Ex.No.4

Programming with Interface

09.09.2020

Mugilan E.S.

2019202033

1.

I. Define an abstract class by name Solid that has the data member radius of double type with default scope. The class Solid has three public abstract methods by name surfaceArea(), volume(), readRadius(). The class Solid also has a concrete method by name baseArea that computes πr^2 by receiving r as an argument which is double type.

II. Two classes by name Cylinder and Sphere inherit the Solid class and implement the methods readRadius(), surfaceArea() and volume(). readRadius() accepts the radius, surfaceArea()computes surface area and volume() computes the volume of the concerned shape. Both the classes should make use of the baseArea method of the abstract class wherever possible.

III. Define another class by name MySolid with the main method that displays the surface area and volume of a cylinder and a sphere.

[Hint: surface area and volume of a cylinder are $2\pi r^2+2\pi rh$ and πr^2h respectively. surface area and volume of a sphere are $4\pi r^2$ and $4/3\pi r^3$ respectively.]

Solid.java

```
package lab.four.solids;
public abstract class Solid {
    double radius;
    public abstract double surfaceArea();
    public abstract double volume();
    public abstract void readRadius();
    public double baseArea(double radius) {
        return Math.PI * radius * radius;
    }
}
```

Cylinder.java

```
package lab.four.solids;
import java.util.Scanner;
public class Cylinder extends Solid {
    double height;
    @Override
    public double surfaceArea() {
        return (2 * baseArea(radius)) + (2 * (baseArea(radius) / radius) * height);
    }
    @Override
    public double volume() {
        return baseArea(radius) * height;
    }
    @Override
    public void readRadius() {
```

```java
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the Radius of the Cylinder: ");
        radius = input.nextDouble();
        input.close();
    }
}
```

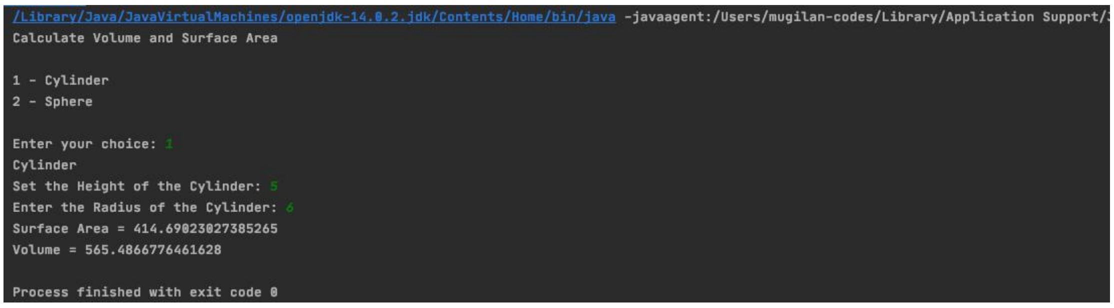
Sphere.java

```java
package lab.four.solids;
import java.util.Scanner;
public class Sphere extends Solid {
    @Override
    public double surfaceArea() {
        return 4 * baseArea(radius);
    }
    @Override
    public double volume() {
        return (4 * baseArea(radius) * radius) / 3;
    }
    @Override
    public void readRadius() {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the Radius of the Sphere: ");
        radius = input.nextDouble();
        input.close();
    }
}
```

