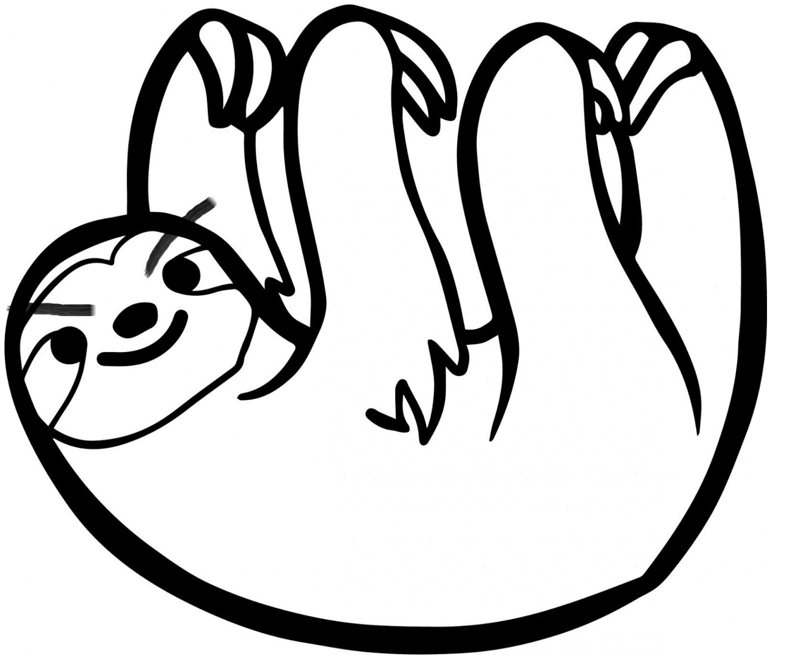
Not Fast, Just Furious



Alexander Hung, David Yau, Gavin Zhang, Jason Fong, Mathu Manogaran

Code Review

Last updated: November 20, 2017

Table of Contents

Strategy
During the code review, we should focus mainly on DOA.java since most of the backend processing is there. Furthermore, there are many things that can be changed in DOA.java to improve the design and structure of the code. We should also look over the code for what can be changed/removed/added so that we use the best practices we learned in class.
What each individual should mainly look at:

David: DOA.java code that Gavin and Mathu wrote
Gavin: DOA.java code that David and Mathu wrote
Jason: Assignment.java, Student(s).java, Question.java
Mathu: Most of the GUI code

After the code review, changes should be made during the next sprint.

Summary
Overall the code follows most principles and is readable but it is not well enough documented (there's an overall lack of comments).
There are some lines that are too long.
Student and Professor passwords should not default to password.
Some redundant or unused code still exists in the program and can be merged or deleted.
Validators can be reorganized into a new validators class rather than extending DOA.java and some validators do not exist.
DOA.java has too many try catch statements and can be redone.
Since DOA.java has to be refactored/reorganized, testing it will most likely not be possible until refactoring is finished.
Testing is also very limited due to having most of the work being done by accessing the database.
The GUI can be made more modular. Core features such as logout buttons, denying access to overdue assignments and writing marks to the database are still missing.

Code Review for commit: eb5b517a412d73df9e7c1adaad2d9e35c6b0393d on master (Nov 17, 2017 @ 3 PM)

Code Review Video Link:
https://drive.google.com/file/d/1k6gQprajwcKWcywfGKxfKZIijrB8unQm/view

List of Points Covered in Code Review

jdbc > DOA.java

- Code is cohesive (these methods do belong together for the most part)
- Database naming follows conventions
- Code is very modular for the most part
- .
- Unnecessary imports
- ~~**In general: Not enough comments**~~
- ~~**Way too many try-catch statements, should improve by throwing exceptions to the higher level modules**~~
- ~~**Need a validator to make sure no professor id is the same as a student one**~~
- ~~**The alternative solution is to merge the student and professor tables, which should be possible**~~
- ~~**When we add students to a course, we need to add values to a student assignment table so that each student in that course has initialized assignment values that we have to mark as null.[INSERT INTO OR UPDATE]**~~
- ~~**There is no statement to update the mark for an assignment.**~~
- Random block of comments at the bottom of DOA
- Currently an assignment number cannot be duplicated within the same course
- Currently a course cannot have the same course number as another course
- ~~**Need validators for manually adding students, can perhaps use some of these validators from getErrorsInStudentFile since they're basically checking the same fields**~~
- ~~**No way to support multiple choice questions**~~
- ~~**Need to create a student course table, containing only student ID and course ID.**~~
- There needs to be some way to drop all the students from the course database at the end of the term.
- Need a method that returns max number of questions in an assignment.
- 27-32: Should use 'final' keyword to declare constants
- Code exceeds recommended line limit in some cases (lines 429, 409, 350, 335, 318, 281, 265, 224 and more)
- ~~**64: deadline attribute could be DATETIME instead of DATE so that professor can also give a time deadline**~~
- ~~**127: ALL validators could be in a different class (by Single Responsibility Principle)**~~
    - ~~**Import DOA.java**~~
- ~~**118, 358 : Students and professors both have passwords defaulted to "password"**~~

