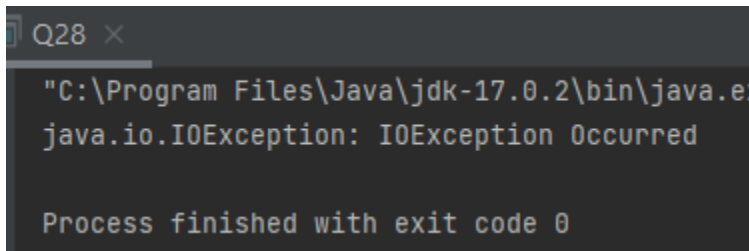
Q-28 Write a program to describe usage of throws clause.

Ans-28

CODE

```
import java.io.*;
class ThrowExample {
    void myMethod(int num)throws IOException, ClassNotFoundException{
        if(num==1) {
            throw new IOException("IOException Occurred");
        }
        else {
            throw new ClassNotFoundException("ClassNotFoundException");
        }
    }
}
public class Q28{
    public static void main(String args[]){
        try{
            ThrowExample t=new ThrowExample();
            t.myMethod(1);
        }catch(Exception e){
            System.out.println(e);
        }
    }
}
```

Output



```
Q28 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe"
java.io.IOException: IOException Occurred

Process finished with exit code 0
```

Q-29 Write a program for creation of user defined exceptions.

Ans-29

CODE

```
import java.util.Scanner;
class underAgeException extends Exception {
    underAgeException(String str){
        super(str);
    }
}
class check {
    static void validate(int age) throws underAgeException{
        if(age < 18) {
            throw new underAgeException("You are Under Age!");
        }
        else {
            System.out.println("Welcome to 18+ Club");
        }
    }
}
```

```

public class Q29 {
    public static void main(String[] args) {
        int age;
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter your age : ");
        age = sc.nextInt();
        try {
            check.validate(age);
        }
        catch(Exception e) {
            System.out.println("Caught an Exception: "+e);
        }
    }
}

```

Output

```

Q29 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent:0
Enter your age : 16
Caught an Exception: underAgeException: You are Under Age!
Process finished with exit code 0

```

Q-30 Write a java program to illustrate 4 different possible NullPointerException cases possible.

Ans-30

CODE

```

class class30{
    public int v=10;
    void show(){
        System.out.println("Class30 Class called");
    }
}
public class Q30 {
    public static void main(String[] args) {
        System.out.println("\n1st Case For Null Pointer is when we throws null");
        try{
            throw null;
        }catch (NullPointerException npe){
            System.out.println("Null Pointer Caught\n");
        }
        System.out.println("2nd Case when we invoke any method with null object");
        try{
            class30 c = null;
            c.show();
        }catch (NullPointerException npe){
            System.out.println("Null Pointer Caught\n");
        }
        System.out.println("3rd Case when we try to change null object class data");
        try{

```

```

class30 c = null;
c.v=100;
}catch (NullPointerException npe){
    System.out.println("Null Pointer Caught\n");
}
System.out.println("4th case if we give array size as Null");
try{
    int[] a = null;
    int length = a.length;
}catch (NullPointerException npe){
    System.out.println("Null Pointer Caught\n");
}
}
}
}

```

Output

```

Q30 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent
1st Case For Null Pointer is when we throws null
Null Pointer Caught

2nd Case when we invoke any method with null object
Null Pointer Caught

3rd Case when we try to change null object class data
Null Pointer Caught

4th case if we give array size as Null
Null Pointer Caught

Process finished with exit code 0

```

Q-31 You have been assigned to create a student database (Just information) of our institute. For a class named StudentInfo, create several objects for this class, each object representing one student (means array of objects). The StudentInfo class is different and the main class is different. The number of students should be taken from the user console. Student details should be like id, name, department name, college name. College name should be common to all of them (means this should be class variable). Now the student id of each student should be unique, if any of the student id is repeating then handle with an user-defined exception. Name and dept name should not be empty, if they are empty then handle with user-defined exception. After all, store the student's data inside a file using stream classes not by using file classes. And the number of students should be a minimum of 10.