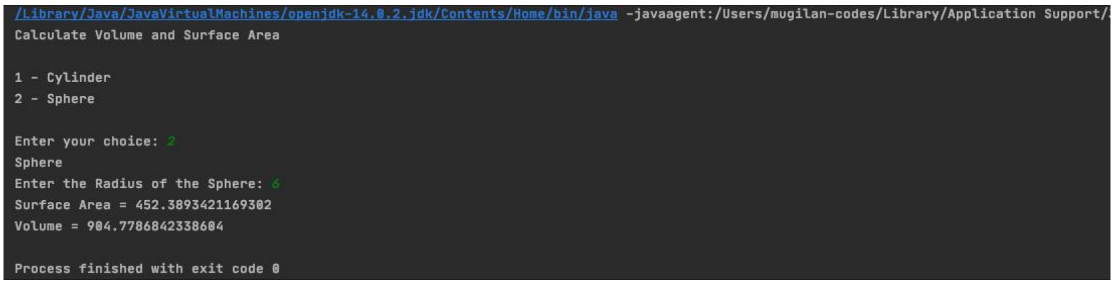
MySolid.java

```java
package lab.four.solids;

import java.util.Scanner;
public class MySolid {
    public static void main(String[] args) {
        System.out.println("Calculate Volume and Surface Area");
        System.out.println("\n1 - Cylinder\n2 - Sphere\n");
        Scanner input = new Scanner(System.in);
        System.out.print("Enter your choice: ");
        int choice = input.nextInt();
        switch (choice) {
            case 1 -> {
                System.out.println("Cylinder");
                Cylinder cylinder = new Cylinder();
                System.out.print("Set the Height of the Cylinder: ");
                cylinder.height = input.nextDouble();
                cylinder.readRadius();
                System.out.println("Surface Area = " + cylinder.surfaceArea());
                System.out.println("Volume = " + cylinder.volume());
            }
            case 2 -> {
                System.out.println("Sphere");
                Sphere sphere = new Sphere();
                sphere.readRadius();
                System.out.println("Surface Area = " + sphere.surfaceArea());
                System.out.println("Volume = " + sphere.volume());
            }
            default -> System.out.println("Invalid Choice");
        }
    }
}
```

Output:



```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/
Calculate Volume and Surface Area

1 - Cylinder
2 - Sphere

Enter your choice: 1
Cylinder
Set the Height of the Cylinder: 5
Enter the Radius of the Cylinder: 6
Surface Area = 414.69023027385265
Volume = 565.4866776461628

Process finished with exit code 0
```



```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/
Calculate Volume and Surface Area

1 - Cylinder
2 - Sphere

Enter your choice: 2
Sphere
Enter the Radius of the Sphere: 6
Surface Area = 452.3893421169302
Volume = 904.7786842338604

Process finished with exit code 0
```

2.

Class Dog can extend to class "Animal" and implement interface as "Pet".

Pet.java

```java
package lab.four.animals;
public interface Pet {
    void play();
}
```

Animal.java

```java
package lab.four.animals;

public class Animal {
    void run() {
        System.out.println("Animal is Running...");
    }
}
```

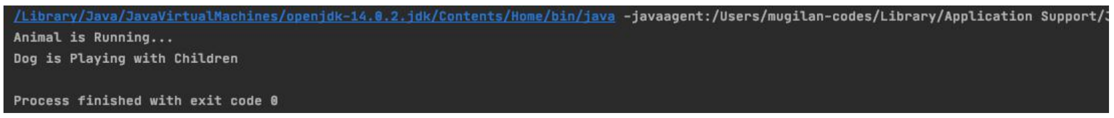
Dog.java

```java
package lab.four.animals;

public class Dog extends Animal implements Pet {
    @Override
    public void play() {
        System.out.println("Dog is Playing with Children");
    }
}
```

Main.java

```java
package lab.four.animals;
public class Main {
    public static void main(String[] args) {
        Dog dog = new Dog();
        dog.run();
        dog.play();
    }
}
```

Output:



```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/J
Animal is Running...
Dog is Playing with Children

Process finished with exit code 0
```

3.

Create an interface ILoan with homeLoan() and vehicleLoan() as method signature. Implement this interface in BankLoan class which is an abstract class with an abstract method calcInterst(). Derive two
classes HomeLoan and VechicleLoan from the base class. A person is eligible for a home loan amount of Rs.20,00,000 if his income is above Rs. 50,000 with an interest rate of 9.5%. A person is eligible for a vehicle loan of 12,00,000 if his income is above Rs. 60,000 with an interest rate of 10.5%. Write a java program to demonstrate runtime polymorphism for the above scenario with an array of objects.

