Faculty of Science

Course:	CSCI 2020U – Software Systems Development & Integration
Due date:	March 16th, 2021
Grade:	30%
Submission:	via Canvas, **pdf (Part I) and zip (Part II) files** – paste short questions' answers into this document and save it as PDF;

General Instructions

- This is an **individual** submission.
- You are **allowed to ask the instructor or the TAs** clarification questions via **private messages on MS Teams**. However, no answers that will directly solve your midterm will be provided.
- You should **not** post questions using **public channels**, neither in official nor unofficial course platforms.
- You must **not** share your questions nor your answers to this assignment.
- Any form of academic misconduct detected will be treated accordingly to the university's procedures.
- You must name your midterm **PDF** file "csci2020umidterm-lastname-firstname-studentID"
- Preferably, use a different text color for your textual answers.

Student Name: Noshen Atashe

Student ID: 100620403

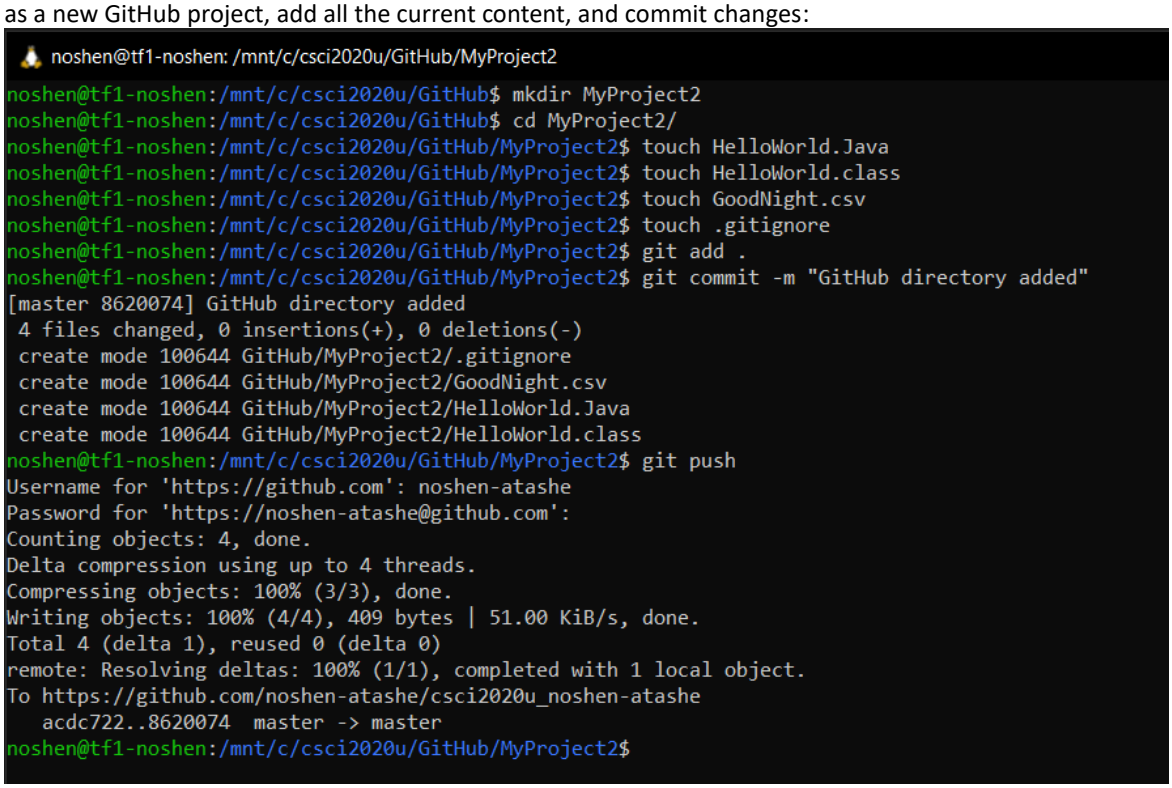
Part I: Short Answer Questions (10 pts)

