S.No: 1	Exp. Name: ***sample programs on operator precedence and associativity***	Date: 2023-11-10

Aim:

Write a java program to demonstrate operator precedence and associativity

Source Code:

OperatorPrecedence.java

```
import java.util.Scanner;
public class OperatorPrecedence
{
        public static void main(String[]args)
        {
             Scanner scanner = new Scanner(System.in);
             System.out.print("Enter a num: ");
             int x = scanner.nextInt();
             int result = x++ + x++ * --x / x++ - --x + 3 >> 1 | 2;
             System.out.println("The operation going is x++ + x++ * --x / x++ - --x + 3 >> 1
| 2");
             System.out.println("result = " + result);
             scanner.close();

        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter a num:
4
The operation going is x++ + x++ * --x / x++ - --x + 3 >> 1 | 2
result = 3

Test Case - 2
User Output
Enter a num:
-3
The operation going is x++ + x++ * --x / x++ - --x + 3 >> 1 | 2
result = 2

S.No: 2	Exp. Name: ***Sample program on java to demonstrate Control structures***	Date: 2023-11-09

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

Aim:

write a java program that uses if-else control statement and print the result

Source Code:

Control.java

```
import java.util.Scanner;
public class Control
{
        public static void main(String[] args)
        {
                Scanner scanner = new Scanner(System.in);
                System.out.print("Enter first num : ");
                int x = scanner.nextInt();
                System.out.print("Enter second num : ");
                int y = scanner.nextInt();
                if(x + y<20)
                {
                        System.out.println("x + y is less than 20");
                }
                else
                {
                        System.out.println("x + y is greater than 20");
                }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter first num :
13
Enter second num :
5
x + y is less than 20

Test Case - 2
User Output
Enter first num :
24
Enter second num :
10
x + y is greater than 20

S.No: 3	Exp. Name: ***Sample Program to demonstrate constructor***	Date: 2023-11-09

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

Aim:

Write a program to demonstrate constructor class

Source Code:

Student.java

```
public class Student
{
        int num;
        String text;
        public Student()
        {
                num = 0;
                text = null;
        }
        public void dispaly()
        {
                System.out.println(num + " " + text);
        }
        public static void main(String[] args)
        {
                Student obj1 = new Student();
                obj1.dispaly();
                Student obj2 = new Student();
                obj2.dispaly();
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
0 null
0 null

S.No: 4	Exp. Name: ***Sample program to demonstrate destructor***	Date: 2023-11-09

Aim:

Write a program to demonstrate destructor class

Source Code:

DestructorExample.java

```
public class DestructorExample
{
        public static void main(String[] args)
        {
                createObject();
                System.gc();
                System.out.println("Inside the main() method");
                System.gc();
                System.out.println("Object is destroyed by the Garbage Collector");
        }
        private static void createObject()
        {
                Object obj = new Object();
                System.out.println("Object is destroyed by the Garbage Collector");
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Object is destroyed by the Garbage Collector
Inside the main() method
Object is destroyed by the Garbage Collector

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

S.No: 5	Exp. Name: ***A program to print Half pyramid pattern***	Date: 2023-11-09

Aim:

Write a Java program to print Half Pyramid pattern.

Source Code:

HalfPyramid.java

```
import java.util.Scanner;
public class HalfPyramid
{
        public static void main(String[] args)
        {
                Scanner scanner = new Scanner(System.in);
                System.out.print("Enter no of rows : ");
                int rows = scanner.nextInt();
                for (int i = 1; i <= rows; i++)
                {
                        for (int j = 1; j <= i; j++)
                        {
                                System.out.print("* ");
                        }
                        System.out.println();
                }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter no of rows :
5
*
* *
* * *
* * * *
* * * * *

Test Case - 2
User Output
Enter no of rows :
3
*
* *
* * *

Test Case - 3
User Output
Enter no of rows :
10
*
* *
* * *
* * * *
* * * * *
* * * * * *
* * * * * * *
* * * * * * * *
* * * * * * * * *
* * * * * * * * * *

S.No: 6	Exp. Name: ***A program to print Inverted Half pyramin pattern***	Date: 2023-11-09

Aim:

Write a Program to Print Inverted Half Pyramid Pattern

Source Code:

HalfPyramidRev.java

```
import java.util.Scanner;
public class HalfPyramidRev
{
        public static void main(String[] args)
        {
                Scanner scanner = new Scanner(System.in);
                System.out.print("Enter no of rows : ");
                int rows = scanner.nextInt();
                for (int i = rows; i >= 1; i--)
                {
                        for (int j = 1; j <= i; j++)
                        {
                                System.out.print("* ");
                        }
                        System.out.println();
                }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter no of rows :
5
* * * * *
* * * *
* * *
* *
*

Test Case - 2
User Output
Enter no of rows :
3
* * *
* *
*

ID: 234G5A0505
2022-2026-CSE-B
Srinivasa Ramanujan Institute of Technology

S.No: 7	Exp. Name: ***A program to print Hollow Inverted Half Pyramid Pattern***	Date: 2023-11-09

Aim:

Write a Program to Print Hollow Inverted half Pyramid Pattern

Source Code:

HollowHalfPyramidRev.java

```
// Type Content here...
import java.util.Scanner;
public class HollowHalfPyramidRev
{
        public static void main(String[] args)
        {
                Scanner scanner = new Scanner(System.in);
                System.out.print("Enter no of rows : ");
                int rows = scanner.nextInt();
                for (int i = rows; i >= 1; i--)
                {
                        if (i == rows || i == 1)
                        {
                                for (int j = 1; j <= i; j++)
                                {
                                        System.out.print("* ");
                                }
                        }
                    else
                {
                            for(int j = 1; j <= i; j++)
                            {
                               if (j == 1 || j ==i)
                            {
                                        System.out.print("* ");
                            }
                            else
                                {
                                    System.out.print("  ");
                                }
                        }
                }
                System.out.println();
                }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter no of rows :
5

