**Operating Systems and Concurrency**

**Student Name:** Raphael Salaja

**AIT Student number:** A00269349

**Assignment:** Assignment 3

**Date of Submission:** 20/10/2021

**FORM A1**

# STUDENT PLAGIARISM DISCLAIMER FORM

****

## PLAGIARISM DISCLAIMER

STUDENT NAME: Raphael Salaja

STUDENT NUMBER: A00269349

PROGRAMME: BSc (Hons) in Software Design with Virtual Reality and Gaming

YEAR: 3

MODULE: Operating Systems and Concurrency

LECTURER: Thiago Braga Rodrigues

ASSIGNMENT TITLE: Lab 3 Assignment

DUE DATE: 30 October 2021

DATE SUBMITTED: 20 October 2021

ADDITIONAL INFORMATION:

I understand that plagiarism is a serious academic offence, and that AIT deals with it according to the AIT Policy on Plagiarism.

I have read and understand the AIT Policy on Plagiarism and I agree to the requirements set out therein in relation to plagiarism and referencing. I confirm that I have referenced and acknowledged properly all sources used in preparation of this assignment. I understand that if I plagiarise, or if I assist others in doing so, that I will be subject to investigation as outlined in the AIT Policy on Plagiarism.

I understand and agree that plagiarism detection software may be used on my assignment. I declare that, except where appropriately referenced, this assignment is entirely my own work based on my personal study/or research. I further declare that I have not engaged the services of another to either assist in, or complete this assignment.

## Signed:Raphael Salaja

**Dated: 13 October 2021**

# Contents

[Classes 4](#_Toc85026586)

[1. ATM Simulator Class 4](#_Toc85026587)

[2. Account Class 5](#_Toc85026588)

[Before Synchronization 5](#_Toc85026589)

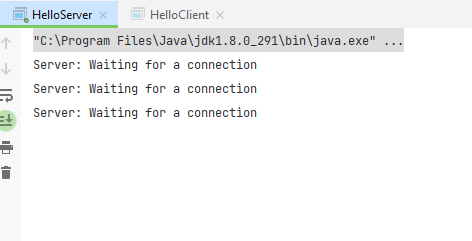
[After Synchronization 6](#_Toc85026590)

[3. Bank Class 7](#_Toc85026591)

[4. Company Class 8](#_Toc85026592)

# Question 1

## Single Threaded File Server



# Question 2

## ThreadedFileServerApp

import java.net.ServerSocket;  
import java.net.Socket;  
  
public class ThreadedFileServerApp {  
 public static void main(String[] args) throws Exception {  
 System.*out*.println("SERVER READY");  
 ServerSocket server\_socket = new ServerSocket(2001);  
 while (true) {  
 Socket socket = server\_socket.accept();  
 Thread t = new WorkerThread(socket);  
 t.start();  
 }  
 }  
}

## WorkerThread

import java.io.FileInputStream;  
import java.io.InputStream;  
import java.io.OutputStream;  
import java.io.PrintWriter;  
import java.net.ServerSocket;  
import java.net.Socket;  
import java.util.Scanner;  
  
public class WorkerThread extends Thread {  
 Socket socket;  
  
 WorkerThread(Socket socket) {  
 this.socket = socket;  
 }  
  
 public void run() {  
 try {  
 InputStream in = socket.getInputStream();  
 Scanner r = new Scanner(in);  
 OutputStream o = socket.getOutputStream();  
 PrintWriter p = new PrintWriter(o);  
  
 String file\_name;  
 file\_name = r.nextLine();  
 System.*out*.println(getName() + " HANDLING: " + file\_name);  
  
 FileInputStream f\_in = new FileInputStream(file\_name);  
 Scanner f = new Scanner(f\_in);  
  
 String line;  
 while (f.hasNextLine()) {  
 line = f.nextLine();  
 p.println(line);  
 }  
 *sleep*(1500);  
 p.close();  
 System.*out*.println(getName() + " FINISHED HANDLING: " + file\_name);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

## ClientSide

import java.io.OutputStream;  
import java.io.PrintWriter;  
import java.net.InetAddress;  
import java.net.Socket;  
import java.util.Scanner;  
  
public class ClientSide {  
  
 public static void main(String[] args) throws Exception {  
 InetAddress inet = InetAddress.*getLocalHost*();  
  
 Socket s = new Socket(inet, 2001);  
 InputStream in = s.getInputStream();  
 Scanner r = new Scanner(in);  
 OutputStream o = s.getOutputStream();  
 PrintWriter p = new PrintWriter(o);  
  
 p.println("testing.txt");  
 p.flush();  
  
 String line;  
 while (r.hasNextLine()) {  
 line = r.nextLine();  
 System.*out*.println(line);  
 }  
 }  
}

