

CIS611
Individual Practice Programming Assignment: PA07
Total Points: 20

GUIs & Event Driven Programming

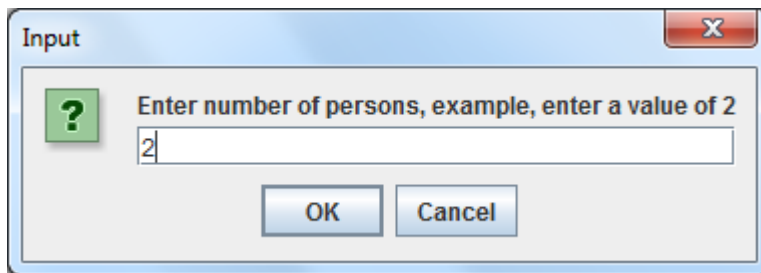
The purpose of this programming assignment is to:

- ☐ Create user interfaces using frames, panels and simple GUI components.

Q1 (20 points):

Write an interactive graphical user interface for the programming problem. Add a `UserGUI.java` class in order to allow a user to input required data fields for instantiating some objects. The coding assignment will involve coding the classes `Person.java`, `Faculty.java`, `Student.java`, `Address.java` using the object hierarchy. These are the classes coded in the prior PA06 assignment. The `MyDate.java` class is not used in this assignment. The GUI will be designed to accept the user input and will create instances of the various objects.

The program first prompts the user to enter the number of persons (which can be faculty and student). Based on the number entered, the program will loop to enter the faculty and the student information.



Both Faculty and Student will have the following common pieces of information that needed to be entered in the text area. These GUI elements have a label, and a corresponding text area, where information can be entered.

- Last Name (e.g. value to be entered in the text area is Doe)
- First Name ((e.g. value to be entered in the text area is John)
- Address
- Phone Number (e.g. value to be entered in the text area is (970) 491-6227)
- eMail (e.g. value to be entered in the text area is abc@gmail.com)

Radio button as Faculty and Student. Only one radio button can be selected.

If the Faculty radio button is selected, then display the list box with faculty with values Lecturer, Assistant Professor, Associate Professor, Professor. Only one value can be chosen. Design it as a

listbox.

If the Student radio button is selected, then display the list box with faculty with values Freshman, Sophomore, Junior, Senior. Only one value can be chosen. Design it as a listbox.

Add button. One click of the Add button creates one instance of Faculty (or Student), and adds it in the array of data type person(or Student). Create an array of personArray, which can hold either Faculty information, or Student information. The size of this personArray is set to the size of the persons, as obtained from the user input. When the Add button is clicked, the personArray gets populated with an instance of Faculty (or Student).

Sort button. When the Sort button is clicked, it sorts the elements in the personArray using the lastName of all the elements (both faculty and student) found in personArray.

Display text-area component. The text-area data must not be editable. When this button is clicked, both the Faculty and the Student information is displayed. Put a line separator between each Faculty/Student info. The String data fields returned by the object *toString()* method (of Faculty and Student) are displayed in the display text-area component. There is no requirement to use a slider (but it is fine if the student codes it), just size it accordingly.

The program will be tested by the grader by:

1. First by adding entering a value for the number of persons.
2. Next, adding one or more student and adding one or more faculty members (depends on the number of persons chosen)
3. Next clicking on the Sort button, which will sort the instances inside the personArray by the lastName.
4. Next, by clicking the Display button, which will display the persons information in the display text area, sorted by the lastName.