        - ~~Keep for now, easier for development~~
    - ~~127: Should enforce format for course id as well~~
    - ~~405, 425: loginStudent() and loginProf() are almost identical.~~
    - ~~162: Need to add validator for unique UTORid~~
    VALIDATORS

jdbc > MySQLAccess.java

- Prints stuff out easily
- .
- Unused methods can be removed
- *Lots of try-catch*
- *Unneeded Imports*

~~**student > Student.java**~~

- ~~(good) There are lots of comments to describe action of functions~~
- ~~.~~
- ~~21: First constructor is now unnecessary as all users are required to have passwords~~

~~**student > Students.java**~~

- ~~Class is no longer used (replaced with database)~~

~~Assignments and Questions should be aligned with table~~

assignment > Assignment.java

- Need these variables and getters/ setters in class
    - Course_id, assignment_id, num_question, assignment_name, deadline
- Constructor with everything

~~**assignment > Question.java**~~

- Need these variables and getters/ setters in class
    - Course_id, assignment_id, question_id, answer_function, lower_range, upper_range, decimal_places
- Name should be changed to SingleAnswerQuestion in preparation for multiple types of questions
- Two constructors will be needed: one with ids, question, and answer. Another with everything to be used in the future
- ~~Parse questions and answer functions, randomize variables and calculate~~

~~**assignment > RandQuests.java**~~

- ~~Can be merged into Assignment.java as a static method ← this class will be removed~~

gui > IntroScreen.java

- *Unneeded import statements*

- Change login behaviour after DOA changes are made

gui > MessageBox.java, ProfessorViewStudents.java, ProfessorAddStudents.java,

gui > StudentPage.java

- **Make GUI more modular by making event handlers call separate methods.**
- Add Label displaying assignment average for each assignment (active after selecting)
- **Add Label displaying assignment deadline**
    - **Disable 'Open' button if deadline is passed**

gui > StudentAssignmentPage.java

- **Select random questions from assignment pool**
- Select only maximum number of questions from pool

GUI

- AddProfessor page
- Add ability to remove everything (students from a course,
- **Logout button for both Professor and Student perspectives**

Jason

- In general: Not enough comments

student > Student.java [David]

- (good) There are lots of comments to describe action of functions
- .
- 21: First constructor is now unnecessary as all users are required to have passwords

student > Students.java [David]

- Class is no longer used (replaced with database)

assignment > Question.java [Gavin]

- Assignments and Questions variables should be aligned with the columns in the table
- Parse questions and answer functions, randomize variables and calculate

assignment > RandQuests.java [Gavin]

- Can be merged into Assignment.java as a static method ← this class will be removed

David

- 64: deadline attribute could be DATETIME instead of DATE so that professor can also give a time deadline [Mathu]
- Need a validator to make sure no professor id is the same as a student one
- The alternative solution is to merge the student and professor tables, which should be possible
- 405, 425: loginStudent() and loginProf() are almost identical.
- There is no statement/function to update the mark for an assignment.

Gavin

- Need validators for manually adding students, can perhaps use some of these validators from getErrorsInStudentFile since they're basically checking the same fields
- There should also be a validator for enforcing course id format
- 127: ALL validators could be in a different class (by Single Responsibility Principle)
    - Import DOA.java
- 162: Need to add validator for unique UTORid

- **Way too many try-catch statements, should improve by throwing exceptions to the higher level modules**

Mathu

- **When we add students to a course, we need to add values to the student assignment table so that each student in that course is properly initialized with the mark as null and will not be factored into average mark calculations until the assignment has actually been completed. INSERT INTO OR UPDATE**
- **118, 358 : Students and professors both have passwords defaulted to “password”**
    - **Keep for now, easier for development**

gui > StudentPage.java

- **Make GUI more modular by making event handlers call separate methods.**
- **Add Label displaying assignment deadline**
    - **Disable ‘Open’ button if deadline is passed**

gui > StudentAssignmentPage.java

- **Select random questions from assignment pool**

GUI

- **Logout button for both Professor and Student perspectives**