ILoan.java

```
package lab.four.loan;
public interface ILoan {
	void homeLoan();
	void vehicleLoan();
}
```

BankLoan.java

```
package lab.four.loan;
abstract class BankLoan implements ILoan {
	abstract void calcInterest();
}
```

HomeLoan.java

```
package lab.four.loan;
public class HomeLoan extends BankLoan {
	double salary;
	HomeLoan(double income) {
		this.salary = income;
	}
	@Override
	void calcInterest() {
		System.out.println("Salary of Rs." + salary);
		if(salary > 50000) {
			homeLoan();
			if (salary > 60000) {
				System.out.print("Also ");
				vehicleLoan();
				System.out.println();
			}
		} else {
			System.out.println("Not Eligible for Home Loan");
		}
	}
	@Override
	public void homeLoan() {
		System.out.println("Eligible for Home Loan of Rs. 20,00,000 with an interest rate of 9.5%");
	}
	@Override
	public void vehicleLoan() {
		System.out.print("Eligible for Vehicle Loan of Rs.12,00,000 with an interest rate of 10.5%");
	}
}
```

VehicleLoan.java

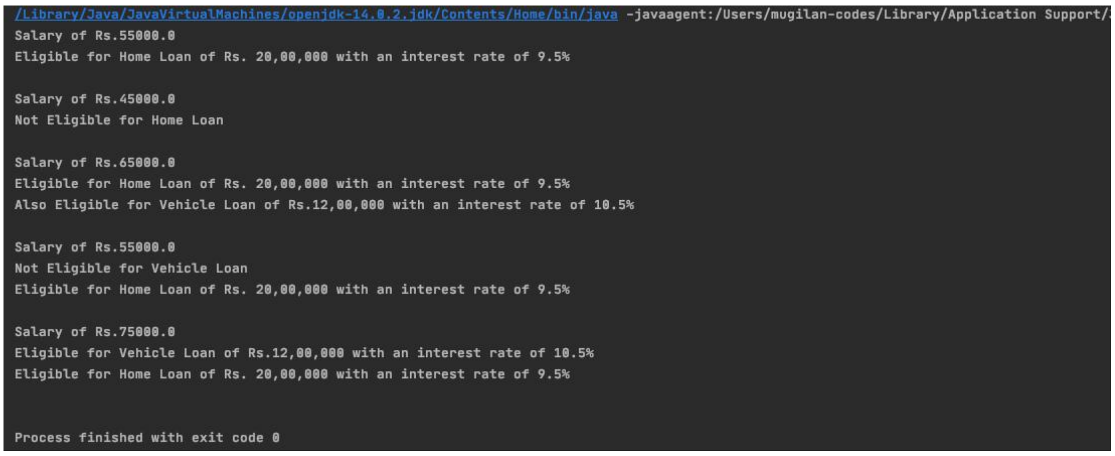
```java
package lab.four.loan;
public class VehicleLoan extends BankLoan {
	double salary;
	VehicleLoan(double income) {
		this.salary = income;
	}
	@Override
	void calcInterest() {
		System.out.println("Salary of Rs." + salary);
		if(salary > 60000) {
			vehicleLoan();
			homeLoan();
		} else {
			System.out.println("Not Eligible for Vehicle Loan");
			if(salary > 50000) {
				homeLoan();
			}
		}
	}
	@Override
	public void homeLoan() {
		System.out.println("Eligible for Home Loan of Rs. 20,00,000 with an interest rate of 9.5%");
	}
	@Override
	public void vehicleLoan() {
		System.out.println("Eligible for Vehicle Loan of Rs.12,00,000 with an interest rate of 
10.5%");
	}
}
```

Loan.java

```java
package lab.four.loan;
public class Loan {
	public static void main(String[] args) {
		BankLoan[] bankLoans = new BankLoan[5];
		bankLoans[0] = new HomeLoan(55000);
		bankLoans[1] = new HomeLoan(45000);
		bankLoans[2] = new HomeLoan(65000);
		bankLoans[3] = new VehicleLoan(55000);
		bankLoans[4] = new VehicleLoan(75000);
		for (BankLoan banks: bankLoans) {
			banks.calcInterest();
```

```java
                System.out.println();
            }
        }
}
```

Output:



```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/
Salary of Rs.55000.0
Eligible for Home Loan of Rs. 20,00,000 with an interest rate of 9.5%

Salary of Rs.45000.0
Not Eligible for Home Loan

Salary of Rs.65000.0
Eligible for Home Loan of Rs. 20,00,000 with an interest rate of 9.5%
Also Eligible for Vehicle Loan of Rs.12,00,000 with an interest rate of 10.5%

Salary of Rs.55000.0
Not Eligible for Vehicle Loan
Eligible for Home Loan of Rs. 20,00,000 with an interest rate of 9.5%

Salary of Rs.75000.0
Eligible for Vehicle Loan of Rs.12,00,000 with an interest rate of 10.5%
Eligible for Home Loan of Rs. 20,00,000 with an interest rate of 9.5%


Process finished with exit code 0
```

4.