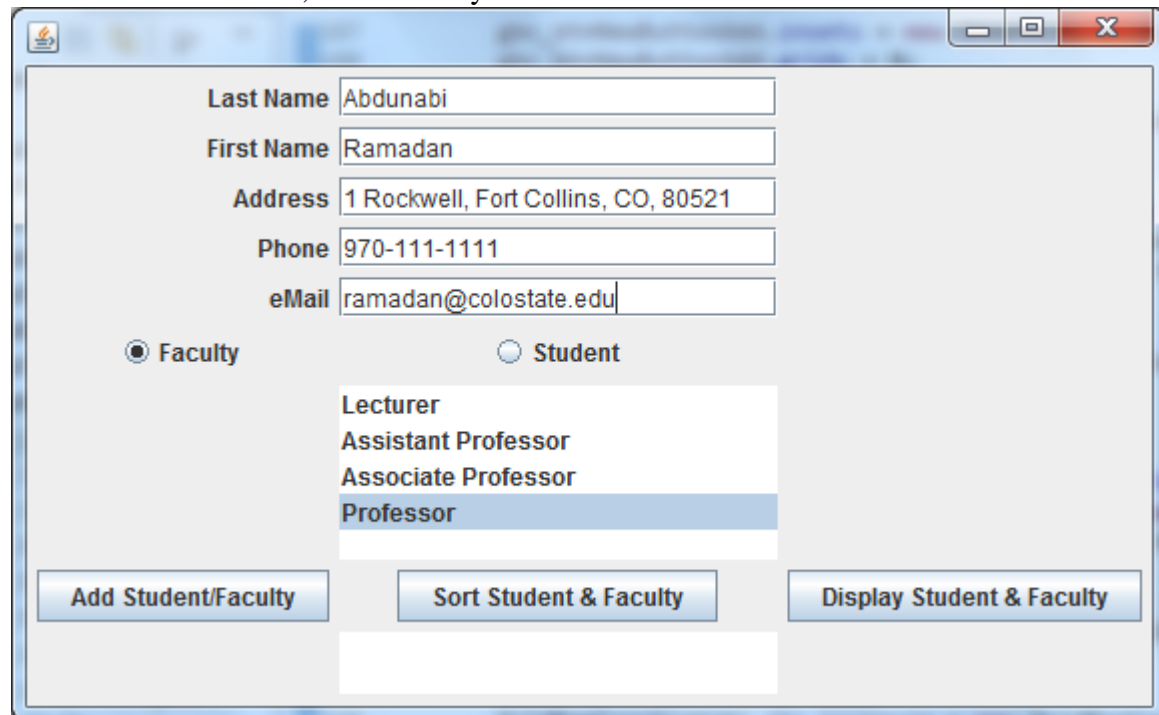
No other sequence needs to be coded, and will be tested by the grader. No coding is necessary to test whether the same lastName is entered for the faculty (or the student), and the program will not be tested by the grader on that. The grader will enter different last names to grade the assignment.

Close button. It will close the GUI/exit the program.

A user GUI must have Java graphical components, labels, text-fields, radio buttons listbox. The student coding the assignment can use the creativity to meet the GUI requirements. All data inputs must be validated against improper data type or empty inputs, all object data fields are required. The required data fields cannot be empty. No coding is needed to validate any pattern of the email, or the phone number. Also, the program must not crash from invalid inputs or terminate upon handling improper inputs, it must allow a user to re-enter the required data fields after handling input errors.

The GUI below is just an example. The design of the GUI and various prompts is left to the creativity of the students, as long as it meets the assignment requirements.

The GUI shown below, if the Faculty radio button is selected.



A Java Swing window titled "Faculty" with a standard Mac OS X title bar (red, yellow, green buttons). The window contains a form for entering faculty information. The fields are: Last Name (Abdunabi), First Name (Ramadan), Address (1 Rockwell, Fort Collins, CO, 80521), Phone (970-111-1111), and eMail (ramadan@colostate.edu). Below the fields are two radio buttons: "Faculty" (selected) and "Student". To the right of the radio buttons is a list box containing the following options: Lecturer, Assistant Professor, Associate Professor, and Professor. The "Professor" option is currently selected. At the bottom of the window are three buttons: "Add Student/Faculty", "Sort Student & Faculty", and "Display Student & Faculty".

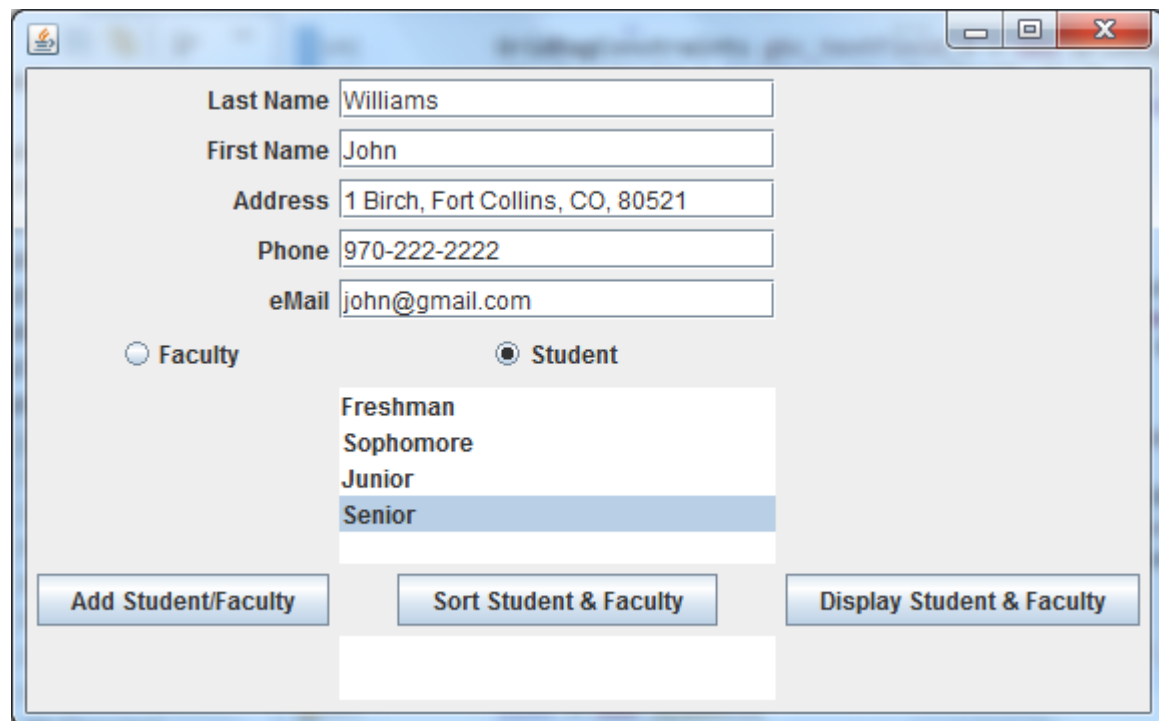
Last Name: Abdunabi
First Name: Ramadan
Address: 1 Rockwell, Fort Collins, CO, 80521
Phone: 970-111-1111
eMail: ramadan@colostate.edu

