COE 593 – COE Application

Welcome to the Midterm Exam
Friday November 21, 2014

Instructor: Dr. Wissam F. Fawaz

Name: _____

Student ID: _____

Instructions:

1. This exam is **Closed Book**. Please do not forget to write your name and ID on the first page.
2. You have exactly **60 minutes** to complete the 4 required problems.
3. Read each problem carefully. If something appears ambiguous, please write your assumptions.
4. Points allocated to each problem are shown in square brackets.
5. Put your answers in the space provided only. No other spaces will be graded or even looked at.

Good Luck!!

Problem 1: Lambdas and Streams (**10 minutes**) [20 points]

1) Which one of the following functional interfaces contains a method called `apply` that takes two `T` arguments, performs an operation on them and returns a value of type `T`.
   a. `Consumer`
   b. `Function`
   c. `Supplier`
   d. `BinaryOperator`

2) Which one of the following generic functional interfaces contains a method called `accept` that takes a `T` argument, performs a task with it, such as outputting the object, and returns `void`? Note that the interface in question is usually used in conjunction with the `forEach` terminal operation.
   a. `Function`
   b. `Consumer`
   c. `BinaryOperator`
   d. `Supplier`

3) Which one of the following generic functional interfaces contains a method called `test` that takes a `T` argument and returns a `boolean`?
   a. `BinaryOperator`
   b. `Supplier`
   c. `Predicate`
   d. None of the above

4) What is the meaning of ( ) (empty set of parentheses) in the following lambda:
   `() -> System.out.println("Welcome to Lambdas");`
   a. The lambda parameters are inferred
   b. The lambda parameters are supplied by a method reference
   c. The lambda has an empty list of parameters
   d. The given expression is not a valid lambda

5) Which of the following intermediate operations results in a stream containing only the elements that satisfy a certain condition?
   a. `distinct`
   b. `map`
   c. `filter`
   d. `limit`

6) Which of the following is not an intermediate stream operation?
   a. `limit`
   b. `reduce`
   c. `filter`
   d. `sorted`

7) Which one of the following methods is a method of the `IntStream` class that can be used to perform the `count`, `min`, `max`, `sum`, and `average` operations in one pass?
   a. `allStatistics`
   b. `completeStatistics`
   c. `entireStatistics`
   d. `summaryStatistics`

8) Which one of the following methods is a method of the `Arrays` class that can be used to create a `Stream` from an array of objects?
   a. `arraysToStream`
   b. `stream`
   c. `createStream`
   d. None of the above

9) Which one of the following methods is a method of the `Map` class that can be used to perform an operation on each key-value pair of the map?
   a. `for`
   b. `forAll`
   c. `forNext`
   d. None of the above

10) Which of the following methods can be used to eliminate duplicate objects in a stream?
   a. `unique`
   b. `discrete`
   c. `different`
   d. None of the above

Problem 2: Miscellaneous (**10 minutes**) [20 points]

1) Which of the following is false?
   a. A `Path` object represents the location of a file or a directory
   b. The static method `get` of class `Paths` converts a `String` representing a file's or directory's location into a `Path` object
   c. A `DirectoryStream` enables a program to iterate through the contents of a directory
   d. None of the above is false

2) Which of the following classes is part of the `java.nio` package?
   a. `File`
   b. `Files`
   c. All of the above
   d. None of the above

3) Which of the following interfaces of the `java.nio` package can be used to filter files based on their extensions?
   a. `FileFilter`
   b. `Filter`
   c. All of the above
   d. None of the above

4) Which of the following statements can be used to create an object that can perform character-based input from a text file called "srcFile.txt"? Assume that the file has already been converted into a `Path` object called `srcPath`.
   a. `BufferedInputStream bis = new BufferedInputStream(Files.newInputStream(srcPath));`
   b. `BufferedReader br = Files.newBufferedReader(srcPath);`
   c. `BufferedReader br = new BufferedReader(new InputStreamReader(Files.newInputStream(srcPath)));`
   d. Both (b) and (c)

5) Consider the following Java segment:

```
String line1 = new String("c=1+2+3");
String[] tokens = line1.split("\\d");
int size = tokens.length;
```

   The value of `size` is:
   a. `1`
   b. `2`
   c. `3`
   d. None of the above

6) Which of the following does not match the regular expression "`\\w{2,}`"?
   a. AA
   b. AAA
   c. Both of the above
   d. None of the above

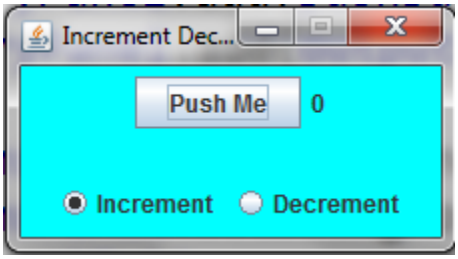
7) Which of the following is true about regular expressions?
   a. Ranges of characters can be represented by placing a ~ between two characters
   b. [^Z] is the same as [A-Y]
   c. Both "A*" and "A+" will match "AAA", but only "A*" will match an empty string
   d. All of the above statements are true

8) Which of the following classes enable input of entire objects from a file?
   a. `SerializedInputStream`
   b. `Scanner`
   c. `Formatter`
   d. None of the above

9) What interface must a class implement to indicate that objects of the class can be output and input as a stream of bytes?
   a. `Synchronize`
   b. `Deserializable`
   c. `ObjectOutputStream`
   d. `Serializable`

10) Which of the following symbols can be used to make a "quantifier" possessive in a regular expression?
   a. +
   b. ?
   c. *
   d. None of the above

Problem 3: GUI-based Applications (**20 minutes)** [30 points]