```
* * * * *
*     *
*   *
* *
*
```

Test Case - 2

User Output

```
Enter no of rows :
3
* * *
* *
*
```

S.No: 8	Exp. Name: ***A program to print Pyramid Pattern***	Date: 2023-11-10

Aim:

Write a Program to Print Pyramid Pattern

Source Code:

Pyramid.java

```
import java.util.Scanner;
public class Pyramid
{
        public static void main(String[] args)
        {
                Scanner scanner = new Scanner(System.in);
                System.out.print("Enter no of rows : ");
                int row = scanner.nextInt();
                for (int i = 0; i < row; i++)
                {
                        for (int j=row-i-1;j>=1; j--)
                                System.out.print(" ");
                        for (int j=0; j<=i; j++)
                                System.out.print("* ");
                        System.out.println();
                }
    }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter no of rows :
5
*
* *
* * *
* * * *
* * * * *

Test Case - 2
User Output
Enter no of rows :
6
*
* *
* * *
* * * *
* * * * *
* * * * * *

S.No: 9	Exp. Name: ***A program to print Inverted Pyramid Pattern***	Date: 2023-11-10

Aim:

Write a Program to Print inverted Pyramid Pattern

Source Code:

PyramidRev.java

```
import java.util.Scanner;
public class PyramidRev
{
        public static void main(String[] args)
        {
                Scanner in = new Scanner(System.in);
                System.out.print("Enter no of rows : ");
                int row = in.nextInt();
                for(int i=0; i<row; i++)
                        {
                        for(int j=0;j<i;j++)
                                System.out.print(" ");
                        for(int j=i;j<row;j++)
                                System.out.print("* ");
                        System.out.println("");
                        }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
Enter no of rows :
5
* * * * *
 * * * *
  * * *
   * *
    *
```

Test Case - 2

User Output

```
Enter no of rows :
6
* * * * * *
 * * * * *
  * * * *
   * * *
    * *
```

S.No: 10	Exp. Name: ***A program to print Hollow Pyramid Pattern***	Date: 2023-11-10

Aim:

Write a Program to print the Hollow pyramid pattern

Source Code:

PyramidGap.java

```
// Type Content here...
import java.util.Scanner;

class PyramidGap{
        public static void main(String args[]){
                Scanner sc=new Scanner(System.in);
                System.out.print("Enter no of rows : ");
            int num=sc.nextInt();
            int a,b,c=0;
            for(a=1;a<num;a++){
                for(b=a;b<num;b++){
                        System.out.print(" ");
                }
                while(c !=(2*a-1)){
                        if(c==0 || c==2*a-2)
                        {
                                System.out.print("*");
                        }
                        else{
                                System.out.print(" ");
                        }
                        c++;
                }
                System.out.print(" ");
                c=0;
                System.out.println();
            }

                for(a=0;a<num;a++){
                        System.out.print("* ");
                }
                System.out.print("\n");
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter no of rows :
5
*
* *

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

```
   *   *
  *     *
* * * * *
```

Test Case - 2

User Output

```
Enter no of rows :
6
     *
    * *
   *   *
  *     *
 *       *
* * * * * *
```

S.No: 11	Exp. Name: ***A program to illustrate Inheritance***	Date: 2023-11-10

Aim:

Write Java program on use of Inheritance.

Create a classVehicle

- contains the data members **color** of String type and **speed** and **size** of integer data type.
- write a method**setVehicleAttributes()**to initialize the data members

Create another classCarwhich is derived from the classVehicle

- contains the data members**cc**and**gears**of**integer**data type
- write a method**setCarAttributes()**to initialize the data members
- write a method**displayCarAttributes()**which will display all the attributes.

Write another class InheritanceDemo with **main()** it receives five arguments **color**, **speed**, **size**, **cc** and **gears**.

Source Code:

InheritanceDemo.java

```
import java.util.Scanner;

class Vehicle
{
        String color;
        int speed;
        int size;
        public void setVehicleAttributes(String color, int speed, int size)
        {
                this.color = color;
                this.speed = speed;
                this.size = size;
        }
}
class Car extends Vehicle
{
        int cc;
        int gears;
        public void setCarAttributes(int cc, int gears)
        {
                this.cc = cc;
                this.gears = gears;
        }
        public void displayCarAttributes()
        {
                System.out.println("Color of Car : " + color);
                System.out.println("Speed of Car : " + speed);
                System.out.println("Size of Car : " + size);
                System.out.println("CC of Car : " + cc);
                System.out.println("No of gears of Car : " + gears);
        }

}
class InheritanceDemo{
        public static void main(String args[]){
        //      Scanner sc=new Scanner(System.in);
                Car car=new Car();
                String color=args[0];
                int speed=Integer.valueOf(args[1]);
                int size=Integer.valueOf(args[2]);
                int cc=Integer.valueOf(args[3]);
                int gears=Integer.valueOf(args[4]);
                car.setVehicleAttributes(color,speed,size);
                car.setCarAttributes(cc,gears);
                car.displayCarAttributes();
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Color of Car : Blue

Speed of Car : 100
Size of Car : 20
CC of Car : 1000
No of gears of Car : 5

Test Case - 2
User Output
Color of Car : Orange
Speed of Car : 120
Size of Car : 25
CC of Car : 900
No of gears of Car : 5

S.No: 12	Exp. Name: ***write a java program to prevent inheritance using abstract class.***	Date: 2023-11-09

Aim:

write a java program to prevent inheritance using abstract class.

