Java Journal Template

Directions: Follow the directions for each part of the journal template. Include in your response all the elements listed under the Requirements section. Prompts in the Inspiration section are not required; however, they may help you to fully think through your response.

Remember to review the Touchstone page for entry requirements, examples, and grading specifics.

Name:Sarah Barettino

Final Replit Program Join Link: https://replit.com/join/wlecexkpge-shempill24

Complete the following template. Fill out all entries using complete sentences.

PART 1: Defining Your Problem

Task
State the problem you are planning to solve.

Requirements
- Describe the problem you are trying to solve.
- Describe any input data you expect to use.
- Describe what the program will do to solve the problem.
- Describe any outputs or results the program will provide.

Inspiration
When writing your entry below, ask yourself the following questions:
- Is your problem clearly defined?
- Why do you want to solve this particular problem?
- What source(s) of data do you believe you will need? Will the user need to supply that data, or will you get it from an external file or another source?
- Will you need to interact with the user throughout the program? Will users continually need to enter data in and see something to continue?
- What are your expected results or what will be the end product? What will you need to tell a user of your program when it is complete?

I would like to create a teacher's grade book. The input would be student information including first and last name, student ID, and 4 grades. There will also be a print to file option for all the student information to be organized and displayed on one page. This program would solve a teacher organizational issue.

PART 2: Working Through Specific Examples

Task
Write down clear and specific steps to solve a simple version of your problem you identified in Part 1.

Requirements
Complete the three steps below **for at least two distinct examples/scenarios**.

- State any necessary input data for your simplified problem.
- Write clear and specific steps in English (not Java) detailing what the program will do to solve the problem.
- Describe the specific result of your example/scenario.

Inspiration
When writing your entry below, ask yourself the following questions:

- Are there any steps that you don't fully understand? These are places to spend more time working out the details. Consider adding additional smaller steps in these spots.
- Remember that a computer program is very literal. Are there any steps that are unclear? Try giving the steps of your example/scenario to a friend or family member to read through and ask you questions about parts they don't understand. Rewrite these parts as clearly as you can.
- Are there interesting edge cases for your program? Try to start one of your examples/scenarios with input that matches this edge case. How does it change how your program might work?

User will need to enter

- Students first name
- Students last name
- Student ID
- Results for Test 1-3 and final grade

- User will enter their students information
- User will hit enter
- Program will calculate the average
- Programs will display the average on the screen
- Program will save a cumulative tally of all students entered in a session.
- If user hits the print button the program will create a .txt file for the user to print.

PART 3: Generalizing Into Pseudocode

Task
Write out the general sequence your program will use, including all specific examples/scenarios you provided in Part 2.

Requirements
- Write pseudocode for the program in English but refer to Java program elements where they are appropriate. The pseudocode should represent the full functionality of the program, not just a simplified version. Pseudocode is broken down enough that the details of the program are no longer in any paragraph form. One statement per line is ideal.

Help With Writing Pseudocode
- Here are a few links that can help you write pseudocode with examples. Remember to check out part 3 of the Example Journal Template Submission if you have not already. Note: everyone will write pseudocode differently. There is no right or wrong way to write it, other than to make sure you write it clearly and in as much detail as you can so that it should be easy to convert to code later.
    - https://www.geeksforgeeks.org/how-to-write-a-pseudo-code/
    - https://www.wikihow.com/Write-Pseudocode

Inspiration
When writing your entry below, ask yourself the following questions:
- Do you see common program elements and patterns in your specific examples/scenarios in Part 2, like variables, conditionals, functions, loops, and classes? These should be part of your pseudocode for the general sequence as well.
- Are there places where the steps for your examples/scenarios in Part 2 diverged? These may be places where errors may occur later in the project. Make note of them.
- When you are finished with your pseudocode, does it make sense, even to a person that does not know Java? Aim for the clearest description of the steps, as this will make it easier to convert into program code later.

