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: Benjamin Tsoumagas

Student ID: 100751395

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.

```
#Ignore class files
*.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:

```
mkdir MyProject2
cd MyProject2
git init
touch HelloWorld.java
touch GoodNight.csv
touch .gitignore
```

(we can add to these files using a text editor/IDE, when we run HelloWorld.java, HelloWorld.class will be made)

```
git add .
git commit -m"Initial Commit"
```

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'.

```
plugins{
      id 'java'
      id 'application'
}

repositories{
      mavenCentral()
}

dependencies{
      implementation 'org.json'
}

application{
      mainClassName = 'CalculateTaxes'
}
```

(in front of CalculateTaxes is the directory structure containing build.gradle and src/main/java where CalculateTaxes.java exists)

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.

Coding best practices are a set of rules that developers should try to adhere to as best as possible. Some best practice rules are general and can apply to all developers, while others are language or company specific. For example, in Java you should use 'yoda conditions' i.e. if(null=account){…} rather than if(account=null){…}. Java programmers should also always use {…} for their if statements and while/for loops. For example, 'if(application.isRunning()){…}' is better than ' if(application.isRunning())…' . Where applicable, Java programmers should also check for null values i.e. 'if(null != fileContents){…}' is better than '…' when reading from a file. You should use nouns for variable names and verbs for methods and use camel case for both (i.e. 'balance' and 'getTotal()'). Documentation in Java is a specific style (contained in /* */) and describes the purpose of the variable/method and all parameters, return values, and exceptions that apply. Following coding best practices makes for code that is more readable across the entire team so other team members can help improve and maintain code that others have made. Following conventions also makes code easier to follow for yourself when you come back to it. Best practices also lead to safer code so that your code is less likely to cause issues in the future. In summary, following coding best practices, increases readability, maintainability, safety, and quality of code.

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?

The programmatic method means to do everything in the same unified style/format of code. It is more readable and generally better to adopt the model view controller architecture which FXML is based on. FXML consists of a main class, a controller class (with additional classes referenced in the controller class), and a fxml file. The programmatic method doesn't require the knowledge of the MVC architecture and requires less files for simple projects. However, the FXML approach helps to break projects into a java file that handles user action (controller), a fxml file that handles the user display (view), and all the domain objects the controller interacts with (model). Breaking down projects into visuals for the user and the logic for the application allows people who only know back-end or front-end to work on a project, whereas the programmatic approach requires knowledge of both.

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

```
year, month, province, region, city, cases
2020, march, ontario, peel, brampton, 160000
2020, march, ontario, dufferin, orangeville, 25000
2020, august, ontario, wellington, erin, 35000
2021, january, ontario, peel, brampton, 180000
2021, february, ontario,  dufferin, orangeville, 30000
```

b. JSON

```
{ “ontario” : [
      “peel” : [
                  {
                   “year”: “2020”,
                    “month”: “march”,
                    “city”: “brampton”,
                    “cases”: 160000
                  },
                  {
                    “year”: “2021”,
                  “month”: “january”,
                  “city”: “brampton”,
                  “cases”: 180000
                 }
                 ],
        “dufferin” : [
                 {
                  “year”: “2020”,
                  “month”: “march”
                  “city”: “orangeville”,
                  “cases”: 25000
                 },
                 {
                  “year”: “2021”,
                  “month”: “february”
                  “city”: “orangeville”,
                  “cases”: 30000
                 }
                 ],
        “wellington” : [
                 {
                  “year”: “2020”,
                  “month”: “august”,
                  “city”: “erin”,
                  “cases”: 35000
                 }
                 ]
}
```

c. XML

