**Problem Statement 1: Multithreading in Java**

**Solve following sub problems,**

**1. Implement three classes: Storage, Counter and Printer as follows,**

**a. The Storage class should store an integer.**

**b. The Counter class should create a thread and starts counting from 0 (0,1,2,3…) and stores each value in the Storage class.**

**c. The Printer Class should create a thread that keeps reading the value in the**

**Storage class and printing it.**

**d. Write a program that creates an instance of the Storage class and set up a**

**Counter and Printer object to operate on it.**

**e. Identify that, whether synchronization is required or not in this problem. If yes, implementit.**

**package** day11assignment;

**class** Storage {

**private** **int** value;

**public** **synchronized** **void** setValue(**int** value) {

**this**.value = value;

}

**public** **synchronized** **int** getValue() {

**return** value;

}

}

**class** Counter **extends** Thread {

**private** Storage storage;

**public** Counter(Storage storage) {

**this**.storage = storage;

}

@Override

**public** **void** run() {

**for** (**int** i = 0; i <= 10; i++) {

storage.setValue(i);

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

Thread.*currentThread*().interrupt();

}

}

}

}

**class** Printer **extends** Thread {

**private** Storage storage;

**public** Printer(Storage storage) {

**this**.storage = storage;

}

@Override

**public** **void** run() {

**int** lastPrinted = -1;

**while** (lastPrinted < 10) {

**int** value = storage.getValue();

**if** (value != lastPrinted) {

System.***out***.println("Printed: " + value);

lastPrinted = value;

}

**try** {

Thread.*sleep*(500);

} **catch** (InterruptedException e) {

Thread.*currentThread*().interrupt();

}

}

}

}

**public** **class** Main {

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

Storage storage = **new** Storage();

Counter counter = **new** Counter(storage);

Printer printer = **new** Printer(storage);

counter.start();

printer.start();

}

}

**Output:**

Printed: 0

Printed: 1

Printed: 2

Printed: 3

Printed: 4

Printed: 5

Printed: 6

Printed: 7

Printed: 8

Printed: 9

Printed: 10

**2. You are given a task to write a program that prints numbers from 1 to 10 using three different threads. Each thread should print numbers in sequential order by adding suitable synchronization.**

**package** day11assignment;

**public** **class** Main1 {

**private** **static** **final** Object ***lock*** = **new** Object();

**private** **static** **int** *number* = 1;

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

Thread thread1 = **new** Thread(**new** Printer(1, 3));

Thread thread2 = **new** Thread(**new** Printer(4, 6));

Thread thread3 = **new** Thread(**new** Printer(7, 10));

thread1.start();

thread2.start();

thread3.start();

}

**static** **class** Printer **implements** Runnable {

**private** **int** start;

**private** **int** end;

**public** Printer(**int** start, **int** end) {

**this**.start = start;

**this**.end = end;

}

@Override

**public** **void** run() {

**synchronized** (***lock***) {

**for** (**int** i = start; i <= end; i++) {

**while** (i != *number*) {

**try** {

***lock***.wait();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

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

*number*++;

***lock***.notifyAll();

}

}

}

}

}

**Output:**

1

2

3

4

5

6

7

8

9

10

**3. Write a Java application that will accept two filenames (text files) as command line arguments and use two threads to read contents from the two text files. Each of the threads should sleep for a random time after displaying filename with each line.**

**Perform following steps in the program,**

**a. Create and start two threads. One thread will read a text file (number.txt)**

**containing five positive integers one on each line.**

**b. The second threadshould calculate factorialof the number readby the first**

**threadand print the message on the screen as “Factorial of x is y” ,here x is**

**number & y is factorial of the number.**

**c. The two threads should work in synchronization. Handle all necessary exceptions**

**package** day11assignment;

import java.io.\*;

import java.util.Random;

public class FactorialApp {

    static class FileReaderThread extends Thread {

        private String fileName;

        private Storage storage;

        public FileReaderThread(String fileName, Storage storage) {

            this.fileName = fileName;

            this.storage = storage;

        }

        @Override

        public void run() {

            try (BufferedReader br = new BufferedReader(new FileReader(fileName))) {

                String line;

                while ((line = br.readLine()) != null) {

                    int number = Integer.*parseInt*(line.trim());

                    storage.setNumber(number);

                    synchronized (storage) {

                        storage.notify(); // Notify the other thread to calculate factorial

                    }

                    Thread.*sleep*(new Random().nextInt(500));

                }

            } catch (IOException | InterruptedException e) {

                e.printStackTrace();

            }

        }

    }

    static class FactorialThread extends Thread {

        private Storage storage;

        public FactorialThread(Storage storage) {

            this.storage = storage;

        }

        @Override

        public void run() {

            try {

                while (true) {

                    synchronized (storage) {

                        storage.wait();

                    }

                    int number = storage.getNumber();

                    long factorial = calculateFactorial(number);

                    System.*out*.println("Factorial of " + number + " is " + factorial);

                    Thread.*sleep*(new Random().nextInt(500));

                }

            } catch (InterruptedException e) {

                e.printStackTrace();

            }

        }

        private long calculateFactorial(int num) {

            long factorial = 1;

            for (int i = 1; i <= num; i++) {

                factorial \*= i;

            }

            return factorial;

        }

    }

    static class Storage {

        private int number;

        public int getNumber() {

            return number;

        }

        public void setNumber(int number) {

            this.number = number;

        }

    }

    public static void main(String[] args) {

        if (args.length != 2) {

            System.*out*.println("Please provide two filenames as arguments.");

            return;

        }

        Storage storage = new Storage();

        FileReaderThread readerThread = new FileReaderThread(args[0], storage);

        FactorialThread factorialThread = new FactorialThread(storage);

        readerThread.start();

        factorialThread.start();

        try {

            readerThread.join();

            factorialThread.interrupt(); // Interrupt factorial thread once reading is done

        } catch (InterruptedException e) {

            e.printStackTrace();

        }

    }

}