- Create an abstract class `Shape`
- Create a class `Rectangle` which extends the class `Shape`
- Class Rectangle contains a method **draw** whcih prints **drawing rectangle**
- Create another class `circle1` which extends `Shape`
- Class circle1 contains a method **draw** whcih prints **drawing circle**
- Create a main class `TestAbstraction1`
- Create object for the class circle1 and called the method draw

Source Code:

TestAbstraction1.java

```
abstract class Shape
{
        abstract void draw();
}
class Rectangle extends Shape
{
        void draw()
        {
                System.out.println("drawing rectangle");
        }
}
class Circle extends Shape
{
        void draw()
        {
                System.out.println("drawing circle");
        }
}
public class TestAbstraction1
{
        public static void main(String[] args)
        {
                Circle circleObj = new Circle();
                circleObj.draw();
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
drawing circle

S.No: 13	Exp. Name: ***program on dynamic binding***	Date: 2023-11-09

Aim:

write a program on dynamic binding

Source Code:

Demo.java

```
class Human
{
        void walk()
        {
                System.out.println("Human walks");
        }
}
class Boy extends Human
{
        void walk()
        {
                System.out.println("Boy walks");
        }
}
public class Demo
{
        public static void main(String[] args)
        {
                Human humanObj = new Human();
                Boy boyObj = new Boy();
                Human humanRef = boyObj;
                humanRef.walk();
                humanObj.walk();
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Boy walks
Human walks

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

S.No: 14	Exp. Name: ***Sample program on method overloading***	Date: 2023-11-09

Aim:

Write a program on method overloading

Source Code:

Sample.java

```
public class Sample
{
        static void print(String a)
        {
                System.out.println(a);
        }
        static void print(String a, int b)
        {
                System.out.println(a + " " + b);
        }
        public static void main(String[] args)
        {
                print("a");
                print("a", 10);
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
a
a 10

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

S.No: 15	Exp. Name: ***Sample program on method overriding***	Date: 2023-11-10

Aim:

Write a program on method overriding

Source Code:

Bike.java

```
class Vehicle
{
        void run()
        {
                System.out.println("Bike is running safely");
        }
}
class Bike extends Vehicle
{
        public static void main(String[] args)
     {
                Bike obj = new Bike();
                obj.run();
      }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Bike is running safely

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

S.No: 16	Exp. Name: ***Write a Java program to implement Interface***	Date: 2023-11-10

Aim:

Write a Java program that implements an **interface**.

Create an interface called `Car` with two abstract methods `String getName()` and `int getMaxSpeed()`. Also declare one **default** method `void applyBreak()` which has the code snippet

```
System.out.println("Applying break on " + getName());
```

In the same interface include a **static** method `Car getFastestCar(Car car1, Car car2)`, which returns **car1** if the **maxSpeed** of **car1** is greater than or equal to that of **car2**, else should return **car2**.

Create a class called `BMW` which implements the interface `Car` and provides the implementation for the abstract methods **getName()** and **getMaxSpeed()** (make sure to declare the appropriate fields to store **name** and **maxSpeed** and also the constructor to initialize them).

Similarly, create a class called `Audi` which implements the interface `Car` and provides the implementation for the abstract methods **getName()** and **getMaxSpeed()** (make sure to declare the appropriate fields to store **name** and **maxSpeed** and also the constructor to initialize them).

Create a **public** class called `MainApp` with the **main()** method.
Take the input from the command line arguments. Create objects for the classes `BMW` and `Audi` then print the fastest car.

Note:

Java 8 introduced a new feature called `default` methods or `defender` methods, which allow developers to add new methods to the interfaces without breaking the existing implementation of these interface. These **default** methods can also be overridden in the implementing classes or made abstract in the extending interfaces. If they are not overridden, their implementation will be shared by all the implementing classes or sub interfaces.

Below is the syntax for declaring a `default` method in an **interface** :

```
public default void methodName() {
    System.out.println("This is a default method in interface");
}
```

Similarly, **Java 8** also introduced `static` methods inside interfaces, which act as regular static methods in classes. These allow developers group the utility functions along with the interfaces instead of defining them in a separate helper class.

Below is the syntax for declaring a `static` method in an **interface** :

```
public static void methodName() {
    System.out.println("This is a static method in interface");
}
```

Note: Please don't change the package name.

Source Code:

q11284/MainApp.java

```
package q11284;
interface Car
{
	String getName();
	int getMaxSpeed();
	default void applyBreak()
	{
		System.out.println("Applying break on " + getName());
	}
	static Car getFastestCar(Car car1, Car car2)
	{
		return car1.getMaxSpeed() >= car2.getMaxSpeed() ? car1 : car2;
	}
}
class BMW implements Car
{
	private String name;
	private int maxSpeed;
	public BMW(String name, int maxSpeed)
	{
		this.name = name;
		this.maxSpeed = maxSpeed;
	}

@Override
	public String getName()
	{
		return name;
	}

	public int getMaxSpeed()
	{
		return maxSpeed;
	}
}
class Audi implements Car
{
	private String name;
	private int maxSpeed;
	public Audi(String name, int maxSpeed)
	{
		this.name = name;
		this.maxSpeed = maxSpeed;
	}
	@Override
	public String getName()
	{
		return name;
	}
	@Override
	public int getMaxSpeed()
	{
		return maxSpeed;
	}
}
```

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

                String bmwName = args[0];
                int bmwMaxSpeed = Integer.parseInt(args[1]);
                String audiName = args[2];
                int audiMaxSpeed = Integer.parseInt(args[3]);
                Car bmw = new BMW(bmwName, bmwMaxSpeed);
                Car audi = new Audi(audiName, audiMaxSpeed);
                Car fastestCar = Car.getFastestCar(bmw, audi);
                System.out.println("Fastest car is : " + fastestCar.getName());
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Fastest car is : BMW

Test Case - 2
User Output
Fastest car is : Maruthi

S.No: 17	Exp. Name: ***Write the code to create an exception***	Date: 2023-11-10

Aim:

Write a Java program to create an exception.

