**Instructions For Compilation:**

1- Go to the “Application.java” file.

2- Run its main method.

3- A menu will be printed for Log-In or Sign-Up.

4- If you logged in as a Teacher, you will navigate to the Teacher menu.

5- If you logged in as a Student, you will navigate to the Student menu.

6- The Teacher menu has options that will direct you to the Quiz menu.

7- The Quiz menu does all the necessary changes a Teacher wants.

**List Of Students Who Submitted:**

1- Anish submitted on Vocareum.

2- Zuhair submitted on Brightspace.

**Description of classes:**

**1-** **Application.java:**

**Functionality:** Displays a menu for sign-in and log-in. Teachers and students can sign-in and register. It will direct them to their menus depending on who logged in. The class saves the log-in information in a notepad.

**Testing:** We ran the compiler and confirmed that it assigns students and teachers and directs both of them to their respective menus.

**Relationship:** When it directs to other menus, it connects to other Teachers and Students classes.

**2-** **Course.java:**

**Functionality:** Saves course information

**Testing:** When we created a course, a teacher was assigned with the specified course and number of student enrollment.

**Relationship:** It is used by all Courses classes.

**3-** **CourseArchive.java:**

**Functionality:** Collects all the courses in an ArrayList and modifies that list.

**Testing:** It returns all the courses created when we ran the compiler. Teachers are able to get access to the courses when created.

**Relationship:** It is used by teachers and other Course classes.

**4-** **InvalidCourseException.java:**

**Functionality:** Throws an exception whenever a course can’t possibly be created.

**Testing:** It throws an exception when appropriate and indicates a descriptive error in the console.

**Relationship:** Used by all Course classes.

**5-** **InvalidQuizException.java:**

**Functionality:** Throws an exception whenever a quiz can’t possibly be created.

**Testing:** It throws an exception when appropriate and indicates a descriptive error in the console.

**Relationship:** Used by all Quiz classes.

**6-** **PrintInformation.java:**

**Functionality:** Used to write and read all quiz information from/to text files.

**Testing:** Ran the compiler and created quizzes, and they are written in a text file. Deleted the existing quizzes and then read the quizzes inside the text file to extract their information. And they were saved in the QuizArchive.java correctly.

**Relationship:** It is used by TheQuizFunction.java. Every time TheQuizFunction runs, it reads the quizzes from the text files and when it finishes modifying the quizzes, it writes them again.

**7-** **Quiz.java:**

**Functionality:** It creates a quiz object and saves its questions, studentAnswers, the correctAnswers, quiz name, calculates the score, randomizes the questions and options for the quiz, modifies the questions and options, and generates a string the contains all quiz information that are ready to be printed to a file.

**Testing:** Using the compiler, it adds questions, assigns a name for the quiz, modified the name, questions, and options upon request, randomizes the quiz questions and options upon teacher request, and the quiz information is printed appropriately.

**Relationship:** It is used by all Quiz classes.

**8-** **QuizArchive.java:**

**Functionality:** Stores all quizzes in one ArrayList.

**Testing:** It retrieves the needed quizzes upon request.

**Relationship**: Used by all Quiz classes.

**9-** **TheQuizFunction.java:**

**Functionality:** Displays a menu that enables Teachers to create a quiz, modify a quiz, launch a quiz, randomize a quiz, and view student submissions. It includes a “startQuiz” method, but it is only used for Students.

**Testing:** Tested the menu in the compiler, it creates a quiz and store it in QuizArchive, modify a quiz from the Quiz class, launched quizzes and made it visible for students, randomized all questions and options appropriately, printed student submissions, and the startQuiz method starts the quiz appropriately.

**Relationship:** It uses Quiz.java, QuizArchive.java, PrintInformation.java, InvalidQuizException.java. Teachers, TheCourseFunction.java, and the main Application uses this class to navigate to its menu.

**10- QuizGrading.java:**

**Functionality:** Method for the teacher class that allowed a teacher to delete a quiz or view a student’s quiz.

**Testing:** Displayed and deleted a quiz when requested.

**Relationship:** It is used in all Quiz classes, Teacher.java, and is called upon in the menu when the teacher wants to delete a quiz created by accident.

**11-** **Student.java:**

**Functionality:** A user is able to create, modify or delete a student account and all the information of all registered accounts is saved in a txt file (allows user to log back in if program is closed but account is already registered)

**Testing:** Created a main method with just this implementation of code where we tried logging in, creating student accounts and then modified and deleted them while checking the txt files to make sure they were updated

**Relationship:** Displayed and used in application.java

**12-** **StudentAnish.java:**

**Functionality:** Displays a menu that only students see. It gives the students the option to delete the account or change your username and password. It also allows a student to access a specific course and take quizzes in specific courses. After they take quizzes, they can view the grade they got for the quiz. It includes the “startQuiz” method, “viewQuiz” method, “getScore” method, and “getModifiedScore” method.

**Testing:** Tested the menu in the main method. It looped perfectly when students entered an invalid option. Made sure the startQuiz method worked and printed the timestamp after a student completed a quiz. I made sure that if there were no available courses, no quizzes, or no quiz grades, the student would be notified and returned to the initial menu. I made sure my getScore method and getModifiedScore method returned the correct values for each specific student, and also accounted if the student wanted to see the correct answers to a quiz and the point values of each question on that quiz.

**Relationship:** Used when someone logs in as a student and when the student wants to take a quiz, access course, or view a quiz grade. This class is used for the student to navigate through menus and also makes sure any quiz answers are saved and sent to Student.java. It is interconnected with the Teacher.java, TheQuizFunction.java, and Quiz.java, CourseArchive.java, and especially Student.java.

**13-** **Teacher.java:**

**Functionality:** A user is able to create, modify or delete a teacher account and all the information of all registered accounts is saved in a txt file (allows user to log back in if program is closed but account is already registered)

**Testing:** Created a main method with just this implementation of code where we tried logging in, creating teacher accounts and then modified and deleted them while checking the txt files to make sure they were updated

**Relationship:** Displayed and used in application.java

**14-** **TheCourseFunction.java:**

**Functionality:** Displays a menu for courses, which enable you to: create a course, navigate TheQuizFunction.java menu, add students to a course, modifies course information like its name, enrollment, or assigned teachers to the course, and deletes a course.

**Testing:** The compiler used the menu, and each option works as specified. It created a course, modified a course, added students to a course, assigned teachers to their courses, deleted courses, and navigated TheQuizFunction.java menu successfully.

**Relationship:** It uses Course.java, CourseArchive.java, InvalidCourseException.java, and Teacher and Student classes. Teachers, and the main Application uses this class to navigate to its menu.

**15-** **TimeStamp.java:**

**Functionality:** Whenever a student has fully completed a quiz, it will print the timestamp of when it was completed and store it in an ArrayList. Teachers can access a timestamp for a specific student to see when the quiz was completed. Also, when students view the grade of a quiz, they will also see the time they submitted it (timestamp).

**Testing:** Everytime a student completed a quiz, the timestamp was printed in the terminal and we made sure the teacher could access this timestamp for a specific quiz for a specific student. Additionally, it was working properly as when a student wanted to view the grade for the quiz, it would print the amount of questions correct out of the total, the grade (the questions with assigned point values out of the total points), and the timestamp of when the quiz was completed!

**Relationship:** It is used in the startAQuiz method and is also called upon when a student or teacher uses the viewGrade method as it will show the grade of the quiz and time when the last question was completed.