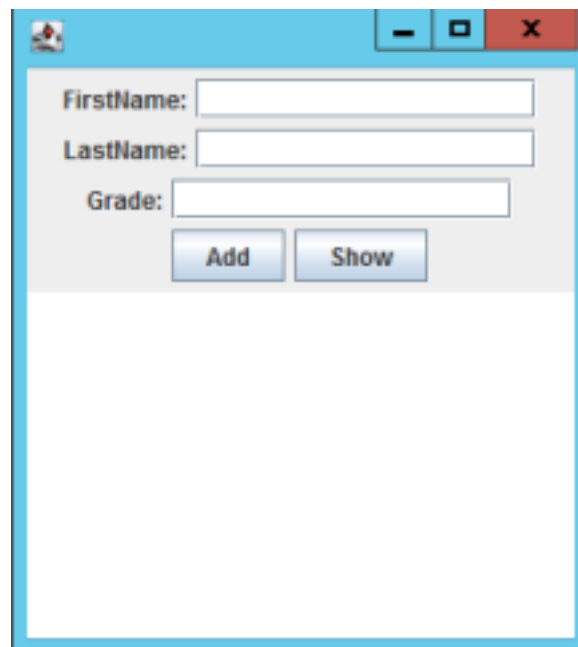


Generally:

- You will develop a GUI application that manages students
- Start with some of the code we developed in class
- Create a custom StudentException class that inherits from Exception
- Create a Student class that has private member variables of “ firstname / lastname / grade” and when validating the input values throws your own custom StudentException
- Create a StudentManager class that adds Student objects to an ArrayList
- Perform necessary “DATA VALIDATION” in all the SETTERS to not allow invalid values to exist within the objects
- Include and use the LOG4J JAR file in your project
- Setup the logging infrastructure just like we did in class (IMPORT statements and the STATIC logger object per class)
- Perform LOG4J logging to the maximum extent possible
- Your logging output text does not have to match mine – but all the other actual program console output text must match
- The JPanel will contain widgets that handle the data of the first name, last name and grade and displays messages in the JTextArea (including exceptions)
- Generally, the FRAME, PANEL and widgets will look like this (remember you can resize the frame to look like this) ::

- In the Student class ::



Specifically:

- Have all appropriate getters, setters and constructors
- Override the parent class's “toString()” method which returns a formatted string showing the internal state of the Student (see the screenshots)
- Remember – in the SETTERS – do necessary data validation to not allow invalid data to be set/exist within the object. Remember to throw the custom exception “StudentException” (see the screenshots)
- Do ERROR logging
- In the StudentException class ::
  - Inherit from the Exception class
- In the StudentManager class ::
  - have an ArrayList collection that can only contain Student objects
  - have a method that can add Student objects to the collection that has the appropriate input