Ans-31

CODE

```
import java.io.*;
import java.util.Scanner;
class StudentInfo implements Serializable
{
    int id;
    String name,dname,cname = "NITW";
}
class duplicateid extends Exception {
    public duplicateid(String s) {
        super(s);
    }
}
class empty extends Exception {
    public empty(String s) {
        super(s);
    }
}
public class Q31 {
    public static void main(String[] args) {
        String nm="",d="",c="";
        int n,x;
        Scanner inp= new Scanner(System.in);
        System.out.print("Enter number of student records to be entered : ");
        n=inp.nextInt();
        inp.nextLine();
        StudentInfo[] rec=new StudentInfo[n];
        for(int i=0;i<n;i++) {
            System.out.println(i+1);
            rec[i]=new StudentInfo();
            System.out.print("Enter student id : ");
            x=inp.nextInt();
            inp.nextLine();
            try {
                for(int j=0;j<i;j++) {
                    if(rec[j].id == x) {
                        throw new duplicateid("duplicate entry");
                    }
                }
            }
            catch(duplicateid de) {
                System.out.println("duplicate entries are not allowed");
                System.out.println("Enter unique student id");
                i--;
                continue;
            }
            rec[i].id=x;
            try {
                System.out.print("Enter Student's name :");
                nm=inp.nextLine();
```

```

        if(nm.trim().isEmpty()) {
            throw new empty("empty name");
        }
    } catch(empty ee) {
        while(nm.trim().isEmpty()) {
            System.out.println("Name cannot be empty. Enter again");
            nm=inp.nextLine();
        }
    }
    try {
        System.out.print("Enter Department name : ");
        d=inp.nextLine();
        if(d.trim().isEmpty()) {
            throw new empty("empty dept");
        }
    }
    catch(empty ee) {
        while(d.trim().isEmpty()) {
            System.out.println("Department name cannot be empty. Enter again");
            d=inp.nextLine();
        }
    }
    System.out.print("Enter College name : ");
    c=inp.nextLine();
    rec[i].name=nm;
    rec[i].dname=d;
    rec[i].cname=c;
    System.out.println("_____");
}
for(int i=0;i<n;i++) {
    System.out.println(rec[i].id + " " + rec[i].name + " " + rec[i].dname + " " +
rec[i].cname);
}
System.out.println("Wrting records into text file");
try {
    FileOutputStream fos = new FileOutputStream("StudentInfoRecords.txt");
    ObjectOutputStream oos = new ObjectOutputStream(fos);
    for(int i=0;i<n;i++) {

        oos.writeObject(rec[i]);
        oos.flush();
    }
    oos.close();
}
catch(IOException e) {
    System.out.println("Exception during writing into file: " + e);
    System.exit(0);
}
System.out.println("success");
}
}

```

Output

```
Q31 x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent:C:\Program Files
Enter number of student records to be entered : 10
1
Enter student id : 1
Enter Student's name :Name1
Enter Department name : D1
Enter College name : NITW
-----
2
Enter student id : 1
duplicate entries are not allowed
Enter unique student id
2
Enter student id : 2
Enter Student's name :Name2
Enter Department name : D1
Enter College name : NITW
-----
3
Enter student id : 3
Enter Student's name :Name3
Enter Department name :
Department name cannot be empty. Enter again
D3
Enter College name : NITW
-----
4
Enter student id : 4
Enter Student's name :Name4
Enter Department name : D6
Enter College name : NITW
-----
5
Enter student id : 5
Enter Student's name :Name1
Enter Department name : D1
Enter College name : NITW
```



```
Q31 x
Enter student id : 5
Enter Student's name :Name1
Enter Department name : D1
Enter College name : NITW
-----
6
Enter student id : 6
Enter Student's name :Name 7
Enter Department name : D2
Enter College name : NITW
-----
7
Enter student id : 7
Enter Student's name :Name8
Enter Department name : D3
Enter College name : NITW
-----
8
Enter student id : 8
Enter Student's name :Name9
Enter Department name : D4
Enter College name : NITW
-----
9
Enter student id : 9
Enter Student's name :Name10
Enter Department name : D5
Enter College name : NItw
-----
10
Enter student id : 10
Enter Student's name :Name11
Enter Department name : D6
Enter College name : NITW
```

```

-----
1   Name1   D1   NITW
2   Name2   D1   NITW
3   Name3   D3   NITW
4   Name4   D6   NITW
5   Name1   D1   NITW
6   Name 7   D2   NITW
7   Name8   D3   NITW
8   Name9   D4   NITW
9   Name10  D5   NItw
10  Name11  D6   NITW
Writing records into text file
success

Process finished with exit code 0

```

Q-32 Write a java program called factorial.java that computes the factorials and catches the result in a long variable. The long type of variable has its own range. For example 20! is as high as the range of long types. So check the argument passed and throw an exception, if it is too big or too small. If x is less than 0 throw an IllegalArgumentException with a message. If x is more than a specific range then throw a custom exception saying "Result will overflow". Here x is a value for which we want to find factorial (it should be user-defined).