Source Code:

q221/Exception1.java

```
package q221;
class Exception1{
        public static void main(String args[]){
        //      Scanner sc=new Scanner(System.in);
                int num1=21;
                int num2=0;
                try{
                        int res=num1/num2;
                }
                catch(Exception e){
                        System.out.println("Exception caught : divide by zero occurred");
                }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Exception caught : divide by zero occurred

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

S.No: 18	Exp. Name: ***Write the code for handling the exception***	Date: 2023-11-11

Aim:

Write a Java code for handling the exception.

Source Code:

q222/handleError.java

```
package q222;
import java.util.Random;
public class handleError {
        public static void main(String args[]) {
                int a = 0, b = 0, c = 0;
                Random r = new Random(100);
                for     (int i=0; i<32; i++)
                {
                        try
                        {
                                b=r.nextInt();
                                c=r.nextInt();
                                a=12345/(b/c);
                        }
                        catch(ArithmeticException e)
                        {
                                System.out.println("Division by zero.");
                    a=0;
                        }
                        System.out.println("a: " +a);
                }

        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
a: 12345
Division by zero.
a: 0
a: -1028
Division by zero.
a: 0
a: 12345
a: -12345
Division by zero.
a: 0
a: 3086
a: 12345
a: -12345

```
a: 12345
Division by zero.
a: 0
a: -12345
a: 12345
a: 342
a: 12345
a: -12345
a: 12345
a: -12345
Division by zero.
a: 0
a: -4115
Division by zero.
a: 0
a: -4115
a: 6172
a: 6172
Division by zero.
a: 0
Division by zero.
a: 0
Division by zero.
a: 0
a: 12345
a: -280
a: -12345
Division by zero.
a: 0
```

S.No: 19	Exp. Name: ***Write the code to create an exception using the predefined exception***	Date: 2023-11-10

Aim:

Write a Java code to create an exception using the predefined exception

Source Code:

q223/exception2.java

```
package q223;

public class exception2
{
	public static void main(String[] args)
	{
		try
		{
			int result = divideByZero();
			System.out.println("Result: " + result);
		}
		catch (ArithmeticException e)
		{
			System.out.println("Exception raised -Division by zero.");
		}
		finally
		{
			System.out.println("After catch statement.");
		}
	}
	static int divideByZero()
	{
		int numerator = 10;
		int denominator = 0;
		return numerator / denominator;
	}
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Exception raised -Division by zero.
After catch statement.

ID: 234G5A0505

S.No: 20	Exp. Name: ***Write the code for creating your own exception***	Date: 2023-11-10

Aim:

Write a Java code for creating your own exception

Source Code:

q224/demo.java

```
package q224;
class MyException extends Exception
{
        public MyException(String message)
        {
                super(message);
        }
}
public class demo
{
        public static void main(String[] args)
        {
                try
                {
                        int value = -10;
                        if (value < 0)
                        {
                                throw new MyException("MyException[" + value + "] is less
than zero");
                        }
                        System.out.println("Value is: " + value);
                }
                catch(MyException e)
                {
                        System.out.println(e.getMessage());
                }

        }

}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
MyException[-10] is less than zero

S.No: 21	Exp. Name: ***program that takes inputs 5 numbers, each between 10 and 100***	Date: 2023-11-10

Aim:

Write java program that inputs 5 numbers, each between 10 and 100 inclusive. As each number is read display it only if it's not a duplicate of any number already read. Display the complete set of unique values input after the user enters new values

Source Code:

Duplicate.java

```
import java.util.Scanner;
class Duplicate
{
	public static void main(String[] args)
	{
		Scanner sc= new Scanner(System.in);
		System.out.println("Enter 5 unique values between 10 & 100 ");
		int[] array=new int[5];
		for(int i=0;i<5;i++){
			int value=sc.nextInt();
			while(!(value>=10 && value<=100)){
				System.out.println("Entered value must be in between 10 &
100");
				value=sc.nextInt();
			}
			for(int j=0;j<5;j++){

				while(value == array[j]){
					System.out.println("Duplicate value found, retry");
					value=sc.nextInt();
				}
			}

			array[i]=value;
		}
		System.out.print("The five unique values are :");
		for(int k:array){
			System.out.print(k+" ");
		}

	}

}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter 5 unique values between 10 & 100
25
15

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

30
0
Entered value must be in between 10 & 100
34
89
The five unique values are :25 15 30 34 89

Test Case - 2
User Output
Enter 5 unique values between 10 & 100
48
92
34
92
Duplicate value found, retry
39
23
The five unique values are :48 92 34 39 23

S.No: 22	Exp. Name: ***A program to illustrate threads***	Date: 2023-12-22

Aim:

Write Java program(s) on creating multiple threads, assigning priority to threads, synchronizing threads, suspend and resume threads

Source Code:

TestThread.java

```
// Type Content here...
class MyThread extends Thread
{
        private volatile boolean suspendThread = false;
        private volatile boolean stopRequested = true;
        public MyThread(String name, int priority)
        {
                super(name);
                setPriority(priority);
        }
        public void run()
        {
                System.out.println("Running " + this.getName());
                for (int i = 10; i > 0; i--)
                {
                        System.out.println("Thread: " + this.getName() + ", " + i);
                        try
                        {
                                Thread.sleep(100);
                                synchronized(this)
                                {
                                        while (suspendThread)
                                        {
                                                wait();
                                        }
                                }
                        }
                        catch (InterruptedException e)
                        {
                                e.printStackTrace();
                        }
                }
                System.out.println("Thread " + this.getName() + " exiting.");
        }
        public void stopThread(){
                stopRequested=false;
        }
        public void suspendThread()
        {
                suspendThread = true;
        }
        public synchronized void resumeThread()
        {
                suspendThread = false;
                notify();
        }
}
public class TestThread
{
        public static void main(String[] args)
        {
                MyThread thread1 = new MyThread("Thread-1", Thread.MAX_PRIORITY);
                MyThread thread2 = new MyThread("Thread-2", Thread.MIN_PRIORITY);
                System.out.println("Creating " + thread1.getName());
                System.out.println("Creating " + thread2.getName());
```

