Full Name: _____

Student ID#: _____

CSE 1325 OBJECT-ORIENTED PROGRAMMING

PRACTICE -=#=- Exam #2 –=# 1 M #=– Java -=#=- PRACTICE

Instructions

1. Students are allowed pencils, erasers, and beverage only.
2. All books, bags, backpacks, (silenced) phones, **smart watches**, and other electronics, etc. must be placed under the chalkboard at the front of the room.
3. PRINT your name and student ID at the top of this page, and verify that you have all pages.
4. If you leave the room, you may not return.
5. Read all questions and answer only what is written. Please do not write random stuff if you don't know the answer.
6. If you have a question, please raise your hand. You may or may not get an answer, but it won't hurt to ask.

Vocabulary

Write the word or phrase from the Words list below to the left of the definition that it best matches. Each word or phrase is used at most once, but some will not be used. {15 at 2 points each}

Vocabulary

Word	Definition
1	A special class member that cleans up when an object is deleted (not supported by Java)
2	An encapsulated bundle of data and code
3	A managed memory technique that tracks the number of references to allocated memory, so that the memory can be freed when the count reaches zero
4	Specifying a general interface while hiding implementation details
5	A named scope
6	A block of memory associated with a symbolic name that contains a primitive data value or the address of an object instance
7	The class inheriting members
8	Reuse and extension of fields and method implementations from another class
9	A derived class or interface inheriting members from two or more base classes or interfaces
10	A template encapsulating data and code that manipulates it
11	A method that returns the value of a private variable
12	A data type that can typically be handled directly by the underlying hardware
13	In program that runs in managed memory systems to free unreferenced memory
14	Bundling data and code into a restricted container
15	An instance of a class containing a set of encapsulated data and associated methods

Word List

Abstraction	Attribute	Class	Constructor	Destructor
Encapsulation	Enumerated type	Garbage Collector	Getter	Inheritance
Instance	Method	Multiple Inheritance	Namespace	Object
Object-Oriented Programming (OOP)	Operator	Override	Package	Primitive type
Reference Counter	Setter	Subclass	Superclass	Variable

Note: ALL vocabulary words may be on the actual exam!

Multiple Choice

Read the full question and every possible answer. Choose the one best answer for each question and write the corresponding letter in the blank next to the number. {15 at 2 points each}

1. ____ **A lambda**

    A. may override one or more abstract methods from the specified superclass

    B. requires that the method body be included in curly braces and include a return statement

    C. creates and instances an anonymous class that implements an interface or abstract class

    D. creates an anonymous method as a first-class object that can be passed as a parameter of type "Method"

2. ____ **To ensure clean handling of I/O errors, use**

    A. nested try / catch as required

    B. try-with-resources

    C. ternary-try

    D. try / catch / finally

3. ____ **The statement**
```
button.addActionListener(event -> System.out.println("Click!\n"));
```

    A. prints "Click!" to the console each time `button` is clicked

    B. displays "Click!" in a dialog box each time `button` is clicked

    C. displays "Click!" on the button each time it is clicked

    D. will not compile, because no type is specified for `event`

4. ____ **To read or write a file, we need to know**

    A. its path and name

    B. its name and file format

    C. it path, name, and size

    D. its name, access permissions, and inode

5. ____ **To open** `filename` **for reading, write**

    A. `File f = fopen(filename, 'r');`

    B. `BufferedReader br = new BufferedReader(new FileReader(filename));`

    C. `File f = filename.open('r');`

    D. `FileReader f = new BufferedReader.createFileReader(filename);`

6. ____ **What is TRUE about painting the Swing JPanel widget?**

    A. The paintComponent method is called by Swing when the the JPanel widget needs to be redrawn

    B. The Graphics context is usually duplicated before modification using its create method

    C. To request that paintComponent be called by Swing, our code calls repaint()

    D. All of these are true

7. ____ **Which of the following declares a Swing main window class for our application named Mainwin?**

