Lab 08: GUI

Due date: By the end of your lab session of the week of Monday, October 17th

Goals

- Creating a simple Graphical User Interface (GUI) using JOptionPane

Description

One of the most well-known sources of aid is for university students is the Federal Student Aid. Millions of dollars are available in this fund to help students offset the costs of education related expenses and can be requested through a free application called the Free Application for Federal Student Aid, or more commonly, the FAFSA. The FAFSA takes into account many factors in determining both students' eligibility and total award amount.

You will be creating a GUI based **Student Financial Aid Calculator** program that can compute **eligibility** and **total award amount** based on input from the user.

The problem of estimating federal financial aid can be broken down into two major sub-problems:

1. Determining eligibility, that is: “Am I entitled to get the financial aid?”
2. Calculating total aid award

Eligibility is subject to a few requirements. Requirements also regulate *how much* is a student eligible to receive. Criteria is based data provided by the student. A backend developing team has already built the model for this program *(*`FAFSA.java`*, available on the wiki)* and is up to you to built a GUI that allows the program to take input from the user.

A diagram of the GUI you will be creating for this lab has been provided on the wiki. You should pay close attention to the flow of the program in the diagram so that your GUI contains all the correct dialog windows and transitions between them as specified.

Create a class called `FAFSAGUI.java` and create a main method. You will be doing all the coding for this lab within the main method.

Details

The following list should help you with the design flow of your GUI. Several methods from the JOptionPane class which you will need to use have been listed in the appendix. You may use these as a reference as you create the dialogue windows. Additionally, use the Java API.
The dialogues that must be displayed are noted by the prefix **[Dialog]** and the bullet lists that follow them are *some* of the parameters that you should pass when calling the JOptionPane method. It is up to you to choose which dialog type will fit best for each situation.

For each of the dialogues that receives input from the user, you should store the result in a variable you should store the result in a variable of the correct data type that will later be passed to the FAFSA constructor.

1. **[Dialog]** Display the welcome dialog.
    1. message: "Welcome to the FAFSA!"
    2. title: "Welcome"
    3. messageType: INFORMATION_MESSAGE

2. **[Dialog]** Display the `isAcceptedStudent` dialog.
    1. message: "Have you been accepted into a degree or certificate program?"
    2. title: "Program Acceptance"
    3. optionType: YES_NO_OPTION
    4. messageType: QUESTION_MESSAGE

3. **[Dialog]** Display the `isSSregistered` dialog.
    1. message: "Are you registered for the selective service?"
    2. title: "Selective Service"
    3. optionType: YES_NO_OPTION
    4. messageType: QUESTION_MESSAGE

4. **[Dialog]** Display the `hasSSN` dialog.
    1. message: "Do you have a social security number?"
    2. title: "Social Security Number"
    3. optionType: YES_NO_OPTION
    4. messageType: QUESTION_MESSAGE

5. **[Dialog]** Display the `hasValidResidency` dialog.
    1. message: "Do you have valid residency status?"
    2. title: "Residency Status"
    3. optionType: YES_NO_OPTION
    4. messageType: QUESTION_MESSAGE

6. **[Dialog]** Display the `age` dialog.
    1. message: “How old are you?”
    2. title: “Age”
    3. messageType: QUESTION_MESSAGE
    4. If input is negative
        1. **[Dialog]** Display an error dialog.
            - message: “Age cannot be a negative number.”
            - title: “Error: Age”
            - messageType: ERROR_MESSAGE
        2. Return to step 6.

7. **[Dialog]** Display the `creditHours` dialog.
    1. message: “How many credit hours do you plan on taking?”
    2. title: “Credit Hours”
    3. messageType: QUESTION_MESSAGE
    4. If input not in the range [1, 24], inclusive of both
        1. **[Dialog]** Display an error dialog.
            - message: “Credit hours must be between 1 and 24, inclusive.”
            - title: “Error: Credit Hours”
            - messageType: ERROR_MESSAGE
        2. Return to step 7.

8. **[Dialog]** Display the `studentIncome` dialog.
    1. message: “What is your total yearly income?”
    2. title: “Student Income”
    3. messageType: QUESTION_MESSAGE
    4. If input is negative
        1. **[Dialog]** Display an error dialog.
            - message: “Income cannot be a negative number.”
            - title: “Error: Student Income”
            - messageType: ERROR_MESSAGE
        2. Return to step 8.

9. **[Dialog]** Display the `parentIncome` dialog.
    1. message: “What is your parent's total yearly income?”
    2. title: “Parent Income”
    3. messageType: QUESTION_MESSAGE
    4. If input is negative
        1. **[Dialog]** Display an error dialog.
            - message: “Income cannot be a negative number.”
            - title: “Error: Parent Income”
            - messageType: ERROR_MESSAGE
        2. Return to step 9.

10. **[Dialog]** Display the isIndependent dialog.
    1. message: “Are you a dependent?”
    2. title: “Dependency”
    3. optionType: YES_NO_OPTION
    4. messageType: QUESTION_MESSAGE

11. **[Dialog]** Display the classStanding dialog.
    1. message: “What is your current class standing?”
    2. title: “Class Standing”
    3. messageType: PLAIN_MESSAGE
    4. selectionValues: {“Freshman”, “Sophomore”, “Junior”, “Senior”, “Graduate”}
    5. Convert selection to either Undergraduate or Graduate

12. Create a new FAFSA object with the user-input values

13. Calculate loan, grant, workStudy, and total awards
    1. Make method calls to the corresponding methods from the FAFSA class

14. **[Dialog]** Display the awards, formatted neatly in a message dialog.
    1. message: refer to the diagram at the beginning for an example. You may format the output however you wish as long as it contains the individual awards for loan, grant, and work-study as well as the total of the three.
    2. title: “FAFSA Results”
    3. messageType: INFORMATION_MESSAGE

15. **[Dialog]** Display the continue dialog.
    1. message: “Would you like to complete another Application?”
    2. title: “Continue”
    3. optionType: YES_NO_OPTION
    4. If user chooses to continue
        1. Return to step 1.
    5. Else
        1. Exit program

Turning in Your Work

You **must** turn in your “lab08” directory before leaving the lab session. Change your current directory to “cs180” and execute the following command:

```
turnin -c cs180=COMMON -p lab08 lab08
```

Rubric

- 40 pts: Correctly implements all the specified dialog windows
    - 2 pts: Welcome dialog
    - 2 pts: Program Acceptance dialog
    - 2 pts: Selective Service dialog
    - 2 pts: Social Security Number dialog
    - 2 pts: Residency Status dialog
    - 4 pts: Age dialog (continues prompting with error message dialog until valid entry)
    - 4 pts: Credit Hours dialog (continues prompting with error message dialog until valid entry)
    - 4 pts: Student Income dialog (continues prompting with error message dialog until valid entry)
    - 4 pts: Parent Income dialog (continues prompting with error message dialog until valid entry)
    - 2 pts: Dependency dialog
    - 4 pts: Class Standing dialog
    - 6 pts: FAFSA Results dialog (correct output, neatly formatted in dialog)
    - 2 pts: Continue dialog
- 5 pts: Proper error checking for Age, Credit Hours, Parent Income, and Student Income dialogues
- 5 pts: Program will keep running until user selects no for the final dialog window