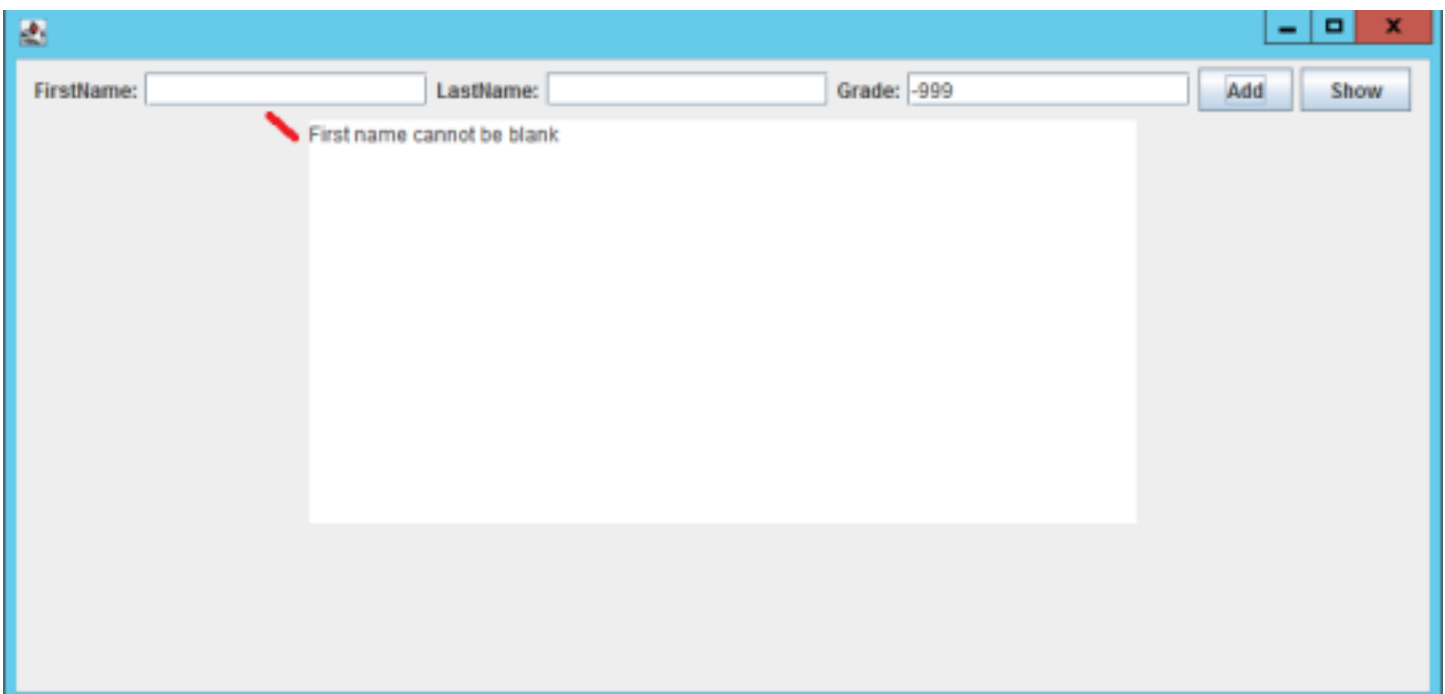
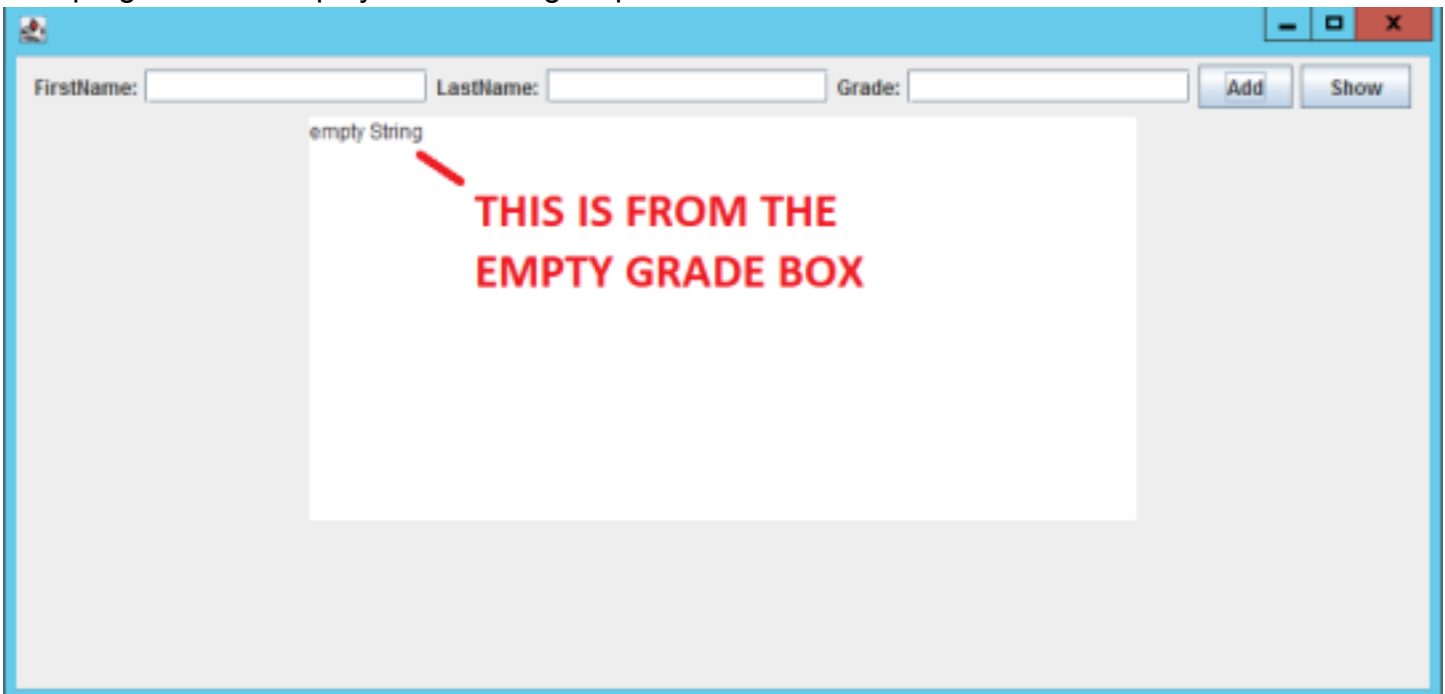
arguments firstname, lastname and grade