Design and implement an application that displays a button, a label and a set of two radio buttons to the end user (as depicted in the figure above). Every time the button is pushed, the value displayed by the label is either incremented or decremented depending on the radio button selected by the end user. If the increment button is turned on, the numeric value displayed is incremented by one each time the button is pushed. Similarly, if the decrement radio button is toggled on, the numeric value is decremented by one each time the button is pushed. *Hint:* the components can be organized into two panels, called primary and options panels. The primary panel can hold the button and the label components and occupy the "center" area of a border layout while the options panel can hold the radio buttons and occupy the "south" area of the border layout.

```
import javax.swing.JFrame;
public class IncrementDecrementFrame {
      public static void main(String[] args) {
            JFrame frame = new JFrame("Increment Decrement App");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.getContentPane().add(new
            IncrementDecrementPanel());
            frame.pack();
            frame.setVisible(true);
            frame.setResizable(false);
            frame.setLocationRelativeTo(null);
      }
} // end of IncrementDecrementFrame

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
public class IncrementDecrementPanel extends JPanel {
      // Declare instance variables here
      private JPanel primaryPanel, optionsPanel;
      private JButton pushButton;
      private JLabel outputTextLabel;
      private JRadioButton incRadioButton, decRadioButton;
      private int counter = 0;
      private ButtonGroup group;
```

```
      // Complete implementation of constructor
      public IncrementDecrementPanel() {
            setLayout(new BorderLayout());
            pushButton = new JButton("Push Me");
           pushButton.addActionListener(new PushButtonListener());
            outputTextLabel = new JLabel(counter + "");
            primaryPanel = new JPanel();
            primaryPanel.setBackground(Color.CYAN);
            primaryPanel.add(pushButton);
            primaryPanel.add(outputTextLabel);
            optionsPanel = new JPanel();
            optionsPanel.setBackground(Color.CYAN);
            incRadioButton = new JRadioButton("Increment");
            incRadioButton.setSelected(true);
            incRadioButton.setBackground(Color.CYAN);
            decRadioButton = new JRadioButton("Decrement");
            decRadioButton.setBackground(Color.CYAN);
            group = new ButtonGroup(); group.add(incRadioButton);
            group.add(decRadioButton);
            optionsPanel.add(incRadioButton);
            optionsPanel.add(decRadioButton);
            add(primaryPanel, BorderLayout.CENTER);
            add(optionsPanel, BorderLayout.SOUTH);
            setPreferredSize(new Dimension(200, 75));
            setBackground(Color.CYAN);
      }

      // Action listener for push button
      private class pushButtonListener implements ActionListener{
            @Override
            public void actionPerformed(ActionEvent e) {
            // Determine which radio button is selected,
            // then take appropriate action.
            if(incRadioButton.isSelected()) counter++;
            else counter--;
            outputTextLabel.setText(counter + "");
            }

      } // End of private class pushButtonListener

} // End of public class IncrementDecrementPanel
```

Problem 4: Regular Expressions (**20 minutes)** [30 points]

Write an application that uses classes from **both** the `java.nio` and the `java.io` packages to copy all the words that begin with a capital letter from an input text file called "**in.txt**" to an output text file called "**out.txt**". In "**out.txt**", each of the copied words must appear on a different line.

```
import java.nio.file.*;
import java.io.*;
public class CopyingCapitilizedWords {
   public static void main(String[] args) throws Exception{
     Path srcPath = Paths.get("in.txt");
     Path trgtPath = Paths.get("out.txt");
     BufferedReader inFromFile =
     Files.newBufferedReader(srcPath);
     BufferedWriter outToFile =
     Files.newBufferedWriter(trgtPath);
     String line;
     String[] words;

     while((line=inFromFile.readLine())!=null) {
          words = line.split("\\s+");
          for(String current : words) {
               if(beginesWithCapital(current)){
                    current = current + "\n";
                    outToFile.write(0, current,
                    current.length());
               }
          }
     }
     inFromFile.close();
     outToFile.close();
   }
   private static Boolean beginsWithCapital(String word) {
     return word.matches("[A-Z][a-z]+");
   }
}
```

JButton
- JButton(String)
- void setEnabled(boolean)
- void setMnemonic(char)
- void addActionListener(ActionListener)

JLabel
- JLabel()
- JLabel(String)
- String getText()
- void setText(String)

JPanel

void setPreferredSize(Dimension)
void add(Component)
void setBackground(Color)
void setLayout(LayoutManager)

JRadioButton
- JRadioButton(String)
- boolean isSelected()
- String getText()
- void setSelected(boolean)

ActionEvent
- String getActionCommand()
- Object getSource()
- int getModifiers()
- long getWhen()

java.io.BufferedReader
- BufferedReader(InputStreamReader)
- String readLine()

java.io.BufferedWriter
- BufferedWriter(FileWriter)
- void write(String s, int off, int len)

java.io.BufferedOutputStream
- BufferedOutputStream(OutputStream)
- void write(byte[] data, int off, int len)

java.io.FileWriter
- FileWriter(File)

java.io.PrintWriter
- PrintWriter(BufferedWriter)
- void print(String)
- void println(String)
- void close()

java.io.InputStreamReader
- InputStreamReader(InputStream)

java.nio.file.Files
- static BufferedReader newBufferedReader(Path)
- static BufferedWriter newBufferedWriter(Path)
- static InputStream newInputStream(Path)
- static OutputStream newOutputStream(Path)

java.nio.file.Paths
- static Path get(String)

java.nio.file.Path
- Path getFileName()
- File toFile()

java.lang.String
- String[] split(String)
- byte[] getBytes
- boolean matches(String)

java.util.regex.Matcher
- boolean find()
- String group()

java.util.regex.Pattern
- static Pattern compile(String)
- Matcher matcher(String)