Create an interface for user
- Container will have 5 sections
    - Welcome Message
        - Create label with Welcome Message, centered
    - Student information
        - 2 columns on for label one for text field for user input
            - Label and text field for Student First Name
            - Label and text field for Student Last Name
            - Label and text field for Student ID
    - Test information
        - 4 columns for the four tests labels will sit on top of text fields
            - Label and text field for test 1
            - Label and text field for test 2
            - Label and text field for test 3
            - Label and text field for Final

- Buttons for action items
  - Three buttons
    - Save
      - On hitting the Save button all of the user inputted information will be put into variables
      - Test variables will be used to compute an average score
      - Student information, Test score and Average will be put into a cumulative variable.
      - Entry is acknowledged with a student summary
      - Entry is acknowledged with a counter increase
      - After all information is gathered and manipulated textfields are reset
      - User can now enter next students information
    - Clear
      - Resets all the text fields DOES NOT reset cumulative student ledger variable.
    - Print
      - Student ledger cumulative variable is written to file in it's entirety.
      - File is saved as a .txt file.
      - Acknowledgement message is updated
      - Student counter is reset
- Summary footer
  - Student Score Summary
    - Initialized as "Please Enter All Student Information"
    - If all fields are not complete updates to "Please complete all fields before hitting Save."
    - When Student entry is saved updates to Student Name and their Average
    - When Print is hit updates to "Grade Book Saved"
  - Total student counter
    - Increments up every save
    - Resets after printing

PART 4: Testing Your Program

Task
While writing and testing your program code, describe your tests, record any errors, and state your approach to fixing the errors.

Requirements
- For at least one of your test cases, describe how your choices for the test helped you understand whether the program was running correctly or not.

For each error that occurs while writing and testing your code:
- Record the details of the error from Replit. A screenshot or copy-and-paste of the text into the journal entry is acceptable.
- Describe what you attempted in order to fix the error. Clearly identify which approach was the one that worked.

Inspiration
When writing your entry below, ask yourself the following questions:
- Have you tested edge cases and special cases for the inputs of your program code? Often these unexpected values can cause errors in the operation of your program.
- Have you tested opportunities for user error? If a user is asked to provide an input, what happens when they give the wrong type of input, like a letter instead of a number, or vice versa?
- Did the outcome look the way you expected? Was it formatted correctly?
- Does your output align with the solution to the problem you coded for?

I was struggling to get the program to not accept an input that did not have all fields full. I was trying to use an if statement but it wasn't acknowledging the == or != on the .getText() items or the variables I had made to hold and compute the information. I turned to stack overflow and found that I should be using the .equals instead of the ==. Then I was trying to figure out what not equals would be but ended up just encapsulating the code run in an else statement. There weren't an errors to screen shot since all the information was still valid the file output was just not what I wanted.

I was using VSCode as my IDE so I could work offline. When I transferred the code over I didn't create a .java file case appropriate so I got an error that I was able to correct by renaming the file.


Console
Shell
Run
9m on 12:06:55, 10/18
./Mainframe.java:12: error: class MainFrame is pub
lic, should be declared in a file named MainFrame.
java
public class MainFrame extends JFrame {
^
1 error
./Mainframe.java:12: error: class MainFrame is pub
lic, should be declared in a file named MainFrame.
java
public class MainFrame extends JFrame {
^
1 error

PART 5: Commenting Your Program

Task
Submit your full program code, including thorough comments describing what each portion of the program should do when working correctly.

Requirements
- The purpose of the program and each of its parts should be clear to a reader that does not know the Java programming language.

