DUE DATE: **By the end of your lab session**

CSE-271: Object-Oriented Programming

Exercise #7

Max Points: 20

Name: Jacob Igel

For your own convenient reference – You should first save/rename this document using the naming convention **`MUid_Exercise7.docx`** (example: `raodm_Exercise7.docx`) prior to proceeding with this exercise.

Objectives: The objectives of this exercise are to:

1. Review the concepts of Graphical User Interfaces (GUI)
2. Review basics of Swing class hierarchy (`JFrame`, `JPanel`, `JButton`)
   a. Develop a custom `JComponent` via inheritance & polymorphism
3. Gain some familiarity layout managers (`GridLayout`)
4. Gain familiarity with GUI builder – Eclipse's `Window Builder` tool

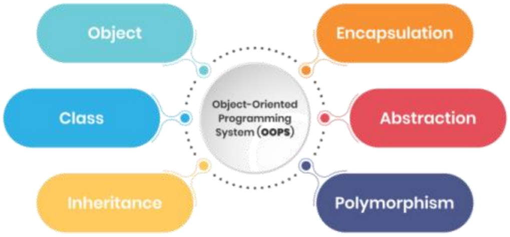
Fill in answers to all of the questions. For some of the questions you can simply copy-paste appropriate text from Eclipse output into this document. You may discuss the questions or seek help from your neighbor, TA, and/or your instructor.

Part #0: One time setup of Eclipse (IDE) – Only if needed

We already configured `Eclipse`'s source formatter and Checkstyle plug-in as part of Lab #1. If your `Eclipse` is not configured (because you are using a different computer) then use the instructions from Lab #1 to configure `Eclipse`.

Part #1: GUI basics and Object-oriented programming

Estimate time: < 30 minutes



Background: Object-oriented Programming (OOP) is a programming paradigm that is heavily used for developing GUI libraries such as Swing and gaming libraries such as Unity. OOP enables developing an flexible and unambiguous (*i.e.*, intention is clearly communicated between programmers who develop libraries versus those who use the libraries).
It is also important to remember that building a GUI essentially involves same basic concepts:

- Creating objects and calling methods with suitable arguments
- Using existing methods for basic problem-solving.

Exercise: Briefly (2-to-3 sentences each) respond to the following questions regarding generic concepts of Graphical User Interface (GUI)

1. What is a Graphical User Interface (GUI)?

   A GUI is a system of interactive visual components for computer software. A GUI displays objects that convey information, and represent actions that can be taken by the user. The objects change color, size, or visibility when the user interacts with them.

   Source: https://www.computerhope.com/jargon/g/gui.htm

2. Briefly (2-to-3 sentences) describe 1 advantage offered by a GUI

   Unlike a command-line operating system or CUI, like Unix or MS-DOS, GUI operating systems are easier to learn and use because commands do not need to be memorized. Additionally, users do not need to know any programming languages.

   Source: https://www.computerhope.com/jargon/g/gui.htm

3. Briefly (2-to-3 sentences) describe 1 disadvantage of GUIs

   A disadvantage of GUIs are that they provide less flexibility to the user. The GUI is can only execute the instructions that are already preprogrammed.

   Source: https://www.ionos.com/digitalguide/websites/web-development/what-is-a-gui/

4. Referring to the following Wikipedia page https://en.wikipedia.org/wiki/Xerox_Alto, briefly (2-to-3 sentences) describe the importance of the `Xerox Alto`.

   "The Alto became well known in Silicon Valley and its GUI was increasingly seen as the future of computing. In 1979, Steve Jobs arranged a visit to Xerox PARC, during which Apple Computer personnel would receive demonstrations of Xerox technology in exchange for Xerox being able to purchase stock options in Apple.[10] After two visits to see the Alto, Apple engineers used the concepts to introduce the Apple Lisa and Macintosh systems."

Source: https://en.wikipedia.org/wiki/Xerox_Alto

5. GUIs use pixels in many different colors to create a picture. The colors of pixels are determined based on a combination of 8-bits (or 256 distinct values) of red, green, and blue (RGB) colors (sadly, RAObot's eyes are only 3-bit color). The colors can be represented as RGB triplets or correspondingly as hex codes. Complete the following table using information from https://htmlcolorcodes.com/color-names (the first one is already completed to illustrate an example):

Name	RGB	HTML
Miami Pantone Red	rgb(152, 0, 46)	#98002e
Light Salmon	rgb(255, 160, 122)	#FFA07A
Deep Pink	rgb(255, 182, 193)	#FFB6C1
Lemon Chiffon	rgb(255, 250, 205)	#FFFACD
Medium Aquamarine	rgb(102, 205, 170)	#66CDAA
Ghost White	rgb(248, 248, 255)	#F8F8FF

6. What is the event-driven programming (or approach) that is used in GUIs?

   In computer programming, event-driven programming is a programming paradigm in which the flow of the program is determined by events such as user actions (mouse clicks, key presses), sensor outputs, or message passing from other programs or threads. Event-driven programming is the dominant paradigm used in graphical user interfaces and other applications (e.g., JavaScript web applications) that are centered on performing certain actions in response to user input.

