KIBABII UNIVERSITY

SCHOOL OF COMPUTING AND INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE

CSC 210: OBJECT ORIENTED PROGRAMMING II TIME:1HR

**Instruction**

Answer ALL questions

1. Write a Java statement that creates an output stream to a binary file named statistics.dat. [2 Marks]

*DataOutputStream outputStream = new DataOutputStream(new FileOutputStream("statistics.dat"));*

1. Use the output stream created in number a) above to write the String BBC to the file named statistics.dat.

[1 Mark]

*FileOutputStream outputStream = new FileOutputStream("statistics.dat");*

*byte[] bytes = "BBC".getBytes();*

*outputStream.write(bytes);*

*outputStream.close();*

1. Write a Java statement to create and open an output stream to a text file named

autos.txt. [2 Marks]

*FileWriter writer = new FileWriter("autos.txt");*

*or*

*BufferedWriter writer = new BufferedWriter(new FileWriter("autos.txt"));*

1. Write a Java method that returns a String representing a file name entered by the user. Use the bufferedReader class to obtain input.

[2 Marks]

public static String getFileNameFromUser() throws IOException {

BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Enter file name: ");

String fileName = reader.readLine();

reader.close();

return fileName;

}

1. Write a complete Java program using a BufferedReader object that opens a file named

autos.txt and displays each line to the screen. [4 Marks]

*import java.io.BufferedReader;*

*import java.io.FileReader;*

*import java.io.IOException;*

*public class ReadFileLineByLine {*

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

*String fileName = "autos.txt";*

*try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {*

*String line;*

*while ((line = reader.readLine()) != null) {*

*System.out.println(line);*

*}*

*} catch (IOException e) {*

*System.err.println("Error reading file: " + e.getMessage());*

*}*

*}*

*}*

1. Write a complete Java program using a Scanner object that opens a file named autos.txt and displays each line to the screen. [4 Marks]

*import java.io.File;*

*import java.io.FileNotFoundException;*

*import java.util.Scanner;*

*public class ReadFileWithScanner {*

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

*String fileName = "autos.txt";*

*File file = new File(fileName);*

*Scanner scanner = new Scanner(file);*

*while (scanner.hasNextLine()) {*

*String line = scanner.nextLine();*

*System.out.println(line);*

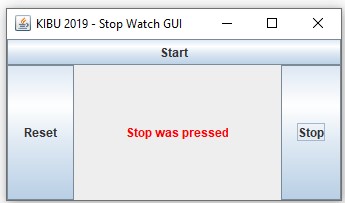
*}*

*scanner.close();*

*}*

*}*

1. Write a java program that will create and implement the following GUI. [8 Marks]



1. Using examples, describe importance of using threads. [3 Marks]

1. Improved Responsiveness for example in a music player application, one thread can play the audio while another thread fetches album information or lyrics, ensuring smooth playback without interruptions.

2. Efficient Resource Utilization as implemented in Download managers which can utilize multiple threads to download files concurrently, effectively maximizing network bandwidth utilization.

3. Enhanced User Experience: In Games, which often use multiple threads to handle various tasks like animation, physics simulation, and user input, creating a smooth and engaging gameplay experience.

4. Background Operations: Multithreaded applications can perform background tasks without interrupting the user's workflow. For example;

Email clients can use a thread to check for new emails in the background without interfering with composing a new message.

Antivirus software can use multiple threads to scan files for threats in the background, ensuring system security without affecting other applications.

1. Using examples, describe two ways of implementing threads in java. [4 Marks]

**1. Extending the Thread Class**

This approach involves creating a class that extends the Thread class and overrides its run() method. The run() method defines the code that the thread will execute concurrently with the main thread.

public class MyThread extends Thread {

*@Override*

*public void run() {*

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

*System.out.println("MyThread: " + i);*

*}*

*}*

*}*

**2. Implementing the Runnable Interface**

This approach involves creating a class that implements the Runnable interface and defines its run() method. The run() method defines the code that the thread will execute.

public class MyRunnable implements Runnable {

*@Override*

*public void run() {*

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

*System.out.println("MyRunnable: " + i);*

*}*

*}*

*}*