Lab 2 [50 marks]

Goals:

* Use UML drawing tools to create UML diagrams
* Practice Generalization
* Practice designing and implementing Java programs that use
  + Inheritance
  + Method overriding
  + The Factory design pattern

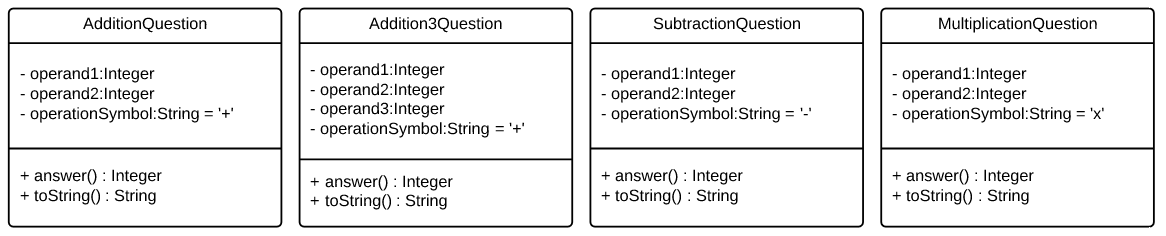
Submission: [How to Submit CSD221 Labs](https://docs.google.com/document/d/1HBW_otFOBU2tWDPexTCU8VjPpkaxyYVUXZOekEDbqhc/edit?usp=sharing); you will lose marks if you do not submit correctly.

## Overview

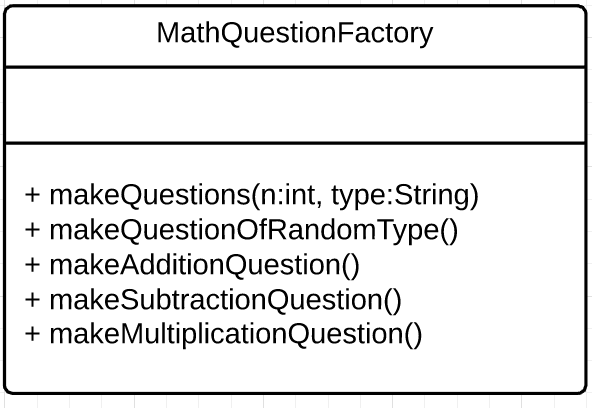
In this Lab, we will begin creating a quizzing application called ‘Quizzard’ that prompts the user for some initial setup information and then asks them a series of questions that they must answer correctly. Once they have finished answering the questions, the user will be shown a report of their results and then asked if they want to play again.

## Setup [5 marks]

* 1. Download the partial implementation for Quizzard from LMS (under the Lab 2 module), and initialize a git repository in it.
  2. Create a new Project called ‘Lab2’ in your CSD221 GitLab group and push your local repository to this project. (See [Configuring Your GitLab Account for CSD221](https://docs.google.com/document/d/1-Z3TFr1yKYlPEJ9GxWie_mQNTimWOA2KuX9NH0dBT3s/edit?usp=sharing) if you forget how to do this.)
  3. Make sure you can run the project. Familiarize yourself with the command line UI of the app. It will be worth reading through the code that is there and understanding how it is driving the current UI.

1. **Design [10 marks]**Quizzard will ask math questions of the types described in the UML diagram below  
     
   Notes:  
    - Each type of question should have a constructor that takes as its **only** parameters the operands for the question.   
    Eg. new AdditionQuestion(2,1) would create the question 2+1  
    - The answer method returns the correct answer for the question  
    - The toString() method should return a String that makes sense in code like this:  
    System.out.println("What is " + question.toString() + "?");
   1. Generalize the classes above into a class hierarchy and create a UML class diagram of your hierarchy. (You may use any drawing tool you like, such as Visio or lucidchart.com.)  
      NOTE: You may find that as you work on this Lab the design of your Question class hierarchy changes. That is fine! Just make sure to come back and update your answer here so that it matches what you have in your code.
      1. Save the UML diagram as an IMAGE or PDF file (something that I’ll for sure be able to open)
      2. It should be named **QuestionTypesClassDiagram**
      3. Save it in the src/main/resources folder of your project

## Implementation [25 marks]

* 1. [5 marks] Implement the class hierarchy you designed in question 2 above
     1. Your classes should go into the ca.saultcollege.csd221.w18.Quizzard.Questions package
  2. [12 marks] Create a MathQuestionFactory that follows the design specified by this UML class diagram:  
       