Inspiration
When writing your entry, you are encouraged to consider the following:
- Is each section or sub-section of your code commented to describe what the code is doing?
- Give your code with comments to a friend or family member to review. Add additional comments to spots that confuse them to make it clearer.

```
import javax.swing.*;
import javax.swing.border.EmptyBorder;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintWriter;

public class MainFrame extends JFrame {

    // Font variable for consistency and convenience

    final private Font mainFont = new Font("Segoe print", Font.BOLD, 18);
    final private Font testFont = new Font("Helvetica", Font.BOLD, 40);

    // global scope variables

    JTextField tfFirstName, tfLastName, tfTest1, tfTest2, tfTest3, tfFinal, tfID;
    JLabel lbWelcome, lbTest1, lbTest2, lbTest3, lbFinal, lbID, lbAcknowledge,
lbStudentCount;
    String outputBook = "Student Grade Book";
    int count = 0;
    String lastName="";

    public void initialize(){

        // Student info labels and textfields initialized and defaulted

        JLabel lbFirstName = new JLabel("Student's First Name");
        lbFirstName.setFont(mainFont);
        lbFirstName.setAlignmentX(Component.CENTER_ALIGNMENT);

        JLabel lbLastName = new JLabel("Student's Last Name");
        lbLastName.setFont(mainFont);

        JLabel lbID = new JLabel("Student's ID");
        lbID.setFont(mainFont);

        tfFirstName = new JTextField();
        tfFirstName.setFont(mainFont);

        tfLastName = new JTextField();
        tfLastName.setFont(mainFont);
```

```
        tfID = new JTextField();
        tfID.setFont(mainFont);

        // Test Labels and text fields initialized and defaulted

        JLabel lbTest1 = new JLabel("Test 1");
        lbTest1.setFont(mainFont);

        JLabel lbTest2 = new JLabel("Test 2");
        lbTest2.setFont(mainFont);

        JLabel lbTest3 = new JLabel("Test 3");
        lbTest3.setFont(mainFont);

        JLabel lbFinal = new JLabel("Final");
        lbFinal.setFont(mainFont);
        lbFinal.setLayout(new GridBagLayout());

        tfTest1 = new JTextField();
        tfTest1.setFont(testFont);

        tfTest2 = new JTextField();
        tfTest2.setFont(testFont);

        tfTest3 = new JTextField();
        tfTest3.setFont(testFont);

        tfFinal = new JTextField();
        tfFinal.setFont(testFont);

        // Header & Footer Labels initialized and set to default language

        lbWelcome = new JLabel("Welcome to your Student Calculator!");
        lbWelcome.setFont(mainFont);

        lbAcknowledge = new JLabel("Please Enter All Student Information");
        lbAcknowledge.setFont(mainFont);

        lbStudentCount = new JLabel("Students Entered: " + count);
        lbStudentCount.setFont(mainFont);


        // wrapper class for better containment of elements
        JPanel wrapper = new JPanel();
        wrapper.setLayout(new BorderLayout());

        // formatting for the header
```

```
        JPanel header = new JPanel();
        header.setBorder(new EmptyBorder(10, 0, 10, 0));
        header.setLayout(new GridBagLayout());
        header.add(lbWelcome);

        // Formatting for the footer

        JPanel footer = new JPanel();
        footer.setLayout(new GridLayout(2,1,10,20));
        footer.setBorder(new EmptyBorder(10, 20, 10, 0));
        footer.add(lbAcknowledge);
        footer.add(lbStudentCount);


        // Formatting and adding elements to the Student information field
        JPanel formPanel = new JPanel();
        formPanel.setLayout(new GridLayout(3,2,5,5));  //row, column, margin between
components
        formPanel.setBorder(new EmptyBorder(10,10,10,10));
        formPanel.add(lbFirstName);
        formPanel.add(tfFirstName);
        formPanel.add(lbLastName);
        formPanel.add(tfLastName);
        formPanel.add(lbID);
        formPanel.add(tfID);

        // Formatting adding elements to the Test field
        JPanel formPanel2 = new JPanel();
        formPanel2.setLayout(new GridLayout(2,4,10,20));
        formPanel2.setBorder(new EmptyBorder(60,10,70,10));
        formPanel2.setSize(100, 100);
        formPanel2.add(lbTest1);
        formPanel2.add(lbTest2);
        formPanel2.add(lbTest3);
        formPanel2.add(lbFinal);
        formPanel2.add(tfTest1);
        formPanel2.add(tfTest2);
        formPanel2.add(tfTest3);
        formPanel2.add(tfFinal);

        // Formatting for action buttons
        JButton btnOK = new JButton("Save");
        btnOK.setFont(mainFont);
        btnOK.setMargin(new Insets(10, 0, 10, 0));

        JButton btnClear = new JButton("Clear");
        btnClear.setFont(mainFont);
```

```
        btnClear.setMargin(new Insets(10, 0, 10, 0));

        JButton saveFile = new JButton("Print to File");
        saveFile.setFont(mainFont);
        saveFile.setMargin(new Insets(10,0,10,0));

        // Code for when the OK button is pushed
        btnOK.addActionListener(new ActionListener(){

            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub


                // Check to make sure all fields are completed
