**WEEK 9**

**a) Write a program to simulate a buzzer. The program should make use of the Thread class. Create a class named Buzzer that extends the Thread class. Create an interface named MonitorTime which contains a method setBuzzerTime() . Your Buzzer class should implement this interface. Override setBuzzerTime() method to set the buzzer time delay in milliseconds and the number of times the buzzer should be repeated. Also, include methodsblowBuzzer to start the buzzer.**

import java.util.Scanner;

interface MonitorTime{

void setBuzzerTime(int time);

}

public class Buzzer extends Thread implements MonitorTime {

@Override

public void setBuzzerTime(int time) {

try {

Thread.sleep(time\* 1000L);

}

catch (InterruptedException e) {

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

}

}

Buzzer( int no ) {

System.out.println( no + " Buzzer started...");

this.start();

}

static void methodsblowBuzzer (int n, int time ){

System.out.println("Buzzer started...");

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

Buzzer buzz = new Buzzer( i );

buzz.setBuzzerTime(time);

}

}

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.print("Enter number of times buzzer should be repeated : ");

int n = in.nextInt();

System.out.print("Set Buzzer time delay in seconds : ");

int time = in.nextInt();

methodsblowBuzzer( n, time);

}

}

**b.)Write a java program which generates a set of random numbers and then determines negative,positive even,positive odd numbers concurrently**

import java.util.Random;

public class Main {

public static void main(String[] args) {

int[] numbers = new int[10];

Random random = new Random();

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

numbers[i] = random.nextInt(100) - 50;

System.out.println("Generated number: " + numbers[i]);

}

Thread negativeThread = new Thread(() -> {

for (int number : numbers) {

if (number < 0) {

System.out.println(number + " is negative");

}

}

});

Thread positiveEvenThread = new Thread(() -> {

for (int number : numbers) {

if (number > 0 && number % 2 == 0) {

System.out.println(number + " is positive even");

}

}

});

Thread positiveOddThread = new Thread(() -> {

for (int number : numbers) {

if (number > 0 && number % 2 != 0) {

System.out.println(number + " is positive odd");

}

}

});

negativeThread.start();

positiveEvenThread.start();

positiveOddThread.start();

}

}

**WEEK10**

**a.Write a program that will count the total occurrences of the number ‘10’ in a text file of strings representing numbers of type int and will show the value of the count on the screen once the whole file is read. The file contains the following numbers separated by space. 104781034111015610**

import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;

public class Main {

public static void main(String[] args) throws FileNotFoundException {

File file = new File("abc.txt");

Scanner scanner = new Scanner(file);

int count = 0;

while (scanner.hasNext()) {

if (scanner.nextInt() == 10) {

count++;

}

}

System.out.println("Total occurrences of 10: " + count);

}

}

b.**Write a program that reads grades of type double of eight students that the user provides. The grades lie between 0 and 10. These grades should be written to a binary file and read from it. The program outputs the highest and lowest grades achieved by students on the screen. The file contains nothing but numbers of type double written to the file with *writeDouble.***

import java.io.\*;

import java.util.Scanner;

public class Main2 {

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

Scanner scanner = new Scanner(System.in);

// Get grades from user

double[] grades = new double[8];

for (int i = 0; i < 8; i++) {

System.out.print("Enter grade for student " + (i + 1) + ": ");

grades[i] = scanner.nextDouble();

}

// Write grades to binary file

DataOutputStream output = new DataOutputStream(new FileOutputStream("grades.txt"));

for (double grade : grades) {

output.writeDouble(grade);

}

output.close();

// Read grades from binary file

DataInputStream input = new DataInputStream(new FileInputStream("grades.txt"));

double highestGrade = Double.MIN\_VALUE;

double lowestGrade = Double.MAX\_VALUE;

while (input.available() > 0) {

double grade = input.readDouble();

if (grade > highestGrade) {

highestGrade = grade;

}

if (grade < lowestGrade) {

lowestGrade = grade;

}

}

input.close();

// Output highest and lowest grades

System.out.println("Highest grade: " + highestGrade);

System.out.println("Lowest grade: " + lowestGrade);

}

}