     NOTE: the methods do not specify a return type, but they SHOULD return some type of question object. The type is not specified because the name of the type will depend on the names you used in your Design step above  
     1. [2 marks] Put the factory in the same Questions package as you put the question classes above.
     2. [3 marks] The makeQuestions method takes an integer n (the number of questions to make) and a String type (the type of question to make—either ‘addition’ ‘subtraction’ ‘multiplication’ or ‘any’) and returns an **array** of questions of length n with each question in the array being an instance of one of the question classes you just implemented, corresponding to the given type. If type is ‘any’ the method should use the makeQuestionOfRandomType method to make its questions. Otherwise it should use the makeAdditionQuestion / makeSubtractionQuestion / makeMultiplicationQuestion method that corresponds with the given type.
     3. [2 marks] The makeQuestionOfRandomType method should return a **single** question that is an instance of one of your question classes. The type of question is chosen randomly by the method itself.
     4. [2 marks] The makeAdditionQuestion method should return a **single** question that is an instance of one of AdditionQuestion or Addition3Question, chosen randomly by this method.
     5. [2 marks] Each of the remaining methods should return a **single** question that is an instance of the question class referred to by the method name.
     6. [1 mark] For all the above methods, whenever the factory needs to supply an integer to a question constructor it should generate a random integer between 0 and 12 to use as the argument. (See the note on constructors in question 2 above.)
  3. [8 marks] Familiarize yourself with the code in the InputController class. This class manages all of the user input and feedback in the Quizzard app.
     1. [1 mark] Add a new method to InputController called askQuestion.
     2. [1 mark] This method should take one question object (an instance of one of your question classes above) as a parameter
     3. [2 marks] It should display that question in a meaningful way to the user, and then prompt the user for their answer.
     4. [2 marks] If the user answers incorrectly, they should be prompted repeatedly until they do finally answer correctly.
     5. [2 marks] The method must keep track of the number of tries before the user got the correct answer, **and return this number**.
     6. It is fine (and good!) to use any of the existing methods in InputController to implement the askQuestion method. Feel free to add other helper methods if you feel it is useful.
  4. [10 marks] Tie it all together!
     1. Find the ‘TODO’ comment in the play method of the QuizzardApp class.
     2. [1 mark] At this point in the app, the currentPrefs variable that you see above the TODO comment will have two public fields: numberOfQuestions (the number of questions the user wants to do) and questionType (the type of questions the user wants—one of ‘addition’ ‘subtraction’ ‘multiplication’ or ‘any’). Use the values in these fields to call the makeQuestions method on your MathQuestionFactory.
     3. [2 marks] With the array of questions that the factory returns, loop through and use the askQuestion method that you implemented above to ask the user for the answer of each question in the array.
     4. [1 mark] Store the values that askQuestion returns (the number of attempts the user made to answer the question correctly) as you loop through the questions.
     5. [2 marks] Finally, call the showReport method using the question array and the stored values you got back from the askQuestion method.  
        NOTE: You will need to uncomment the commented lines in the showReport method, and change the type of the Object[] questions parameter to be of the most general class in your question class hierarchy in order for the showReport method to function properly. Eg. If your most general class is called BaseQuestion, then the first parameter of the showReport method should be BaseQuestion[] questions instead of Object[] questions.
     6. [4 marks] You should now be able to run your project and select a number and type of questions when prompted. The app should display a set of questions corresponding to the options you selected, and when you finish them display a report of your results.

## Documentation & Formatting [10 marks]

* 1. Make sure you have documented you code well using Javadoc where appropriate
  2. Use meaningful names for classes, attributes, methods, and variables that you create

## Submit your Lab

* 1. See [How to Submit CSD221 Labs](https://docs.google.com/document/d/1HBW_otFOBU2tWDPexTCU8VjPpkaxyYVUXZOekEDbqhc/edit?usp=sharing)
  2. You will lose marks if you do not submit correctly.

## 

## Sample Quizzard Command Line Session

Enter your name:

Rodney

How many questions would you like?

Please enter a number from 1 to 20:

3

What would you like to practice?

(1) addition

(2) subtraction

(3) multiplication

(4) any

1

What is 8 + 11?

19

Correct!

What is 1 + 12?

13

Correct!

What is 7 + 11?

20

Try again...

What is 7 + 11?

19

Try again...

What is 7 + 11?

18

Correct!

Report:

1) 8 + 11 = 19 ✓

2) 1 + 12 = 13 ✓

3) 7 + 11 = 18 (3 tries)

Would you like to play again, Rodney?

(1) y

(2) n

y

Would you like to use the same number and type of questions as last time?

(1) y

(2) n

n

How many questions would you like?

Please enter a number from 1 to 20:

2

What would you like to practice?

(1) addition

(2) subtraction

(3) multiplication

(4) any

4

What is 5 - 2?

3

Correct!

What is 2 + 12 + 5?

19

Correct!

Report:

1) 5 - 2 = 3 ✓

2) 2 + 12 + 5 = 19 ✓

Would you like to play again, Rodney?

(1) y

(2) n

y

Would you like to use the same number and type of questions as last time?

(1) y

(2) n

y

What is 6 + 8?

14

Correct!

What is 2 + 4?

6

Correct!

Report:

1) 6 + 8 = 14 ✓

2) 2 + 4 = 6 ✓

Would you like to play again, Rodney?

(1) y

(2) n

n