- have a method that can create and output/return a list of student “toString()” strings, and the average class grade (see the screenshot)
- In the “Panel” class ::
  - have all the appropriate widgets layed out in the manner like in the screenshots
  - handle all the appropriate actions in the appropriate manner
  - display all the appropriate output, including exceptions in the “text area”

In the MAIN class's MAIN method:

- Declare, instantiate and launch the Frame

Your program shall display the following output EXACTLY as shown :: Like so:



First Name:  Last Name:  Grade:

Last name cannot be blank

First Name:  Last Name:  Grade:

Grade must be positive

First Name:  Last Name:  Grade:

Grade must be  $\leq 100$

First Name:  Last Name:  Grade:

Grade must be  $\leq 100$

Student Management System

FirstName:  LastName:  Grade:

Student added successfully

Student Management System

FirstName:  LastName:  Grade:

Student added successfully

Student Management System

FirstName:  LastName:  Grade:

Student added successfully

Student Management System

FirstName:  LastName:  Grade:

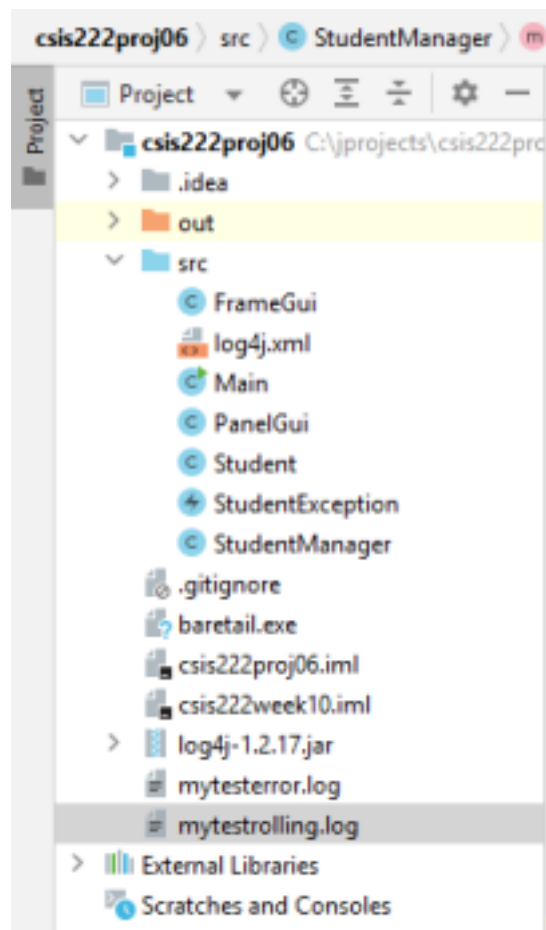
Student{firstName='Sandy', lastName='Beach', grade=98.6}  
Student{firstName='Paige', lastName='Turner', grade=91.2}  
Student{firstName='Ford', lastName='Karr', grade=89.3}