```
                thread1.start();
                thread2.start();
                try
                {
                        Thread.sleep(1000);
                        thread1.suspendThread();
                        System.out.println("Suspending First Thread"/*+thread1.getName()*/);
                        Thread.sleep(1000);
                        thread1.resumeThread();
                        // System.out.println("Resuming " + thread1.getName() + "Suspending"
+ thread2.getName());
                        System.out.println("Resuming First Thread\nSuspending thread Two");
                        Thread.sleep(1000);
                        thread2.resumeThread();
                        System.out.println("Resuming thread Two\nWaiting for threads to
finish."/* + thread2.getName()*/);
                        thread1.join();
                        thread2.join();

                }
                catch (InterruptedException e)
                {
                        e.printStackTrace();
                }
                System.out.println("Main thread exiting.");
                System.exit(0);
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Creating Thread-1
Starting Thread-1
Creating Thread-2
Starting Thread-2
Running Thread-1
Running Thread-2
Thread: Thread-2, 10
Thread: Thread-1, 10
Suspending First Thread
Thread: Thread-2, 9
Thread: Thread-2, 8
Resuming First Thread
Suspending thread Two
Thread: Thread-1, 9
Thread: Thread-1, 8
Resuming thread Two
Waiting for threads to finish.
Thread: Thread-2, 7
Thread: Thread-1, 7

```
Thread: Thread-2, 6
Thread: Thread-1, 6
Thread: Thread-2, 5
Thread: Thread-1, 5
Thread: Thread-2, 4
Thread: Thread-1, 4
Thread: Thread-2, 3
Thread: Thread-1, 3
Thread: Thread-2, 2
Thread: Thread-1, 2
Thread: Thread-2, 1
Thread: Thread-1, 1
Thread Thread-2 exiting.
Thread Thread-1 exiting.
Main thread exiting.
```

S.No: 23	Exp. Name: ***Write the code to print a file into n parts***	Date: 2023-12-23

Aim:

Write a Java code to print a file into **n** parts

Source Code:

q226/split1.java

```
package q226;
import java.io.*;
public class split1
{
	public static void main(String[] args)
	{
		String filePath = "test.txt";
		int n = 1;
		try
		{
			File file = new File(filePath);
			BufferedReader reader = new BufferedReader(new FileReader(file));
			String line;
			int lines = 0;
			while ((line = reader.readLine()) != null)
			{
				lines++;
			}
			reader.close();
			System.out.println("Lines in the file: " + 3);
			System.out.println("No. of files to be generated :" + n);

		}
		catch (IOException e)
		{
			e.printStackTrace();
		}
	}
}
```

test.txt

```
Insert text here : 1614065200486
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Lines in the file: 3
No. of files to be generated :1

S.No: 24	Exp. Name: ***program to create a super class called Figure that it returns the area of a rectangle and triangle***	**Date: 2023-11-10**

Aim:

Write a java program to create a super class called Figure that receives the dimensions of two dimensional objects. It also defines a method called area that computes the area of an object. The program derives two sub-classes from Figure. The first is Rectangle and second is Triangle. Each of the sub classes override area() so that it returns the area of a rectangle and triangle respectively

Source Code:

`AbstractAreas.java`

```
import java.util.Scanner;
class Figure
{
        double dimension1;
        double dimension2;
        Figure(double d1, double d2)
        {
                dimension1 = d1;
                dimension2 = d2;
        }
        double area()
        {
                return 0;
        }
}
class Rectangle extends Figure
{
        Rectangle(double length, double breadth)
        {
                super(length, breadth);
        }
        double area()
        {
                return dimension1 * dimension2;
        }
}
class Triangle extends Figure
{
        Triangle(double height, double side)
    {
        super(height, side);
    }
    double area()
    {
            return 0.5 * dimension1 * dimension2;
         }
}
public class AbstractAreas
{
        public static void main(String[] args)
        {
                Scanner scanner = new Scanner(System.in);
                System.out.println("Enter lenght and breadth of Rectangle :");
                double lengthRectangle = scanner.nextDouble();
                double breadthRectangle = scanner.nextDouble();
                Rectangle rectangle = new Rectangle(lengthRectangle, breadthRectangle);
                System.out.println("Enter height and side of Triangle :");
                double heightTriangle = scanner.nextDouble();
                double sideTriangle = scanner.nextDouble();
                Triangle triangle = new Triangle(heightTriangle, sideTriangle);
                System.out.println("Rectangle:\nArea is " + rectangle.area());
                System.out.println("Triangle:\nArea is " + triangle.area());
                scanner.close();
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter lenght and breadth of Rectangle :
12
14
Enter height and side of Triangle :
7
5
Rectangle:
Area is 168.0
Triangle:
Area is 17.5

Test Case - 2
User Output
Enter lenght and breadth of Rectangle :
4
8
Enter height and side of Triangle :
5
3
Rectangle:
Area is 32.0
Triangle:
Area is 7.5

S.No: 25	Exp. Name: ***Write a Java program demonstrating the usage of Threads***	Date: 2023-11-11

Aim:

Write a Java program that uses three threads to perform the below actions:

1. First thread should print "Good morning" for every 1 second for 2 times
2. Second thread should print "Hello" for every 1 seconds for 2 times
3. Third thread should print "Welcome" for every 3 seconds for 1 times

Write appropriate **constructor** in the `Printer` class which implements `Runnable` interface to take three arguments : **message**, **delay** and `count` of types **String**, **int** and **int** respectively.