Consider an interface Shape with computeArea() as method signature. Implement this in an abstract class Shape with compCircumference() as abstract method. Derive Square, Rectangle and Circle classes from Shape class and override computeArea() member function. Create an array of Shape class objects and demonstrate runtime polymorphism.

IShape.java

```java
package lab.four.shape;
public interface IShape {
    void computeArea();
}
```

Shape.java

```java
package lab.four.shape;

abstract class Shape implements IShape {
    abstract void compCircumference();
}
```

Square.java

```java
package lab.four.shape;
public class Square extends Shape{
    double side;
    Square(double side) {
        this.side = side;
    }
    @Override
    void compCircumference() {
        double circumference = 4 * side;
        System.out.println("Circumference of Square of side " + side + " cm = " + circumference);
    }
    @Override
    public void computeArea() {
        double area = side * side;
        System.out.println("Area of square of side " + side + " cm = " + area);
    }
}
```

Rectangle.java

```java
package lab.four.shape;
public class Rectangle extends Shape {
    double length, width;
    Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }
    @Override
    void compCircumference() {
        double circumference = 2 * (length + width);
        System.out.println("Circumference of Rectangle of length " + length + " cm and width " +
width + " = " + circumference);
    }
    @Override
    public void computeArea() {
        double area = length * width;
        System.out.println("Area of Rectangle of length " + length + " cm and width " + width + " =
" + area);
    }
}
```

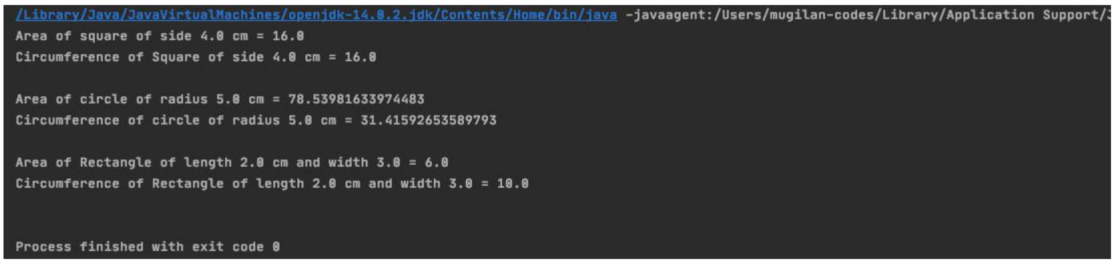
Circle.java

```java
package lab.four.shape;
public class Circle extends Shape {
    double radius;
    Circle(double radius) {
        this.radius = radius;
    }
    @Override
    void compCircumference() {
        double circumference = 2 * Math.PI * radius;
        System.out.println("Circumference of circle of radius " + radius + " cm = " + circumference);
    }
    @Override
    public void computeArea() {
        double area = Math.PI * radius * radius;
        System.out.println("Area of circle of radius " + radius + " cm = " + area);
    }
}
```

ArrayOfShape.java

```java
package lab.four.shape;
public class ArrayOfShape {
    public static void main(String[] args) {
        Shape[] shapes = new Shape[3];
        shapes[0] = new Square(4);
        shapes[1] = new Circle(5);
        shapes[2] = new Rectangle(2,3);
        for(Shape shape: shapes) {
            shape.computeArea();
            shape.compCircumference();
            System.out.println();
        }
    }
}
```

Output:



```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/J
Area of square of side 4.0 cm = 16.0
Circumference of Square of side 4.0 cm = 16.0

Area of circle of radius 5.0 cm = 78.53981633974483
Circumference of circle of radius 5.0 cm = 31.41592653589793

Area of Rectangle of length 2.0 cm and width 3.0 = 6.0
Circumference of Rectangle of length 2.0 cm and width 3.0 = 10.0


Process finished with exit code 0
```

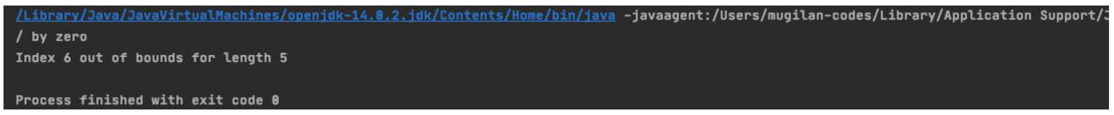
5. Exception Handling

1. Develop a Java program that
demonstrates ArithmeticException and ArrayIndexOutOfBoundsException. (Both the exceptions should be handled in the same program using a single array.)

