FACULTY OF SCIENCE

ACADEMY OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING

MODULE	**COMPUTER SCIENCE 2A** CSC02A2
CAMPUS	AUCKLAND PARK CAMPUS (APK)
FINAL SUMMATIVE ASSESSMENT OPPORTUNITY SSSA	AUGUST
DATE: 2021-06-29	**SESSION:** 08:00 - 10:00
ASSESSOR(S):	**MR. A. MAGANLAL**
	MR. S. SITHUNGU
MODERATOR:	**MR. R. MALULEKA**
DURATION: 120 MINUTES	**MARKS:** 100

Please read the following instructions carefully:

1. You must complete the assessment **by yourself** within the prescribed time limits.
2. No communication concerning the assessment is permissible during the assessment session except with **ACSSE** staff members.
3. You are bound by all university regulations including, but not limited, to assessment, plagiarism, and ethical conduct.
4. You may not directly take any code from any source, including your own previous submissions. All code must be written by yourself during the assessment.
5. If you **do not have access to a computer** then you can do a *pen and paper submission*:
    - Write *cleanly* and *legibly*.
    - Make use of CamScanner app to create a PDF from your written work.
6. **All answers** must be in a *single* **PDF** file. Make sure your details appear at the top of the first page of the PDF file.
7. Complete the **Honesty Declaration: Online Assessment** and upload it as part of your submission. The completed **Honesty Declaration** is required for a submission to be eligible to be marked.
8. Your answers to the question (in a single PDF file) together with the declaration must be submitted in a zip archive named in the following format:
   `SURNAME_INITIALS_STUDENTNUMBER_CSC2A_2021_FSAO_SSSA.zip`
9. Additional time for submission is allowed for as per the posted deadlines on EVE.
10. This paper contains **9** question(s).
11. This paper consists of **5** page(s) excluding the cover page.

QUESTION 1: Java Overview

(a) **Discuss** the *differences* between ***machine code*** and ***assembly Language***. **[04]**

(b) **Compare** the **Java** and C++ programming languages. Your *comparison* must include only the ***differences*** between the languages. **[06]**

Total: 10

QUESTION 2: Elementary Java Programming

(a) **Discuss** when would you use `continue` instead of `break`. **[03]**

(b) **Analyse** the following **Java** code segment and answer the questions that follow: **[06]**

```
1 Object object = null;
2 Object 1stObject = null;
3 Object null = null;
4 Object $Float = new Float(45.0f);
5 Object __ = object;
6 Object false =  Boolean.TRUE;
```

Java has strict rules for identifiers. Some of the identifiers in the code segment above break those rules. **Indicate** the line where the ***problem*** occurs and what the error is on that line.

(c) **Provide** ***Java*** *source code* to reference the ***standard input stream*** of a Java application. **[01]**

Total: 10

QUESTION 3: Text Processing and Persistence

(a) **Can** an `ObjectOutputStream` be used to ***read binary data***? **Provide** a *reason* for your answer. **[03]**

(b) **Provide** the ***canonical (full) name*** of the *interface* which a class must implement in order for that class to be written to file with an `ObjectOutputStream`. **[02]**

(c) **Provide** a *single* ***regular expression*** that matches *all dates* in the following format. **[05]**

- 14th February 2018
- 17th April 2016
- 23rd May 2015
- 02nd January 2019
- 01st March 2021

Total: 10

~~ Assessment continues on the next page. ~~

QUESTION 4: Object Orientation

Analyse the following **Java** code segment and answer the questions that follow:

```
1  public abstract class Database{
2    public abstract void commit(String data);
3    /* Remainder of class omitted */
4  }
5  public class Cloud<T extends DataBase> {
6    private T db;
7    public void save(String data) {
8      db.commit(data);
9    }
10   /* Remainder of Class omitted */
11 }
12 public class DataDerived extends DataBase{
13   public void commit(String data) { /* code omitted */}
14   /* Remainder of class omitted */
15 }
16 public class DBSim {
17   public static void main(String[] args) {
18     Cloud<DataDerived> instance = new Cloud<>();
19     instance.save();
20   }
21 }
```

Identify five (5) unique **Java** ***programming constructs/principles*** in the code above. **List** the line number and the ***programming constructs/principles***, for example: *Line 99 - Exception Handling*.

Total: 10

QUESTION 5: Graphical User Interfaces

Your employer has asked you to help **choose** between Swing and JavaFX as the ***Java GUI framework*** for the company's next project. The company's next project must be able to run on as many devices as possible, however, they also require the use of some older GUI controls. In your answer, you must discuss:

- Can either Swing or JavaFX be used?
- Which parts of Swing are well-suited to what the employer wants.
- Which parts of JavaFX are well-suited to what the employer wants.
- Which ***Java GUI framework*** would you choose?