1. (1.5 pts) Given the directory and files below:
   - GitHub
     - MyProject 1
     - MyProject 2
       - HelloWorld.java
       - HelloWorld.class
       - GoodNight.csv
       - .gitignore

   a. Write the content of the .gitignore file for **MyProject 2** to ignore compiled files.

```
#compiled class file
*.class
```

   b. Assume you opened a terminal in the GitHub folder, write the command(s) you would use to start **MyProject 2** as a new GitHub project, add all the current content, and commit changes:



2. (1.5 pts) Write a build.gradle file's content to run a Java program **CalculateTaxes.java**, which has an external dependency on 'org.json'.

```
apply plugin: 'java'

repositories{
  mavenCentral();
}
dependencies {
  compile group: 'org.json',
      name: 'json',
      version: ' 20180813'
}
```

3. (2 pts) Imagine you are a senior software developer in a company and you are asked to write a brief introduction to the company's best coding practices to newly hired coop students. Your introduction should:

- Explain what are coding best practices.
- Provide 2-3 examples based on Java as the programming language.
- Explain why it is so important that everyone on-board learns and applies these practices.

Write your introduction below.

It is important to maintain best coding practices for avoiding errors and making it easier to understand for everybody in the team. When projects get large, it can be difficult to track an error and having best coding practices can make it easier to debug. Here are some general rules for best practices:

Rule	Bad practice	Best practice

Use Yoda condition	`if (account = null) {`
	`  …`
	`}`
		`if (null = account) {`
		`  …`
		`}`

Always use {} for conditionals	`if (account.isActive())`
	`  txMgr.record(account, amt);`
	`  account.transfer(amt);`
		`if (account.isActive()) {`
		`  txMgr.record(account, amt);`
		`  account.transfer(amt);`
		`}`

Always check for null	`DBConnection conn = connectToDB();`
	`conn.performTransaction(tx);`
		`DBConnection conn = connectToDB();`
		`if (null != conn) {`
		`  conn.performTransaction(tx);`
		`}`

Always check for null first	`DBConnection conn = connectToDB();`
	`if (conn.isActive() && null != conn) {`
	`  conn.performTransaction(tx);`
	`}`
		`DBConnection conn = connectToDB();`
		`if (null != conn && conn.isActive()) {`
		`  conn.performTransaction(tx);`
		`}`

Always use "finally" block to handle all clean up [try →catch→finally]	`DBConnection conn = connectToDB();`
	`try {`
	`  conn.performTransaction(tx);`
	`  conn.disconnect();`
	`} catch (TransactionException e) {`
	`  e.printStackTrace();`
	`}`
		`DBConnection conn = connectToDB();`
		`try {`
		`  conn.performTransaction(tx);`
		`} catch (TransactionException e) {`
		`  e.printStackTrace();`
		`} finally {`
		`  conn.disconnect();`
		`}`

Some other rules:

- Variable names should be nouns that are descriptive of their purpose.
- Single character variables used for loop counters.
- Function names should be verbs that are descriptive of what they will do.
- Always document your codes. No need to over describe.

4. (2 pts) Using your own words, contrast creating UI using FXML with programmatically method. Are there any conceptual advantages, or disadvantages between the two when it comes to software architecture?

Creating UI programmatically method can give the programmer a greater control over the source code. This requires a thorough understanding of all parts and more time to implement. However, using FXML can allow you to create projects faster as it has a more familiar approach for web developers. Using FXML allows developer to easily maintain more complex interfaces.