ProgramOne.java

```
package lab.four.exceptions;
public class ProgramOne {
    public static void main(String[] args) {
        int[] arr = new int[5];
        try {
            arr[0] = 50 / 0;
        } catch (ArithmeticException e) {
            System.out.println(e.getMessage());
        }
        try {
            arr[6] = 60;
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Output:



```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/J
/ by zero
Index 6 out of bounds for length 5

Process finished with exit code 0
```

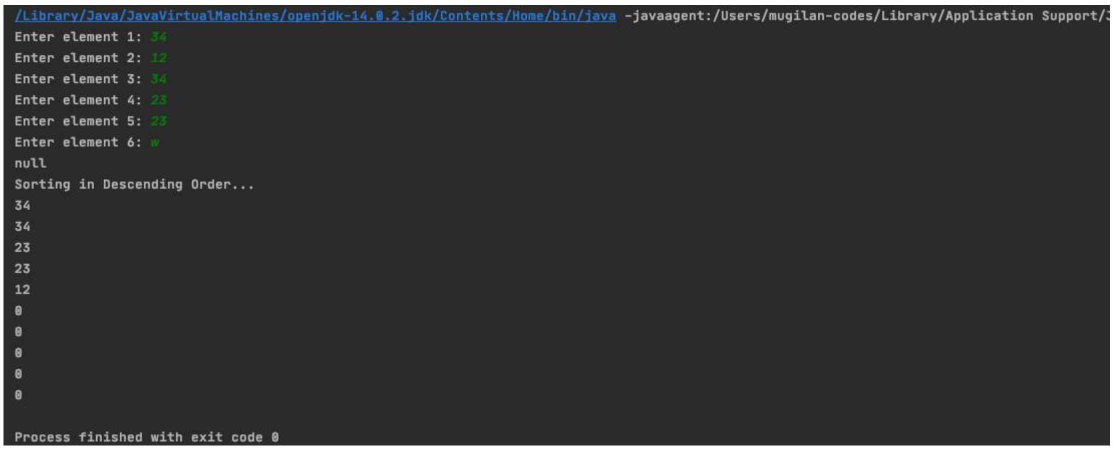
2. Develop a Java program that accepts 10 integers in an array and sort them in ascending order (Use any sorting algorithm). The program should handle two exceptions namely NumberFormatException (or InputMismatchException) and ArrayIndexOutOfBoundsException. NumberFormatException is to be handled when you accept the contents of the array
and ArrayIndexOutofBoundsException when you display the contents of the sorted array.

ProgramTwo.java

```
package lab.four.exceptions;
import java.util.InputMismatchException;
import java.util.Scanner;
public class ProgramTwo {
    public static void main(String[] args) {
        int[] arr = new int[10];
```

```java
        Scanner sc = new Scanner(System.in);
        try {
            for (int i = 0; i < arr.length; i++) {
                System.out.print("Enter element " + (i + 1) + ": ");
                arr[i] = sc.nextInt();
            }
        } catch (InputMismatchException e) {
            System.out.println(e.getMessage());
        }
        System.out.println("Sorting in Descending Order...");
        for(int i=0;i< arr.length;i++) {
            for(int j=i+1; j< arr.length;j++) {
                int temp =0;
                if(arr[i]<arr[j]) {
                    temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }
        try {
            for (int j : arr) {
                System.out.println(j);
            }
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Output:



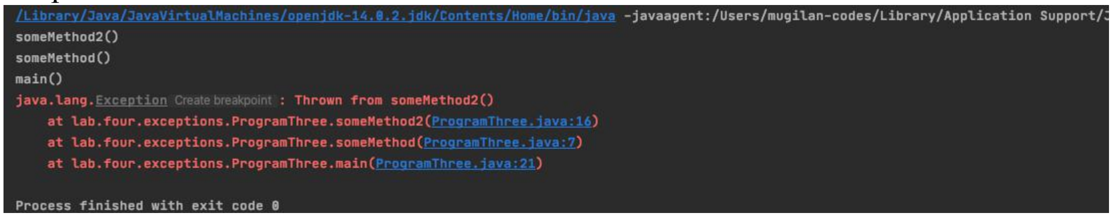
```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/J
Enter element 1: 34
Enter element 2: 12
Enter element 3: 34
Enter element 4: 23
Enter element 5: 23
Enter element 6: w
null
Sorting in Descending Order...
34
34
23
23
12
0
0
0
0
0

Process finished with exit code 0
```

3. Write a program that illustrates rethrowing an exception. Define methods someMethod and someMethod2. Method someMethod2 should initially throw an exception. Method someMethod should call someMethod2, catch the exception and rethrow it. Call someMethod from method main, and catch the rethrown exception. Print the stack trace of this exception.

ProgramThree.java

```java
ppackage lab.four.exceptions;
public class ProgramThree {
    public static void someMethod() throws Throwable {
        try {
            someMethod2();
        } catch (Exception e) {
            System.out.println("someMethod()");
            throw e;
        }
    }
    public static void someMethod2() throws Exception {
        System.out.println("someMethod2()");
        throw new Exception("Thrown from someMethod2()");
    }
    public static void main(String[] args) throws Throwable {
        try {
            someMethod();
        } catch (Exception e) {
            System.out.println("main()");
            e.printStackTrace();
        }
    }
}
```

Output:



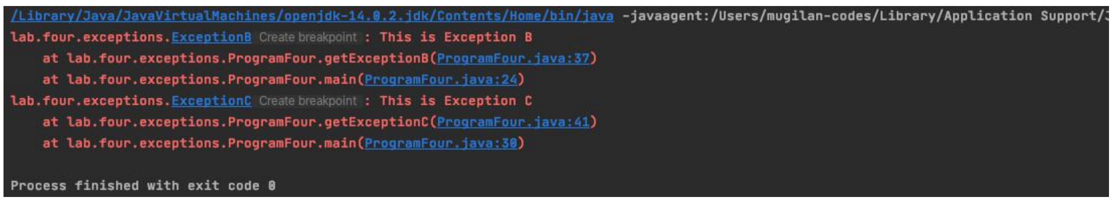
```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/J
someMethod2()
someMethod()
main()
java.lang.Exception Create breakpoint : Thrown from someMethod2()
    at lab.four.exceptions.ProgramThree.someMethod2(ProgramThree.java:16)
    at lab.four.exceptions.ProgramThree.someMethod(ProgramThree.java:7)
    at lab.four.exceptions.ProgramThree.main(ProgramThree.java:21)

Process finished with exit code 0
```

4. Use inheritance to create an exception superclass (called ExceptionA) and exception subclasses ExceptionB and ExceptionC, where ExceptionB inherits from ExceptionA and ExceptionC inherits from ExceptionB. Write a program to demonstrate that the catch block for type ExceptionA catches exceptions of types ExceptionB and ExceptionC.

ProgramFour.java

```java
package lab.four.exceptions;
class ExceptionA extends Exception {
	ExceptionA(String s) {
		super(s);
	}
}
class ExceptionB extends ExceptionA {
	ExceptionB(String s) {
		super(s);
	}
}
class ExceptionC extends ExceptionB {
	ExceptionC(String s) {
		super(s);
	}
}
public class ProgramFour {
	public static void main(String[] args) {
		try{
			getExceptionB();
		} catch (ExceptionA e) {
			e.printStackTrace();
		}
		try {
			getExceptionC();
		} catch (ExceptionA e) {
			e.printStackTrace();
		}
	}
	public static void getExceptionB() throws ExceptionB {
		throw new ExceptionB("This is Exception B");
	}
	public static void getExceptionC() throws ExceptionC {
		throw new ExceptionC("This is Exception C");
	}
}
```

Output:



```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/J
lab.four.exceptions.ExceptionB Create breakpoint : This is Exception B
    at lab.four.exceptions.ProgramFour.getExceptionB(ProgramFour.java:37)
    at lab.four.exceptions.ProgramFour.main(ProgramFour.java:24)
lab.four.exceptions.ExceptionC Create breakpoint : This is Exception C
    at lab.four.exceptions.ProgramFour.getExceptionC(ProgramFour.java:41)
    at lab.four.exceptions.ProgramFour.main(ProgramFour.java:30)

Process finished with exit code 0
```