   Source: https://en.wikipedia.org/wiki/Event-driven_programming

7. What does mean to "*fire an event*"? When are events fired?

   Process of generating an event is called firing an event. All actions (drawing, mouse movements, key press on keyboard) are represented as events occurring in the system.

   Source: Powerpoint

8. Briefly (2-to-3 sentences) describe 1 advantage offered by a GUI-builder program

   The system responds to instructions that the user can enter intuitively thanks to the visual format. This gives users with little technical knowledge the ability to use a GUI-builder program.

   Source: https://www.ionos.com/digitalguide/websites/web-development/what-is-a-gui/

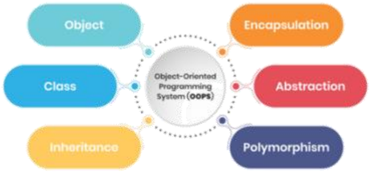
9. Briefly (2-to-3 sentences) describe 1 disadvantage of GUI-builder programs

   Sometimes the GUI-builder can be slow when actually editing the program. If the editing is slow then the actual performance will be slow as well.

   Source: https://cheesecakelabs.com/blog/interface-builder-pros-cons-thoughts/

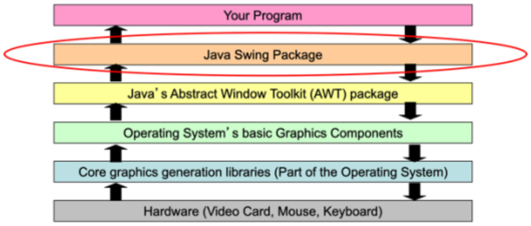
Part #2: Java-Swing and GUI programming

Estimate time: < 30 minutes



Background: Object-oriented Programming (OOP) is a programming paradigm that is heavily used for developing GUI libraries such as Swing and gaming libraries such as Unity. OOP enables developing a flexible and unambiguous (*i.e.*, intention is clearly communicated between programmers who develop libraries versus those who use the libraries).
It is also important to remember that building a GUI essentially involves same basic concepts:

- Creating objects and calling methods with suitable arguments
- Using existing methods for basic problem-solving.


Your Program
Java Swing Package
Java's Abstract Window Toolkit (AWT) package
Operating System's basic Graphics Components
Core graphics generation libraries (Part of the Operating System)
Hardware (Video Card, Mouse, Keyboard)

Exercise: Briefly (2-to-3 sentences each) respond to the following questions involving the Java-Swing GUI library

1. What is a layout manager?

   The layout manager automatically positions all the components within the container. Even if you do not use the layout manager, the components are still positioned by the default layout manager.

   Source: https://www.tutorialspoint.com/swing/swing_layouts.htm

2. What is a “functional interface”?

   Java streamlines the use of Lambdas for functional interface:
   – Functional interfaces have exactly 1 method
   – Hence their parameters & returns values are easily deduced

Source: Powerpoint

3. Rewrite the following methods that involve anonymous classes to use the more concise lambda syntax:

Anonymous class syntax	Corresponding lambda syntax

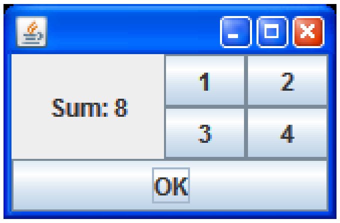
`doIt.addActionListener(new`
`  ActionListener() {`
`    @Override`
`    public void`
`    actionPerformed(ActionEvent e) {`
`      System.out.println("Click!");`
`    }`
`  });`
	doIt.addActionListener((e) -> { System.out.println(“Click!”); });

Part #3: Working with Eclipse’s Window Builder

Estimated time: < 30 minutes

Background: GUI-builders are software tools that are used to ease development and/or rapid-prototyping of GUI. Window Builder is a Java-Swing GUI-builder that is available for Eclipse. It provides a visual approach for laying out components and adding actions to components. As the GUI is developed, the corresponding source code for it is automatically generated/updated by Window Builder.

Exercise: In this part of the exercise, you are expected to install and use Window Builder to develop the GUI shown in the adjacent screenshot.



There is a video on Canvas demonstrating the use of Eclipse’s `Window Builder`. It may be useful to quickly review these videos as part of this exercise.

Part #4: Submit to Canvas via CODE plug-in

Estimated time: < 5 minutes

Exercise: You will be submitting the following files on Canvas as normal submissions:

1. This MS-Word document saved as a PDF file – **Only submit PDF file**.
2. The Java source file for the GUI that you developed using Window Builder

DUE DATE: **By the end of your lab session**

Ensure you actually complete the submission on Canvas by verifying your submission (after you submit)