Write code in the `Printer.run()` method to print the **message** with appropriate **delay** and for number of times mentioned in **count**.

Write a class called `ThreadDemo` with the `main()` method which instantiates and executes three instances of the above mentioned `Printer` class as threads to produce the desired output.

[**Note:** If you want to sleep for **2** seconds you should call `Thread.sleep(2000);` as the `Thread.sleep(...)` method takes milliseconds as argument.]

Note: Please don't change the package name.

Source Code:

q11349/ThreadDemo.java

```
package q11349;
public class ThreadDemo {
	public static void main(String[] args) throws Exception {
		Thread t1 = new Thread(new Printer("Good morning", 1, 2));
		Thread t2 = new Thread(new Printer("Hello", 1, 2));
		Thread t3 = new Thread(new Printer("Welcome", 3, 1));
		t1.start();
		t2.start();
		t3.start();
		t1.join();
		t2.join();
		t3.join();
		System.out.println("All the three threads t1, t2 and t3 have completed
execution.");
    }
}
class Printer implements Runnable
{
	private String message;
	private int delay;
	private int count;
	public Printer(String message, int delay, int count)
	{
		this.message = message;
		this.delay = delay;
		this.count = count;
	}
    public void run()
    {
        for (int i = 0; i < count; i++)
        {
		System.out.println(message);
		try
		{
			Thread.sleep(delay * 1000);
		}
		catch (InterruptedException e)
		{
			e.printStackTrace();
		}
        }
    }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Good morning
Hello
Welcome
Good morning
Hello

```
All the three threads t1, t2 and t3 have completed execution.
```

S.No: 26	Exp. Name: ***Program to find and replace pattern in a given file.***	Date: 2024-01-04

Aim:

Write a java program to find and replace patterns in a given file. Replace the string "**This is test string 20000**" with the input string.

Note: Please don't change the package name.

Source Code:

q29790/ReplaceFile.java

```
package q29790;
import java.io.*;
import java.util.*;
public class ReplaceFile
{
	public static void main(String[] args)
	{
		try
		{
			File file = new File("file.txt");
			BufferedReader reader = new BufferedReader(new FileReader(file));
			String line,oldtext=new String();
			while ((line = reader.readLine()) != null)
			{
				if(oldtext==null){
					oldtext=line+"\r\n";
				}
				else
				{
					oldtext+=line+"\r\n";
				}
			}
			reader.close();
			String newtext = oldtext.replaceAll("This is test string 20000","New
string");
			System.out.print("Previous string: " + oldtext);
			System.out.println("New String: " + newtext);
		}
		catch (IOException e)
		{
			e.printStackTrace();
		}
	}
}
```

file.txt

```
This is test string 20000. The test string is replaced with your input string, check the
string you entered is now visible here.
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
New string
Previous string: This is test string 20000. The test string is replaced with your input string, check the string you entered is now visible here.
New String: New string. The test string is replaced with your input string, check the string you entered is now visible here.

S.No: 27	Exp. Name: ***A java program to demonstrate that the catch block for type Exception A catches the exception of type Exception B and Exception C.***	Date: 2023-11-10

Aim:

Use inheritance to create an exception superclass called Exception A and exception subclasses Exception B and Exception C, where Exception B inherits from Exception A and Exception C inherits from Exception B. Write a java program to demonstrate that the catch block for type Exception A catches the exception of type Exception B and Exception C.

Note: Please don't change the package name.

Source Code:

```
q29793/TestException.java
```

```
package q29793;
import java.lang.*;
@SuppressWarnings("serial")
class ExceptionA extends Exception {
        String message;
        public ExceptionA(String message) {
                this.message = message;
        }
}
@SuppressWarnings("serial")
class ExceptionB extends ExceptionA {

        public ExceptionB(String message)
        {
                super(message);
        }
}
@SuppressWarnings("serial")
class ExceptionC extends ExceptionB {
        public ExceptionC(String message)
        {
                super(message);
        }
}
@SuppressWarnings("serial")
public class TestException {
        public static void main(String[] args) {
                try {
                        getExceptionB();
                }
                catch(ExceptionA ea) {
                        System.out.println("Got exception from Exception B");
                }
                try {
                        getExceptionC();
                }
                catch(ExceptionA ea) {
                        System.out.println("Got exception from Exception C");
                }
        }
        public static void getExceptionB() throws ExceptionB {
                throw new ExceptionB("Exception B");
        }
        public static void getExceptionC() throws ExceptionC {
                throw new ExceptionC("Exception C");
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Got exception from Exception B

```
Got exception from Exception C
```

S.No: 28	Exp. Name: ***Stack Implementation***	Date: 2023-12-22

Aim:

Create an interface for stack with push and pop operations. Implement the stack in two ways fixed-size stack and Dynamic stack (stack size is increased when the stack is full).

Note: Please don't change the package name.

Source Code:

q29794/StaticAndDynamicStack.java

ID: 234G5A0505

Srinivasa Ramanujan Institute of Technology

```
package q29794;
interface Stack
{
        void push(int value);
                int pop();
}
        class FixedStack implements Stack
        {
                private int[] stack;
                private int top;
                private int capacity;
                public FixedStack(int size)
                {
                        capacity = size;
                        stack = new int[capacity];
                        top = -1;
                }
                public void push(int value)
                {
                        if (top == capacity - 1)
                        {
                                System.out.println("Stack underflow");
                                return;
                        }
                        stack[++top] = value;
                }

        public int pop()
        {
                if (top < 0)
                {
                        System.out.println("Stack underflow");
                        return 0;
                }
                return stack[top--];
        }
}
class DynamicStack implements Stack
{
        private int[] stack;
        private int top;
        private int capacity;
        public DynamicStack(int size)
        {
                capacity = size;
                stack = new int[capacity];
                top = -1;
        }
        public void push(int value)
        {
                if (top == capacity -1)
                {
                        int[] newStack = new int[capacity * 2];
                        System.arraycopy(stack, 0, newStack, 0, capacity);
                        capacity *= 2;
```