Average class grade: 93.03

Plugins supporting *.log files found.				
1	2023-03-28 20:40:37	[main]	DEBUG	Main:9 - LAUNCHING THE FRAME
2	2023-03-28 20:40:38	[main]	DEBUG	FrameGui:12 - LAUNCHING AND ADDING THE PANEL
3	2023-03-28 20:40:38	[main]	DEBUG	PanelGui:24 - instantiating the widgets
4	2023-03-28 20:40:38	[main]	DEBUG	StudentManager:11 - inside the default constructor
5	2023-03-28 20:40:38	[main]	DEBUG	PanelGui:34 - configuring the buttons
6	2023-03-28 20:40:38	[main]	DEBUG	PanelGui:44 - adding the widgets to the panel
7	2023-03-28 20:40:57	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
8	2023-03-28 20:40:57	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
9	2023-03-28 20:40:57	[AWT-EventQueue-0]	ERROR	PanelGui:107 - empty String
10	2023-03-28 20:41:04	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
11	2023-03-28 20:41:04	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
12	2023-03-28 20:41:04	[AWT-EventQueue-0]	ERROR	PanelGui:107 - empty String
13	2023-03-28 20:41:13	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
14	2023-03-28 20:41:13	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
15	2023-03-28 20:41:13	[AWT-EventQueue-0]	DEBUG	StudentManager:18 - adding a student to the list
16	2023-03-28 20:41:13	[AWT-EventQueue-0]	ERROR	PanelGui:102 - Last name cannot be blank
17	2023-03-28 20:41:22	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
18	2023-03-28 20:41:22	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
19	2023-03-28 20:41:22	[AWT-EventQueue-0]	DEBUG	StudentManager:18 - adding a student to the list
20	2023-03-28 20:41:22	[AWT-EventQueue-0]	ERROR	PanelGui:102 - First name cannot be blank
21	2023-03-28 20:41:30	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
22	2023-03-28 20:41:30	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
23	2023-03-28 20:41:30	[AWT-EventQueue-0]	DEBUG	StudentManager:18 - adding a student to the list
24	2023-03-28 20:41:30	[AWT-EventQueue-0]	ERROR	PanelGui:102 - Grade must be positive
25	2023-03-28 20:41:36	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
Plugins supporting *.log files found.				
19	2023-03-28 20:41:22	[AWT-EventQueue-0]	DEBUG	StudentManager:18 - adding a student to the list
20	2023-03-28 20:41:22	[AWT-EventQueue-0]	ERROR	PanelGui:102 - First name cannot be blank
21	2023-03-28 20:41:30	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
22	2023-03-28 20:41:30	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
23	2023-03-28 20:41:30	[AWT-EventQueue-0]	DEBUG	StudentManager:18 - adding a student to the list
24	2023-03-28 20:41:30	[AWT-EventQueue-0]	ERROR	PanelGui:102 - Grade must be positive
25	2023-03-28 20:41:36	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
26	2023-03-28 20:41:36	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
27	2023-03-28 20:41:36	[AWT-EventQueue-0]	DEBUG	StudentManager:18 - adding a student to the list
28	2023-03-28 20:41:36	[AWT-EventQueue-0]	ERROR	PanelGui:102 - Grade must be <= 100
29	2023-03-28 20:41:46	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
30	2023-03-28 20:41:46	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
31	2023-03-28 20:41:46	[AWT-EventQueue-0]	DEBUG	StudentManager:18 - adding a student to the list
32	2023-03-28 20:41:46	[AWT-EventQueue-0]	DEBUG	PanelGui:98 - Student added successfully
33	2023-03-28 20:42:05	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
34	2023-03-28 20:42:05	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
35	2023-03-28 20:42:05	[AWT-EventQueue-0]	DEBUG	StudentManager:18 - adding a student to the list
36	2023-03-28 20:42:05	[AWT-EventQueue-0]	DEBUG	PanelGui:98 - Student added successfully
37	2023-03-28 20:43:00	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
38	2023-03-28 20:43:00	[AWT-EventQueue-0]	DEBUG	PanelGui:86 - attempting to add a student to the manager
39	2023-03-28 20:43:00	[AWT-EventQueue-0]	DEBUG	StudentManager:18 - adding a student to the list
40	2023-03-28 20:43:00	[AWT-EventQueue-0]	DEBUG	PanelGui:98 - Student added successfully
41	2023-03-28 20:43:09	[AWT-EventQueue-0]	DEBUG	PanelGui:60 - handling the button clicks
42	2023-03-28 20:43:09	[AWT-EventQueue-0]	DEBUG	PanelGui:78 - about to show the students
43	2023-03-28 20:43:09	[AWT-EventQueue-0]	DEBUG	StudentManager:24 - generating and returning the student output

The program must compile, build, launch, and run correctly.

When you are done with the assignment there should be at least one self-written file(s), consisting of the source code file(s): like so ::



(along with any and all other associated IDEA files)

The final version of the program must have all the code for meeting the problem specifications.