☒ Faculty ☐ Student

Lecturer
Assistant Professor
Associate Professor
Professor

Add Student/Faculty Sort Student & Faculty Display Student & Faculty

The GUI shown below, if the Student radio button is selected.



A Java Swing window titled "Student" with a standard Mac OS X title bar (red, yellow, green buttons). The window contains a form for entering student information. The fields are: Last Name (Williams), First Name (John), Address (1 Birch, Fort Collins, CO, 80521), Phone (970-222-2222), and eMail (john@gmail.com). Below the fields are two radio buttons: "Faculty" and "Student" (selected). To the right of the radio buttons is a list box containing the following options: Freshman, Sophomore, Junior, and Senior. The "Senior" option is currently selected. At the bottom of the window are three buttons: "Add Student/Faculty", "Sort Student & Faculty", and "Display Student & Faculty".

Last Name: Williams
First Name: John
Address: 1 Birch, Fort Collins, CO, 80521
Phone: 970-222-2222
eMail: john@gmail.com

☐ Faculty ☒ Student

Freshman
Sophomore
Junior
Senior

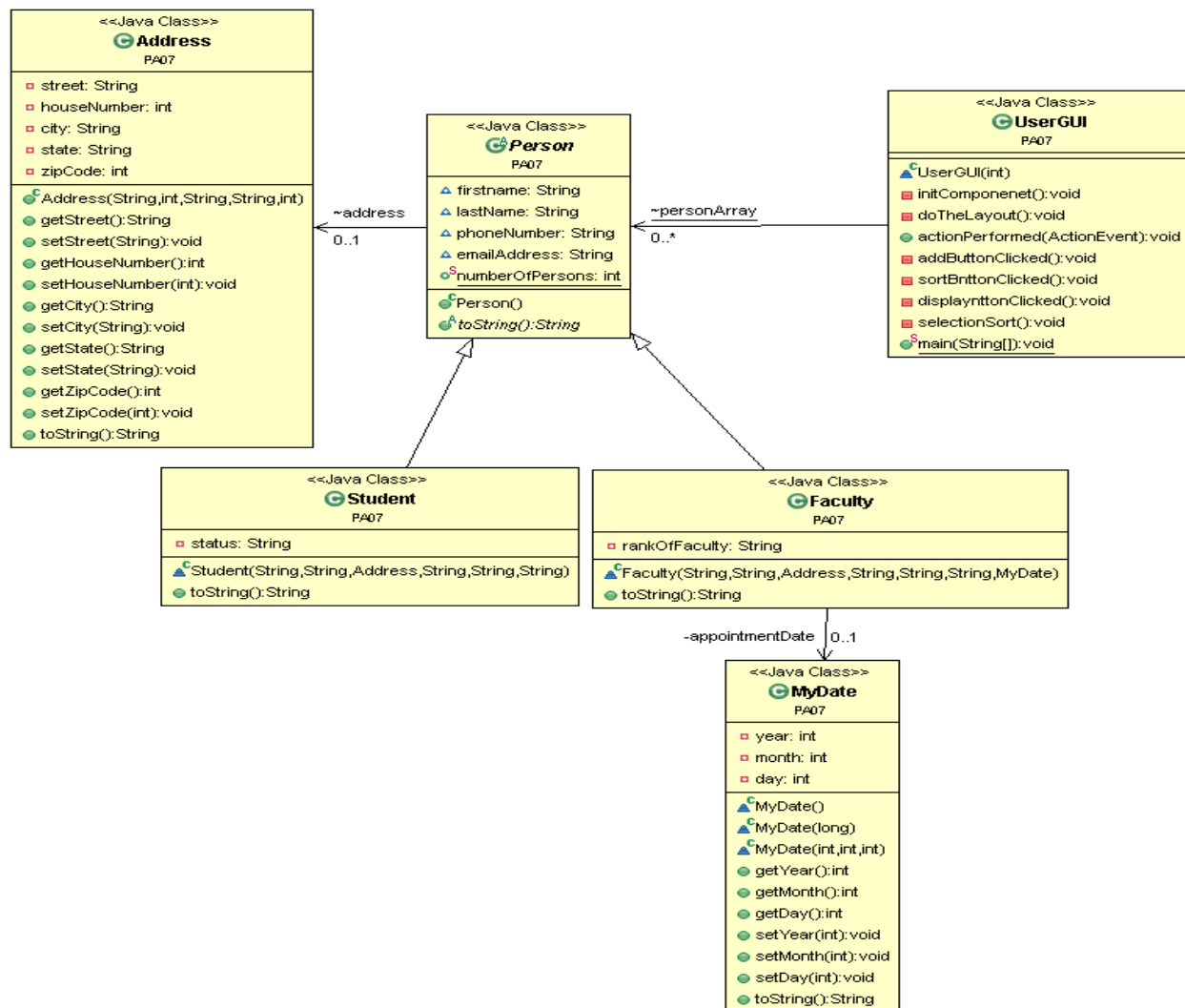
Add Student/Faculty Sort Student & Faculty Display Student & Faculty