Ans-32

CODE

```

import java.util.Scanner;
class overflow extends ArithmeticException {
    public overflow(String s) {
        super(s);
    }
}

public class factorial{
    static long fact(int n) {
        if(n<=1){
            return 1;
        }
        return n*fact(n-1);
    }
}

public static void main(String[] args) {
    Scanner sc=new Scanner(System.in);
    try {
        int num;
        System.out.print("Enter a number : ");
        num = sc.nextInt();
    }
}

```

```

    if (num < 0) {
        throw new IllegalArgumentException("Number less than 0!");
    }
    else if (num > 20){
        throw new overflow("Big Number overflow");
    }
    System.out.println("factorial : "+ fact(num));
}
catch(IllegalArgumentException ie) {
    System.out.println(ie.getMessage());
}
catch(overflow o) {
    System.out.println(o.getMessage());
}
}
}

```

Output

```

factorial ×
"C:\Program Files\Java\jdk-17.0.2\bin
Enter a number : 52
Big Number overflow

Process finished with exit code 0

```

```

factorial ×
"C:\Program Files\Java\jdk-17.0.2\bin
Enter a number : -3
Number less than 0!

Process finished with exit code 0

```

```

factorial ×
"C:\Program Files\Java\jdk-17.0.2\bin
Enter a number : 10
factorial : 3628800

Process finished with exit code 0

```

Q-33 Write a java program to write the name and phone numbers of 15 people inside a file. Each line contains a name and phone number. Now enter a user defined variable called phone number, you need to match (compare) this phone number against all the phone numbers in the file, if matches then print corresponding name on to the screen, if not present then throw a custom exception.

Ans-33

CODE

```
import java.io.*;
import java.io.IOException;
import java.util.Scanner;
class contact implements Serializable {
    String name;
    long pn;
    contact(String n , long p) {
        name=n;
        pn=p;
    }
}
class recordnotfound extends Exception {
    public recordnotfound(String s) {
        super(s);
    }
}

public class Q33 {
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        contact[] record=new contact[15];
        String s;
        int i,flag=0;
        long num,key;
        try {

            for(i=0;i<15;i++) {
                System.out.println(i+1);
                System.out.print("Enter Name : ");
                s=sc.nextLine();
                System.out.print("Enter Phone No. : ");
                num=sc.nextLong();
                sc.nextLine();
                record[i]=new contact(s, num);
                System.out.println("_____");
            }
            FileOutputStream fos=new FileOutputStream("phonebook");
            ObjectOutputStream oos=new ObjectOutputStream(fos);
            oos.writeObject(record);
            oos.flush();
            oos.close();
        }
        catch(IOException ie) {
            System.out.println("exception occurred during writing....");
            System.exit(0);
        }
        System.out.print("Enter phone number to be searched : ");
        key=sc.nextLong();
        sc.nextLine();
        try {
```

```

contact[] temp=new contact[15];
FileInputStream fis = new FileInputStream("phonebook");
ObjectInputStream ois = new ObjectInputStream(fis);
temp =(contact[])ois.readObject();
for(i=0;i<15;i++) {
    if(temp[i].pn==key) {
        System.out.println("contact name- "+temp[i].name);
        ois.close();
        flag=1;
        break;
    }
}
if(flag==0) {
    throw new recordnotfound("record not found");
}
}
catch(recordnotfound re) {
    System.out.println(re.getMessage());
    System.exit(0);
}
catch(Exception ce) {
    System.out.println("exception occurred during searching....");
    System.exit(0);
}
}
}