5. (3 pts) Considering the hierarchy Ontario/Region/City is used to report on daily new covid-19 cases. Give an example of how that data would be represented in the different formats:
***Note**. You can create random data for your example data, it also only requires 3-5 records to display the structure, you do not need to come up with a large amount of data.*

a. CSV
b. JSON
c. XML

Data used for the examples:

province	city	#new cases
ontario	Toronto	3710
ontario	Ottawa	180
ontario	Waterloo	2734
ontario	Peterborough	120
ontario	Niagara region	1900
ontario	York region	1063

CSV format:

```
province,city,#new cases
ontario,Toronto,3710
ontario,Ottawa,180
ontario,Waterloo,2734
ontario,Peterborough,120
ontario,Niagara region,1900
ontario,York region,1063
```

JSON format:

```
[
 {
  "province": "ontario",
  "city": "Toronto",
  "#new cases": 3710
 },
 {
  "province": "ontario",
  "city": "Ottawa",
  "#new cases": 180
 },
 {
  "province": "ontario",
  "city": "Waterloo",
  "#new cases": 2734
 },
 {
  "province": "ontario",
  "city": "Peterborough",
  "#new cases": 120
 },
 {
  "province": "ontario",
  "city": "Niagara region",
  "#new cases": 1900
 },
 {
  "province": "ontario",
  "city": "York region",
  "#new cases": 1063
 }
]
```

XML format:

```
<?xml version="1.0" encoding="UTF-8"?>
<database>
  <province id="province">
    <city-name>city</city-name>
    <cases>#new cases</cases>
  </province>


  <province id="ontario">
    <city-name>Toronto</city-name>
    <cases>3710</cases>
  </province>


  <province id="ontario">
    <city-name>Ottawa</city-name>
    <cases>180</cases>
  </province>

  <province id="ontario">
    <city-name>Waterloo</city-name>
    <cases>2734</cases>
  </province>

  <province id="ontario">
    <city-name>Peterborough</city-name>
    <cases>120</cases>
  </province>

  <province id="ontario">
    <city-name>Niagara region</city-name>
    <cases>1900</cases>
  </province>

  <province id="ontario">
    <city-name>York region</city-name>
    <cases>1063</cases>
  </province>
</database>
```

Part II: Programming Question (20 pts)

For this part, it's recommended you use the **CSCI2020U-Midterm-Basecode.zip** given to you as base template for a JavaFX application. You can choose your environment of choice for JavaFX, however, the base code is an IntelliJ project which you can adapt to your setup.

You will implement the necessary code to fulfill the action of each of the buttons shown in Figure 1. You are given specific instructions for each of the buttons (Animation, 2D Graphics, About). However, the **general instructions** are

1. You may change the Scene, or the root object of the main scene (as you prefer) in order to change the application UI for each of the buttons accordingly.
2. You main create as many (if any) additional classes as needed.
3. You must keep coding best practices while you design your solution.
4. You may create the UI components programmatically, or via FXML/CSS according to your preference.
5. All 3 buttons different UI must contain a "Back to Main" link, which if clicked on, will allow the user to navigate back to the original mainScene as you see it on the screenshot in Figure 1.

```
//        setting the Event handlers for each buttons
        btApp1.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent actionEvent) {
//                TODO: Replace the scene or the root
//                      Display the "Animation" in the CENTER,
//                      and a "Back to Main" on the TOP
                System.out.println("Clicked on Animation button");
            }
        });

        btApp2.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent actionEvent) {
//                TODO: Replace the scene or the root
//                      Display the "2D Drawing" in the CENTER,
//                      and a "Back to Main" on the TOP
                System.out.println("Clicked on Graphics 2D button");
            }
        });

        btApp3.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent actionEvent) {
//                TODO: Replace the scene or the root
//                      Display the "About" in the CENTER,
//                      and a "Back to Main" on the TOP
                System.out.println("Clicked on About button");
            }
```

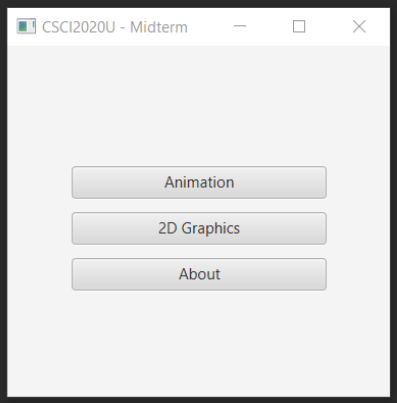


Figure 1. Event Handlers screenshot and the Main application UI.