You must register the button for Add with an implementer class, which implements the `ActionListener` interface in order to respond to a user click event. You must override the action performed method (`public void actionPerformed(ActionEvent e)`) for each class that implements `ActionListener` in order to correctly respond to the user's request to add a new object.. The code that you will write in the action performed method must create a new object and display its `String` data in the text-area component. It is optional to define one or two inner implementer `ActionListener` classes, or preferably make the `UserGUI` class implement the `ActionListener` interface.

The GUI screen shots shown are examples. You may choose your own creativity to design user GUI, such as positioning graphical components and/or enable or disable some of these components as needed at the run time, any color scheme etc. You must use sub-containers panels (the `JPanel` class) in order group and layout the graphical components in the main Java window defined by the `UserGUI` class. To have a good look and feel in user GUI, some panels must be sub-contained in another panels with the appropriate selection and use of one of the layout managers' classes (`FlowLayout`, `GridLayout`, and `BorderLayout`).

Class Diagram (same as that of PA07):

The same classes and the class diagram from PA06 are also used in this assignment, except for `UserGUI` class.



Evaluation Criteria:

- The program must compile cleanly (no compile errors, but compile warnings are sometimes accepted)
- The program must handle any invalid data inputs by users, includes required fields
- The program must not crash while running and it should terminate gracefully, use try catch blocks as applicable
- All tasks (requirements) in this assignment must be completed in order to receive credit
- The correct understanding and implementation (coding) of the requirements (programs should behave as anticipated):
 - o The program must terminate with proper/correct outputs
 - o All the logical relationship between classes should be performed correctly

Submission: (***This is an individual Assignment!***)

Copy the .java source files from the *src* folder in your *work space* to another folder that should be named following the provided naming format in this course, then zip and

upload the file under this assignment answer in Canvas.

File Name: *FLLLLPA07.zip* (*F = first letter in your first name and LLLL = your last name*)

Grading Rubric PA07

Student Name: _____

Question 1

Requirements	Comment	Max Points Allowed	Points Earned
General Code Structure: Proper naming convention used for file (0.25) Comments used in the code to explain the purpose of the code (0.25) Indentation of the code for better readability (0.25) Good choice of the variable names (0.5)		1	
Input, Output, User Interface: Proper design of the GUI components to get the number of persons (1) Proper design of the GUI components common to both faculty and student (1) Display Faculty list box with faculty info, if the Faculty radio button is selected (1) Display Student list box with student info, if the Student radio button is selected (1) Non-editable text area showing information on Faculty/Student (1)		5	
General Algorithm and Logic: Use of the Person, Student, Faculty, Address classes, and the subclass hierarchy (4)		14	

<p>Populating the personArray with the data entered in GUI, when Add button is clicked (3)</p> <p>Sorting the data in the personArray (both faculty and student) when the Sort button is clicked (4)</p> <p>Use of toString() method to display the information on Faculty and the Student in the display area, when the Display button is clicked. Put a line separator after displaying each person information (3)</p>			
Total		20	

Total ____/20