```

Output

Q33 x

```
"C:\Program Files\Java\jdk-17.0.2\bin\java.e
```

```
1
```

```
Enter Name : Name1
```

```
Enter Phone No. : 0000
```

```
-----
```

```
2
```

```
Enter Name : Name2
```

```
Enter Phone No. : 1111
```

```
-----
```

```
3
```

```
Enter Name : Name3
```

```
Enter Phone No. : 2222
```

```
-----
```

```
4
```

```
Enter Name : Name4
```

```
Enter Phone No. : 3333
```

```
-----
```

```
5
```

```
Enter Name : Name5
```

```
Enter Phone No. : 4444
```

```
-----
```

```
6
```

```
Enter Name : Name6
```

```
Enter Phone No. : 5555
```

```
-----
```

```
7
```

```
Enter Name : Name7
```

```
Enter Phone No. : 6666
```

```
-----
```

```
8
```

```
Enter Name : Name8
```

```
Enter Phone No. : 7777
```

```
-----
```

```
9
```

```
Enter Name : Name9
```

```
Enter Phone No. : 8888
```

```
-----
```

```
Q33 x
8
Enter Name : Name8
Enter Phone No. : 7777
-----
9
Enter Name : Name9
Enter Phone No. : 8888
-----
10
Enter Name : Name10
Enter Phone No. : 9999
-----
11
Enter Name : Name11
Enter Phone No. : 1010
-----
12
Enter Name : Name12
Enter Phone No. : 1111
-----
13
Enter Name : Name13
Enter Phone No. : 1212
-----
14
Enter Name : Name14
Enter Phone No. : 1313
-----
15
Enter Name : Name15
Enter Phone No. : 1414
-----
Enter phone number to be searched : 9999
contact name- Name10

Process finished with exit code 0
```

Q-34 Write a program to show usage of throws keywords.

Ans-34

CODE

```
import java.util.Scanner;
```

```

public class Q34 {
    //If someone try to give Position out of Array indices it throws exceptions
    //Which do not interrupt program even with exception found.
    static void modify(int pos ,int val, int arr[]) throws IndexOutOfBoundsException {
        arr[pos]=val;
        System.out.println("success");
    }
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        int[] arr={11,22,33,44,55};
        System.out.print("Enter the position to be modified : ");
        int p=sc.nextInt();
        System.out.print("Enter new value : ");
        int v=sc.nextInt();
        try {
            modify(p,v,arr);
        }
        catch(IndexOutOfBoundsException ie) {
            System.out.println(ie.getMessage());
        }

        System.out.println("Remaining Code Executed");
    }
}

```

Output

```

Q34 x
"C:\Program Files\Java\jdk-17.0.2\bin\
Enter the position to be modified : 12
Enter new value : 123
Index 12 out of bounds for length 5
Remaining Code Executed

Process finished with exit code 0

```

Q-35 Write a program to show rethrowing exceptions.

Ans-35

CODE

```

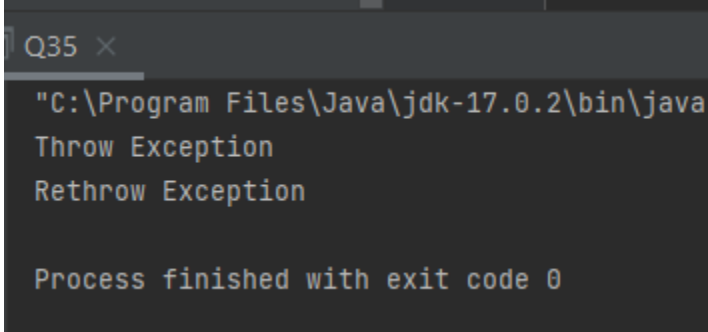
class rethrow{
    static void divide(int n1,int n2) {
        double q;
        try {
            q=n1/n2;
        }
        catch(ArithmeticException ae) {
            System.out.println("Throw Exception");
            throw ae;
        }
        System.out.println(q);
    }
}

```



```
}  
}  
public class Q35 {  
    public static void main(String[] args) {  
        try {  
            rethrow.divide(10, 0);  
        }  
        catch(Exception e) {  
            System.out.println("Rethrow Exception");  
        }  
    }  
}
```

Output



```
Q35 x  
"C:\Program Files\Java\jdk-17.0.2\bin\java  
Throw Exception  
Rethrow Exception  
  
Process finished with exit code 0
```