5. ***(Catching Exceptions Using Class Exception)*** Write a program that demonstrates how various exceptions are caught with

catch (Exception exception)

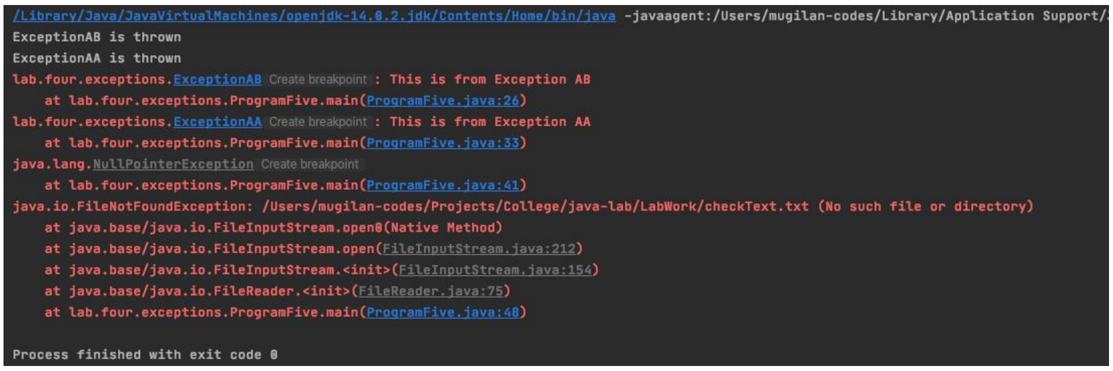
This time, define classes ExceptionA (which inherits from class Exception) and ExceptionB (which inherits from class ExceptionA). In your program, create try blocks that throw exceptions of
types ExceptionA, ExceptionB, NullPointerException and IOException. All exceptions should be caught with catch blocks specifying type Exception.

ProgramFive.java

```java
package lab.four.exceptions;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
class ExceptionAA extends Exception {
	ExceptionAA() {
		super("This is from Exception AA");
	}
	ExceptionAA(String s) {
		super(s);
	}
}
class ExceptionAB extends ExceptionAA {
	ExceptionAB() {
		super("This is from Exception AB");
	}
}
public class ProgramFive {
	public static void main(String[] args) {
		try{
			throw new ExceptionAB();
		} catch (Exception e) {
			System.out.println("ExceptionAB is thrown");
			e.printStackTrace();
		}
		try {
			throw new ExceptionAA();
		} catch (Exception e) {
			System.out.println("ExceptionAA is thrown");
			e.printStackTrace();
		}
		try {
			String str = null;
```

```
                System.out.println(str.toString());
            } catch (NullPointerException e) {
                e.printStackTrace();
            }
            try{
                File file = new File("checkText.txt");
                FileReader fr = new FileReader(file.getAbsoluteFile());
                System.out.println(fr);
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
}
```

Output:



```
/Library/Java/JavaVirtualMachines/openjdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/mugilan-codes/Library/Application Support/
ExceptionAB is thrown
ExceptionAA is thrown
lab.four.exceptions.ExceptionAB Create breakpoint : This is from Exception AB
    at lab.four.exceptions.ProgramFive.main(ProgramFive.java:26)
lab.four.exceptions.ExceptionAA Create breakpoint : This is from Exception AA
    at lab.four.exceptions.ProgramFive.main(ProgramFive.java:33)
java.lang.NullPointerException Create breakpoint
    at lab.four.exceptions.ProgramFive.main(ProgramFive.java:41)
java.io.FileNotFoundException: /Users/mugilan-codes/Projects/College/java-lab/LabWork/checkText.txt (No such file or directory)
    at java.base/java.io.FileInputStream.open0(Native Method)
    at java.base/java.io.FileInputStream.open(FileInputStream.java:212)
    at java.base/java.io.FileInputStream.<init>(FileInputStream.java:154)
    at java.base/java.io.FileReader.<init>(FileReader.java:75)
    at lab.four.exceptions.ProgramFive.main(ProgramFive.java:48)

Process finished with exit code 0
```

Source Code:

https://github.com/Mugilan-Codes/java-lab-exercises