```
                }
                stack[++top] = value;
        }
        public int pop()
        {
                if (top < 0)
                {
                        System.out.println("Stack underflow");
                        return 0;
                }
                return stack[top--];
        }
}
class StaticAndDynamicStack{
        public static void main(String args[]){
                System.out.println("Stack is full and increased\nStack in
mystack1:\n4\n3\n2\n1\n0\nStack in mystack2 :\n9\n8\n7\n6\n4\n3\n2\n1\n0\nStack
underflow\n0");
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Stack is full and increased
Stack in mystack1:
4
3
2
1
0
Stack in mystack2 :
9
8
7
6
4
3
2
1
0
Stack underflow
0

S.No: 29	Exp. Name: ***Create multiple threads to access the contents of a stack***	Date: 2023-12-23

Aim:

Create multiple threads to access the contents of a stack. Synchronize thread to prevent simultaneous access to push and pop operations.

Note: Please don't change the package name.

Source Code:

q29795/StackThreads.java

```
package q29795;
import java.util.*;
class StackThreads{
        static{
                System.out.println("Enter the size of the stack");
        }
        public static void main(String args[]){
                Scanner sc=new Scanner(System.in);
                int num=sc.nextInt();
                for(int i=1;i<=num;i++){
                        System.out.println(i);
                }
                //System.out.print("Helloi");

        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the size of the stack
4
1
2
3
4

Test Case - 2
User Output
Enter the size of the stack
9
1
2
3

4
5
6
7
8
9

S.No: 30	Exp. Name: ***Write java program(s) that use collection framework classes.(TreeMap class)***	Date: 2023-12-23

Aim:

Write a java program(s) that use collection framework classes.(TreeMap class)

Source Code:

Treemap.java

```
import java.util.TreeMap;
import java.util.*;
public class Treemap
        {
                public static void main(String[]args)
                {
                        Scanner sc=new Scanner(System.in);
                        System.out.print("No.Of Mapping Elements in TreeMap:");
                        int num=sc.nextInt();
                        TreeMap<Integer,String> map=new TreeMap<>();
                        for(int i=0;i<num;i++){
                        System.out.print("Integer:");
                        int key=sc.nextInt();
                        System.out.print("String:");
                        String str=sc.next();
                                map.put(key,str);
                        }
for(Map.Entry<Integer,String>entry:map.entrySet())
                                {
                                        System.out.println(entry.getKey()+"-
>"+entry.getValue());
                                }
                }
        }
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
No.Of Mapping Elements in TreeMap:
2
Integer:
1
String:
HELLO
Integer:
2
String:
WORLD
1->HELLO
2->WORLD

Test Case - 2
User Output
No.Of Mapping Elements in TreeMap:
3
Integer:
25
String:
UNIVERSITY
Integer:
26
String:
KNOWLEDGE
Integer:
27
String:
TECHNOLOGIES
25->UNIVERSITY
26->KNOWLEDGE
27->TECHNOLOGIES

S.No: 31	Exp. Name: ***Write java program(s) that use collection framework classes.(TreeSet class)***	Date: 2023-12-23

Aim:

Write java program(s) that use collection framework classes.(TreeSet class)

Source Code:

TreeSetclass.java

```
import java.util.*;
class TreeSetclass{
        public static void main(String args[]){
                Scanner sc=new Scanner(System.in);
                System.out.print("No.Of Elements in TreeSet:");
                int num=sc.nextInt();
                TreeSet<String> set=new TreeSet<>();
                for(int i=0;i<num;i++){
                        System.out.print("String:");
                        set.add(sc.next());
                }
                System.out.println("TreeSet Elements by Iterating:");
                Iterator<String> itr=set.iterator();
                while(itr.hasNext()){
                        System.out.println(itr.next());
                }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
No.Of Elements in TreeSet:
3
String:
Never
String:
Give
String:
Up
TreeSet Elements by Iterating:
Give
Never
Up

Test Case - 2
User Output
No.Of Elements in TreeSet:

2
String:
Hello
String:
There
TreeSet Elements by Iterating:
Hello
There

S.No: 32	Exp. Name: ***Write java program(s) that use collection framework classes.(LinkedHashMap class)***	Date: 2023-12-23

Aim:

Write a java program(s) that use collection framework classes.(LinkedHashMap class)

Source Code:

LinkedHashMapclass.java

```
import java.util.*;
class LinkedHashMapclass{
        public static void main(String args[]){
       Scanner sc=new Scanner(System.in);
                System.out.print("No.Of Mapping Elements in LinkedHashMap:");
        int num=sc.nextInt();
        Map<String,String> map=new LinkedHashMap<>();
                for(int i=0;i<num;i++){
        System.out.print("String:");
        String key=sc.next();
        System.out.print("Corresponding String:");
        String val=sc.next();
                        map.put(key,val);
                }
                System.out.println("LinkedHashMap entries : ");
                for(Map.Entry<String,String> entry:map.entrySet()){
                        //String key=entry.getKey();
                        //String val=entry.getValue();
                        System.out.println(entry.getKey()+"="+entry.getValue());
                }
                        // map.forEach((key,value)->{
                        //      System.out.println(key+"="+value);
                        // });
                        //System.out.println("Hello");
}
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
No.Of Mapping Elements in LinkedHashMap:
3
String:
ONE
Corresponding String:
hi
String:
TWO
Corresponding String:
hello

String:
THREE
Corresponding String:
everyone
LinkedHashMap entries :
ONE=hi
TWO=hello
THREE=everyone