    A. `public class Mainwin extends JFrame`

    B. `public JFrame Mainwin`

    C. `public class Mainwin implements Window`

    D. `public class Mainwin extends JMainWindow`

8. ____ **Which of these statements about Swing is FALSE?**

    A. Swing is implemented using Java classes with multiple inheritance

    B. Swing programs are portable, and run on Linux, Windows, or OS X

    C. A Swing program may include both a menu bar and a tool bar in a main window

    D. All of these statements are true

9. ____ **To respect encapsulation while saving and loading program data,**

    A. use Object.pickle(File) so automatically save each files attributes to the file

    B. delegate reading and writing each class's attributes to that class

    C. add a save() method to each class that returns a struct of its private data

    D. create an I/O class in the package and set all attributes to package-private

10. ____ **The layout of widgets in a Swing window is determined by**

    A. Absolute coordinates and widget widths measured in pixels

    B. Callbacks registered with JWindow layout handlers

    C. Containers with predefined layout rules such as Box or GridBag

    D. A layout manager object such as GridBagLayout

11. ____ **The best dialog to enable an artist to select a color for drawing on a JPanel is**

    A. ColorChooserDialog

    B. JOptionPane.showInputDialog

    C. JDialog with a JSlider each for red, green, blue, and alpha

    D. A menu on the menu bar with simple color names like "Red", "Green", "Blue", and "Orange"

12. ____ **To print a string on JPanel canvas, call**

    A. `canvas.translate("Hola", 30, 30)`

    B. `canvas.drawString("Hi", 30, 30)`

    C. `canvas.setFont("Hi", 30, 30)`

    D. `canvas.println("Hi", 30, 30)`

13. ____ **Given interface Openable with abstract method** `void open();`**, a lambda implementation of Openable would be**

    A. `new Openable {void open() {System.out.println("Open!");}`

    B. `class Open implements Openable {System.out.println("Open!");}`

    C. `() -> System.out.println("Open!")`

    D. `event -> System.out.println("Open!")`

14. ____ **To display bold, monospaced (teletype) text in JLabel label, write**

    A. `label.setText("<html><tt><b>I am Groot</b></tt></html>");`

    B. `label.setText("<tt><b>I am Groot</b></tt>");`

    C. `label.setText("I am Groot").bold().tt();`

    D. `label.bold(); label.tt(); label.setText("I am Groot");`

15. ____ **Swing represents a color using**

    A. a Color class

    B. a 4-byte int with the bytes representing red, green, blue, and alpha

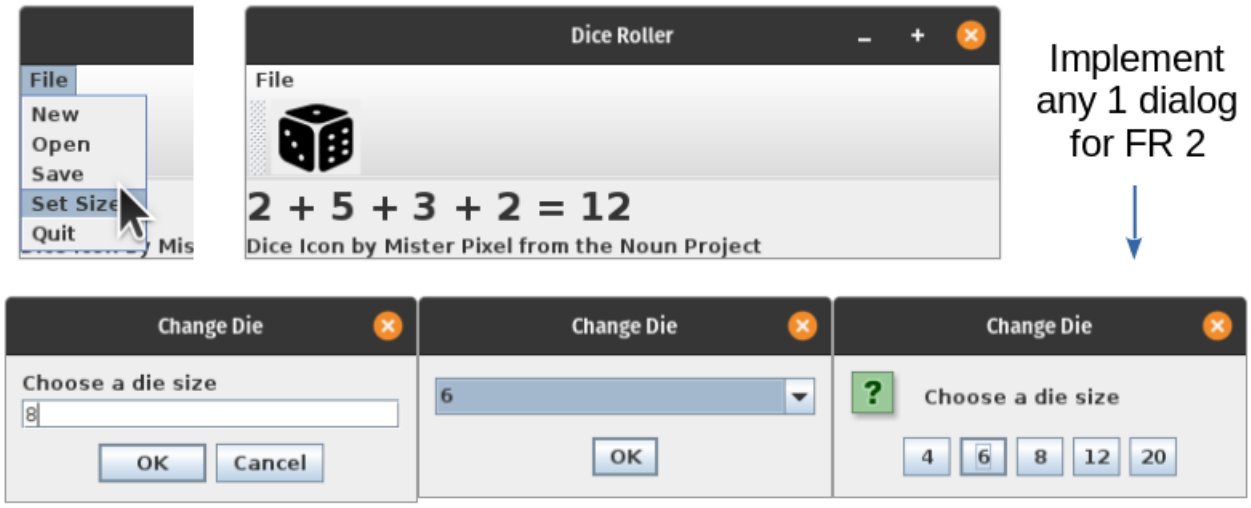
    C. a 3-byte int with the bytes representing red, green, and blue