The “Animation” Option

This UI will display an animation to the user using the ducks.png (Figure 2) sprite located in the “resources” folder. You **must not** alter the image, for example, image cropping.

- The sprite sheet has 4 types of ducks {wild, white, tan, grey} divided in columns {1-3, 4-6, 7-9, and 10-12} respectively.
- Each of the character actions are from left to right, and each row is a different view.

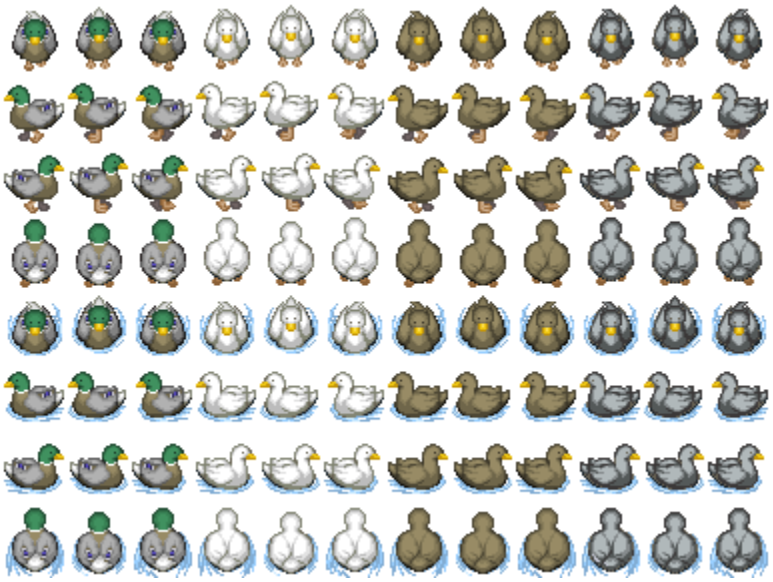


Figure 2. Source: http://www.rpgmakervxace.net/topic/2399-grannys-lists-animal-sprites/

Your task is to build an animation for **only 1 type of duck (see table below)**, where each row (view) will last 4 seconds and it will switch to the next view which will play for 4 seconds, and so on.
After all views were animated, you will loop back into the first view.

Last 2 Digits of your Student ID – OOOOOOOXX	Duck Type
00 – 25	Wild
25 – 50	White
50 – 75	Tan
75 – 100	Grey

The “2D Graphics” Option

Using 2D graphics, you will draw your name initials (letters) without using a **Text** or **SVGPath** objects, you must only use other basic shapes like lines, ellipses, rectangles, etc. You may also use properties like fill, stroke, etc.

Feel free to make it personal and express your personality with colors and typeface style. Besides the requirements of not using Text or SVGPath, the initials should be readable using the Roman alphabet as the baseline.

For reference add a label with your initials, so the grader can know which letters you are supposed to draw.

The “About” Option

This option will display information about you.
You may choose how to display the information (i.e. Text, Label, TextBox, Link).
The information displayed should be read from a XML file created by you in the “resources” folder following the template:

```
<info>
      <student id=””>
            <name>  </name>
           <email>  <\email>
      </student>
      <software-description>
           Some description in text
      </software-description>
</info>
```

Part II Submission

You will submit 1 zip file with your project named “CSCI2020U-Midterm-LastName-FirstName-studentID.zip”.

Submit your solution including

- the “resources” folder with your info.XML file
- all the .java files
- README.md file with instruction on “How to run” your program.

If you used IntelliJ you may save you project as .zip file (File/Export/Project as zip)