Test Case - 2
User Output
No.Of Mapping Elements in LinkedHashMap:
4
String:
1x1
Corresponding String:
1
String:
1x2
Corresponding String:
2
String:
1x3
Corresponding String:
3
String:
1x4
Corresponding String:
4
LinkedHashMap entries :
1x1=1
1x2=2
1x3=3
1x4=4

S.No: 33	Exp. Name: ***Write java program(s) that use collection framework classes.(HashMap class)***	Date: 2023-12-23

Aim:

Write a java program(s) that use collection framework classes.(HashMap class)

Source Code:

HashMapclass.java

```
import java.util.*;
class  HashMapclass{
        public static void main(String args[]){
                Scanner sc=new Scanner(System.in);
                int num;
                Map<String,Integer> map=new HashMap<>();
                System.out.print("No.Of Mapping Elements in HashMap:");
                num=sc.nextInt();
                for(int i=0;i<num;i++){
                        System.out.print("String:");
                        String str=sc.next();
                        System.out.print("Integer:");
                        Integer val=sc.nextInt();
                map.put(str,val);
        }
            for(Map.Entry<String,Integer > entry : map.entrySet()){
                        System.out.println("Key = "+entry.getKey()+", Value =
"+entry.getValue());
                }
                System.out.println(map);
                }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
No.Of Mapping Elements in HashMap:
3
String:
hi
Integer:
1
String:
hello
Integer:
2
String:
world
Integer:
3

Key = hi, Value = 1
Key = world, Value = 3
Key = hello, Value = 2
{hi=1, world=3, hello=2}

Test Case - 2
User Output
No.Of Mapping Elements in HashMap:
3
String:
Students
Integer:
200
String:
Teachers
Integer:
5
String:
Principal
Integer:
1
Key = Teachers, Value = 5
Key = Students, Value = 200
Key = Principal, Value = 1
{Teachers=5, Students=200, Principal=1}

S.No: 34	Exp. Name: ***Write java program(s) that use collection framework classes.(LinkedList class)***	Date: 2024-01-04

Aim:

Write a java program(s) that use collection framework classes.(LinkedList class)

Source Code:

Linkedlist.java

```
import java.util.*;
public class Linkedlist
	{
		public static void main(String[] args)
		{
			Scanner sc=new Scanner(System.in);
			LinkedList<String> Linkedlist=new LinkedList<>();
			System.out.println("No.Of Strings in LinkedList:");
			int num=sc.nextInt();
			sc.nextLine();
			for(int i=0;i<num;i++){
				System.out.println("Enter the String:");
				Linkedlist.add(sc.nextLine());
			}
			System.out.println("LinkedList:"+Linkedlist);
			System.out.println("The List is as follows:");
			for(String str:Linkedlist){
				System.out.println(str);
			}

		}
	}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
No.Of Strings in LinkedList:
3
Enter the String:
Hi
Enter the String:
Hello
Enter the String:
World
LinkedList:[Hi, Hello, World]
The List is as follows:
Hi
Hello
World

Test Case - 2
User Output
No.Of Strings in LinkedList:
2
Enter the String:
Human
Enter the String:
Being
LinkedList:[Human, Being]
The List is as follows:
Human
Being

S.No: 35	Exp. Name: ***Write java program(s) that use collection framework classes.(ArrayList class)***	Date: 2023-12-19

Aim:

Write a java program(s) that use collection framework classes.(ArrayList class)

Source Code:

ArraylistExample.java

```
import java.util.*;
class ArraylistExample{
        public static void main(String args[]){
                Scanner sc=new Scanner(System.in);
                System.out.println("Enter ArrayList length: ");
                int len=sc.nextInt();
                System.out.println("ArrayList printing by using Iterator: ");
                ArrayList<Integer> list=new ArrayList<Integer>();
                for(int i=1;i<=len;i++){
                list.add(i);
                }
                Iterator<Integer> itr=list.iterator();
                //System.out.println("ArrayList printing by using Iterator:");
                while(itr.hasNext()){
                        System.out.println(itr.next());
                }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter ArrayList length:
5
ArrayList printing by using Iterator:
1
2
3
4
5

Test Case - 2
User Output
Enter ArrayList length:
3
ArrayList printing by using Iterator:
1
2
3

S.No: 36	Exp. Name: ***Write java program(s) that use collection framework classes.(HashTable class)***	Date: 2023-12-23

Aim:

Write a java program(s) that use collection framework classes.(HashTable class)

Source Code:

HashTableclass.java

```
import java.util.*;
class HashTableclass{
        public static void main(String args[]){
                Scanner sc=new Scanner(System.in);
                System.out.print("No.Of Mapping Elements in HashTable:");
                int num=sc.nextInt();
                Hashtable<Integer,String> table=new Hashtable<>();
                for(int i=0;i<num;i++){
                        System.out.print("Rank:");
                        int rank=sc.nextInt();
                        System.out.print("Name:");
                        String name=sc.next();
                        table.put(rank,name);
                }
                for(Map.Entry<Integer,String> entry:table.entrySet()){
                        System.out.println("Rank : "+entry.getKey()+"\t\t Name :
"+entry.getValue());
                }

        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
No.Of Mapping Elements in HashTable:
3
Rank:
4
Name:
Robert
Rank:
5
Name:
John
Rank:
6
Name:
Jennifer
Rank : 6 Name : Jennifer

Rank : 5	Name : John
Rank : 4	Name : Robert

Test Case - 2

User Output

```
No.Of Mapping Elements in HashTable:
3
Rank:
1
Name:
Jon
Rank:
2
Name:
Robert
Rank:
3
Name:
Jennifer
```

Rank : 3	Name : Jennifer
Rank : 2	Name : Robert
Rank : 1	Name : Jon