```
<reports latestUpdate=”123982479528” version=”2.0”>
        <ontario>
                <regions>
                        <region = peel>
                                <year>2020</year>
                                <month>march</month>
                                <city>brampton</city>
                                <cases>160000</cases>
                        </region>
                        <region = peel>
                                <year>2021</year>
                                <month>january</month>
                                <city>brampton</city>
                                <cases>180000</cases>
                        </region>
                        <region = dufferin>
                                <year>2020</year>
                                <month>march</month>
                                <city>orangeville</city>
                                <cases>25000</cases>
                        </region>
                        <region = dufferin>
                                <year>2021</year>
                                <month>february</month>
                                <city>orangeville</city>
                                <cases>30000</cases>
```

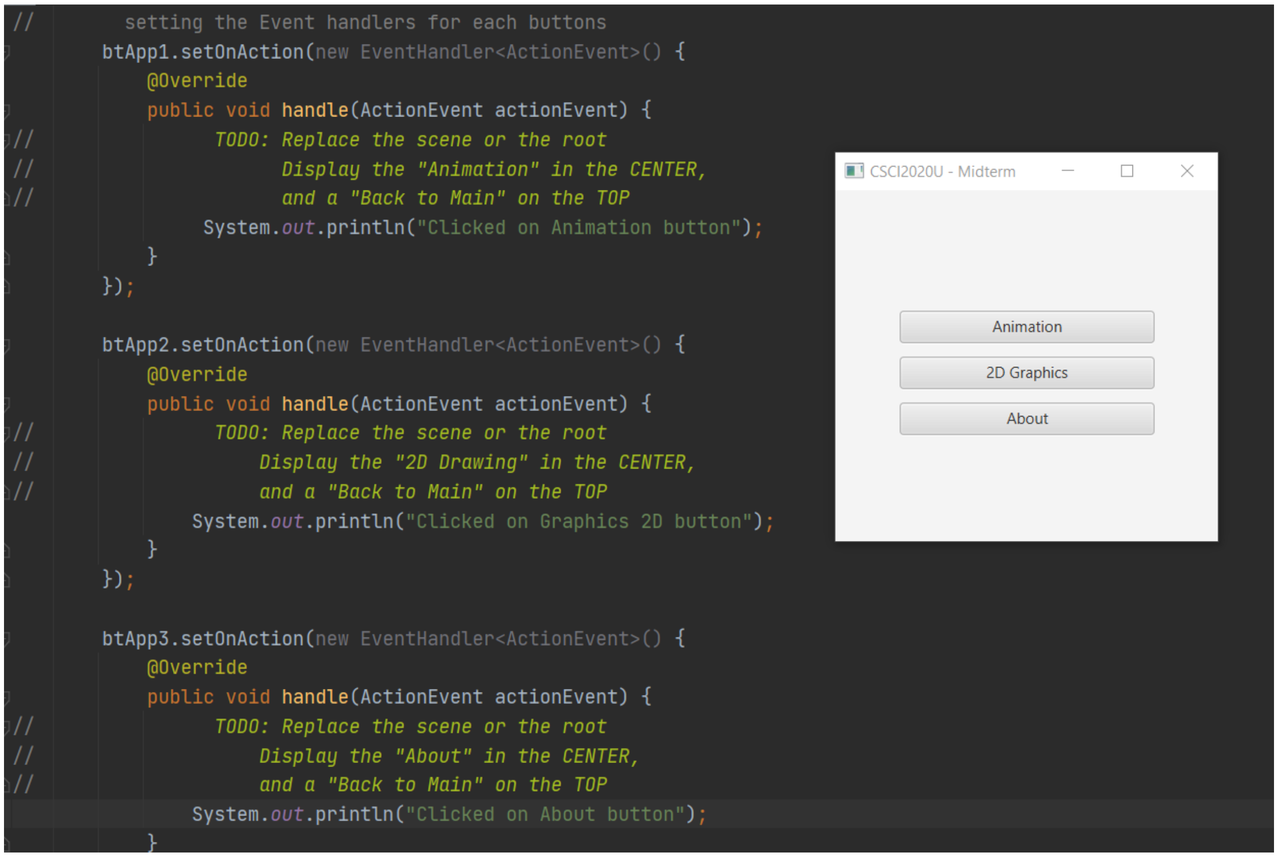
```
				</region>
				<region = wellington>
					<year>2020</year>
					<month>august</month>
					<city>erin</city>
					<cases>35000</cases>
				</region>
			</regions>
		</ontario>
	</reports>
```

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.

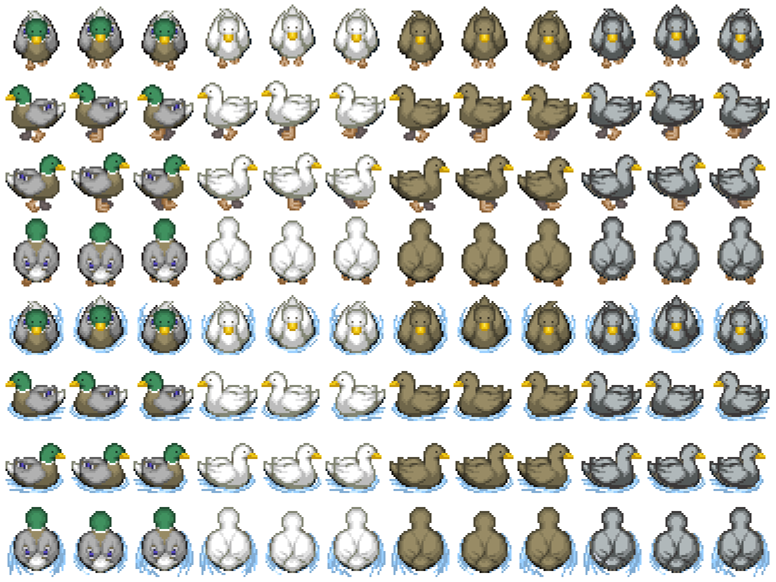


Figure 2. Source: http://www.rpgmakervxace.net/topic/2399-grannys-lists-animal-sprites/

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)