    D. None of these

Free Response

Provide clear, concise answers to each question. Each question implements a small portion of the Java Swing application whose GUI and introductory code is shown below for reference. Each question, however, is *completely independent* of the other questions, and is intended to test your understanding of one aspect of Java programming. **Write only the code that is requested.** You will NOT write the entire application!

The information on class Dice below applies to Free Response questions 1 through 3.




```
// Assume all needed imports

public class Dice extends JFrame {

   private ArrayList<Integer> rolls = new ArrayList<>();
   private int size = 6;

   private JLabel data;
   private JLabel msg;

   public static void main(String[] args) {
       Dice myApp = new Dice("Dice Roller");
       myApp.setVisible(true);
   }

   public Dice(String title) {
       super(title);
       setSize(480,160);

       // Continue writing the constructor as specified below
```

1.a. {2 points} In the Dice constructor, write code to configure the main window Dice to exit if the close button (the 'x' in the top corner of the window) is clicked.

1.b. {4 points} In the Dice constructor, write code to create and add to the main window a menu bar. Show code to add File > Set Size ONLY that calls onSetSizeClick() when selected. (Other menu items should NOT be coded.)

1.c. {4 points} In the Dice constructor, write code to create and add to the main window a toolbar just below the menubar. Show code to add a button with its icon from file "roll.png", with action command and tool tip both "Roll die", and that calls onRollDieClick() when clicked.

1.d. {2 points} In the Dice constructor, write code to create and add to the main window a label in the center section to display data, referenced by attribute `data` (shown in the code above).

1.e. {2 points} In the Dice constructor, write code to display and execute the main window.

2. {8 points} Write method Dice.onSetSizeClick(). This method should display a dialog allowing the user to select from integers 4, 6, 8, 12, or 20. You may use buttons, a combo box, or a text box for this selection. A JOptionPane static method or custom JDialog is acceptable. Assign the selected integer to attribute `size`. If the user selects an invalid size (for text box only), display an error dialog. If the user selects Cancel or the close button (x), exit the dialog with no error dialog and with no change to `size`.

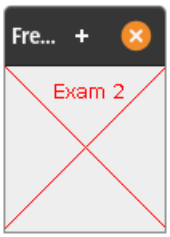
3. {10 points} Write method onOpenClick(). Open file "untitled.dice" for reading, which contains a sequence of previous die rolls, one per line. Read each line, converting the String to an int and adding that to ArrayList rolls. Use try-with-resources to display a message dialog "Unable to read untitled.dice" on an IOException.

4. {8 points} Give class X below, write package-private class Canvas. Canvas should draw a 100-pixel sized red X on the screen, and write "Exam 2" in red starting at (30,20) in the default font.

```
// Assume all needed imports

public class X extends JFrame {
   public X() {
       super("Free Response 4");
       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       setSize(100,140);
       Canvas canvas = new Canvas();
       add(canvas);
   }

   public static void main(String[] args) {
       X x = new X();
       x.setVisible(true);
   }
}
```



Bonus: {up to 4 points} List and briefly describe the two different backlogs used to plan work in a small Scrum project.

Mark question number for solutions. Additional paper is available on request.