                if(tfFirstName.getText().equals("") || tfLastName.getText().equals("") ||
tfID.getText().equals("") || tfTest1.getText().equals("")||tfTest2.getText().equals("")||
tfTest3.getText().equals("") || tfFinal.getText().equals("")){
                        lbAcknowledge.setText("Please complete all fields before hitting
Save");

                }
                // runs the code for calculating students average
                else{
                    // inc student count for display
                    count ++;

                    // retrieve student information
                    String firstName = tfFirstName.getText();
                    String lastName = tfLastName.getText();
                    String studentID = tfID.getText();

                    // convert string entry to float so math can be applied
                    Float test1f = Float.parseFloat(tfTest1.getText());
                    Float test2f = Float.parseFloat(tfTest2.getText());
                    Float test3f = Float.parseFloat(tfTest3.getText());
                    Float finalf = Float.parseFloat(tfFinal.getText());

                    // Calculate the Students average
                    Float averagef = (test1f +test2f +test3f +finalf)/4;

                    // Update display for acknowledgement of student entry
                    lbAcknowledge.setText("Student: " +firstName + " "+ lastName + " with
an average of " + averagef);
                    lbStudentCount.setText("Students Entered: " + count);

                    // Add new student to output variable to be used for full student body
print out
```

```
                    outputBook +=  "\n" + lastName + ", " + firstName + " - " +studentID +"
| Test 1: " + test1f + " Test 2: " + test2f+ " Test 3: " + test3f + " Final: " + finalf + "
| Average: " + averagef;

                    // Resets text fields for next entry
                    tfFirstName.setText("");
                    tfLastName.setText("");
                    tfID.setText("");
                    tfTest1.setText("");
                    tfTest2.setText("");
                    tfTest3.setText("");
                    tfFinal.setText("");
                }

                }


        });

        // Code for when Clear button is pushed
        btnClear.addActionListener(new ActionListener(){

            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub

                // Resets all text fields
               tfFirstName.setText("");
               tfLastName.setText("");
               tfID.setText("");
               tfTest1.setText("");
               tfTest2.setText("");
               tfTest3.setText("");
               tfFinal.setText("");

            }

        });

        // Code for when Print button is pushed
        saveFile.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub

                // Prints to file code
```

```
                PrintWriter outputStudents;
                try {
                    outputStudents = new PrintWriter(new FileOutputStream(new
File("StudentsGrades.txt"), false));
                    outputStudents.println(outputBook);
                    outputStudents.close();

                    // reset variables and labels
                    lbAcknowledge.setText("Student Grade Book Saved!");
                    count = 0;
                    lbStudentCount.setText("Students Entered: " + count);


                } catch (FileNotFoundException e1) {
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                }

            }



        });



        // User interface edding elements, builds and formatting
        JPanel buttonsPanel = new JPanel();
        buttonsPanel.setLayout(new GridLayout(1, 3, 5, 5));
        buttonsPanel.add(btnOK);
        buttonsPanel.add(btnClear);
        buttonsPanel.add(saveFile);

        wrapper.add(formPanel, BorderLayout.NORTH);
        wrapper.add(formPanel2,BorderLayout.CENTER);
        wrapper.add(buttonsPanel, BorderLayout.SOUTH);

        JPanel mainPanel = new JPanel();
        mainPanel.setLayout(new BorderLayout());
        mainPanel.add(header, BorderLayout.NORTH);
        mainPanel.add(wrapper, BorderLayout.CENTER);
        mainPanel.add(footer, BorderLayout.SOUTH);

        add(mainPanel);

        //  main Panel default settings

        setTitle("Student Grade Book");
```

```
        setSize(500,600);
        setMinimumSize(new Dimension(300,300));
        setLocationRelativeTo(null);
        setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        setVisible(true);
        setResizable(false);

    }
// Starting the program calling and establishing the mainframe
    public static void main(String[] args) {
        MainFrame myFrame = new MainFrame();
        myFrame.initialize();

    }
}
```

PART 6: Your Completed Program

Task
Provide the Replit link to your full program code.

Requirements

- The program must work correctly with all the comments included in the program.

Inspiration

- Check before submitting your Touchstone that your final version of the program is running successfully.

https://replit.com/join/wlecexkpge-shempill24