Remember that this is a 10 mark question so you should make roughly 10 valid statements including which framework you choose.

Total: 10

~~ Assessment continues on the next page. ~~

QUESTION 6: Advanced Java Programming

(a) With respect to ***inner classes*** in `Java`, **provide** *source code* for instantiating a non-static inner class from outside the containing (outer) class. **[04]**

(b) **Define *type erasure***. **[03]**

(c) With regards to ***Java*** *multi-threaded programming*:

i. **Define** a ***task***. **[01]**

ii. **Define** a ***thread***. **[01]**

iii. **Discuss** how these two concepts are ***related***. **[01]**

Total: 10

QUESTION 7: Design Patterns

You have decided to create a blog to raise awareness on good programming practices. You want readers to be able to subscribe to your blog so that they receive notifications whenever you post new content. To achieve this, your blog should be able to maintain a list of subscribers and must be able to add, remove or notify subscribers whenever there is new content. You want to use a design pattern to achieve this. You have reached out to your friends who are also programmers, and they helped you narrow down your options to the following three design patterns. Now you need to select the one that would be the most appropriate for the task at hand.

- Visitor Design Pattern
- Abstract Factory Design Pattern
- Observer Design Pattern

(a) **Which *design pattern*** do you think would best fit the *requirements*? **[02]**

(b) **Provide** reasons for your choice of design pattern. **[04]**

(c) **Discuss** the ***limitations*** that would need to be considered before implementing the design pattern. **[04]**

Total: 10

~~ Assessment continues on the next page. ~~

QUESTION 8: UML

The **Utopian Factory of the Future** is in the process of automating most of its production processes. As part of the team, you are responsible for designing the communication workflow between the devices. There is one **ControllerDevice** responsible for invoking **Command**s to the **ControlledDevice**s involved in the production line. You have decided that the **Command Design Pattern** would be the best approach to follow in successfully implementing what is required. Now you have to provide a UML diagram to your colleagues to give them a clear picture of how you would apply the design pattern to the problem. The classes for the different kinds of objects that need to be supported are shown below in the form of Java code.

```
1  public class ControllerDevice{
2    private Command command;
3    public void executeCommand(){command.execute();}
4  }
5  public class ControlledDevice{
6    private boolean isRunning = false;
7    private int workDone = 0;
8    public void start(){isRunning = true;}
9    public void stop(){isRunning = false;}
10   public void doWork(){workDone++;}
11 }
12 public interface Command{
13   public void execute();
14 }
15 public class StartCommand implements Command{
16   private ControlledDevice device;
17   @Override
18   public void execute(){device.start();}
19 }
20 public class StopCommand implements Command{
21   private ControlledDevice device;
22   @Override
23   public void execute(){device.stop();}
24 }
25 public class DoWorkCommand implements Command{
26   private ControlledDevice device;
27   @Override
28   public void execute(){device.doWork();}
29 }
```

Provide a UML class diagram for the **Command Design Pattern** applied to the problem stated above.

Total: 15

~~ Assessment continues on the next page. ~~

QUESTION 9: Cold code

Provide Java source code for the following problem. You can assume that all relevant packages have been imported.

OurMusic sells **MusicalInstruments** and allows potential buyers to test the instruments inside the store before buying them. A **MusicalInstrument** is either an **AcousticInstrument** or a **DigitalInstrument**. The **DigitalInstruments** must always be connected to power so that potential buyers can switch them on and play them while in the store. This uses a lot of electricity, and the manager wants to know the minimumVoltage consumed by each **DigitalInstrument**. The code below provides more information on how **MusicalInstrument**s are stored in the system:

```
1  public class MusicalInstrument{
2    public int getID() {/* code omitted */}
3    public String getKey() {/* code omitted */}
4    public String getName() {/* code omitted */}
5    /* Contructors, accessor, mutator methods ommitted*/
6  }
7  public class AcousticInstrument extends MusicalInstrument {
8    public String getMaterial() {/* code omitted */}
9    /* Contructors, accessor, mutator methods ommitted*/
10 }
11 public class DigitalInstrument extends MusicalInstrument {
12   public double getMinimumVoltage() {/* code omitted */}
13   /* Contructors, accessor, mutator methods ommitted*/
14 }
```

Create a static ***writeDigitalInstrumentInfo*** method. The method accepts an **ArrayList** of **MusicalInstruments** and a **File** handle as parameters. The method will write the ID, name and minimumVoltage of each **DigitalInstrument** to the text file using **Automatic Resource Management**.

Total: 15

~~ THE END ~~