Who Submitted What:

- Vocareum: Anushka Gupta
- Report: Luke Cooley

Instructions on How to Run your Project:

Our program comes preloaded with a teacher, a student, a test course, an example submission, an example answer file, and two test quizzes. The information for logging in with the teacher and student are included in the instructions. When you need to select a course or quiz, it will inform you of the options. Examples for file format are included below the instructions.

1. Run the main method in the session.java class
2. Login or create a user
    a. If you want to login:
        i. Input: 1
        ii. If you want to login as a student use these credentials:
            1. Name: Test Teacher
            2. Username: teach1
            3. Password: password
            4. Teacher or Student: 1
        iii. If you want to login as a teacher use these credentials:
            1. Name: Test Student
            2. Username: stud1
            3. Password: password
            4. Teacher or Student: 0
    b. If you want to create a user:
        i. Input: 2
        ii. Enter name: Enter your name
        iii. Enter username: Enter your username
        iv. Enter password: Enter your password
        v. Enter teacher or student: 0 for teacher or 1 for student
    c. If you want to quit:
        i. Input: 3
3. If you are a teacher continue here:
    a. If you want to create a course:
        i. Input: 1
        ii. Enter the name of the course: input a name for the course
    b. If you want to modify a course:
        i. If you want to create a quiz:
            1. Enter the course name: input the course you want to modify
            2. If you want to create quiz manually: Input 1

a. Enter a name for the quiz: input a name for your quiz
b. Enter a question type:
    i. Enter MC for Multiple Choice
    ii. Enter TF for True False
    iii. Enter FIB for fill in the blank
c. Enter the question: input your question
d. If a Multiple Choice Question:
    i. Enter Option A: input the option A
    ii. Enter Option B: input the option B
    iii. Enter Option C: input the option C
    iv. Enter Option D: input the option D
e. Enter the correct answer: Input the correct answer
f. Would you like to enter another question:
    i. If yes: input Y
        1. Return to top of modify course
    ii. If no: input N
        1. Return to top of modify course

3. If you want to import quiz from a file: Input 2
    a. Enter the filename: Enter the name of the file (Example: filename.txt)
        i. Example file is below these instructions
    b. Return to top of modify course

ii. If you want to edit a quiz:
1. Input: 2
2. Enter a quiz name: input a quiz name from the list provided
3. Enter the question number: Input the question number you want to change
4. Enter a question type:
    a. Enter MC for Multiple Choice
    b. Enter TF for True False
    c. Enter FIB for fill in the blank
5. Enter the question: input your question
6. If a Multiple Choice Question:
    a. Enter Option A: input the option A
    b. Enter Option B: input the option B
    c. Enter Option C: input the option C
    d. Enter Option D: input the option D
7. Enter the correct answer: Input the correct answer
8. Return to top of modify course

iii. If you want to delete a quiz:

1. Input: 3
2. Enter a quiz name: input a quiz name from the list provided
3. Return to top of modify course

iv. If you want to view/grade:
1. Input: 4
2. Enter a quiz name: input a quiz name from the list provided
3. Enter the student username: Enter the username of the student you want to view
4. Would you like to modify the student's grade:
    a. If yes: Input 1
        i. Enter student's updated grade:
            1. Enter a double value for the new grade based on the submission you saw
            2. Return to top of modify course
    b. If no: input 2
        i. Return to top of modify course

v. If you want to exit:
1. Input: 5
2. Return to top of step 3

4. If you are a student continue here
    a. If you want to select a course: Input 1
        i. Enter the course name: input a course name from the course list
        ii. Enter a quiz name: input a quiz name from the list printed
        iii. Would you like to start the quiz?
            1. If yes: input 1
                a. For each question:
                    i. Do you want to input a file for the answer:
                        1. If yes: Input 1
                            a. Input the filename that contains your answer: input filename (Example: AnswerFile.txt)
                                i. Example format of file below instructions
                        2. If no: input 0
                            a. Enter your answer below: input your answer
                        3. This continues until the quiz ends
                b. Return to select a course
            2. If no: input 2
                a. Return to select a course

b. If you want to logout: Input 2
    i. Return to step 2

Files and their formats:

Quiz Input File Format: Example in Repository Filename = "QuizInputExample.txt"

quiz1 //(this is the quiz name)
--Question Start--
MC //(question type)
mcanswer //correct answer
mcquestion // the question
optA // option A
optB // option B
optC // option C
optD // option D
--Question Start--
TF
tfquiz1correctanswer
tfquiz1question
--Question Start--
FIB
fibcorrectanswer
fibquestion
--Quiz End--
--Quiz Start--
quiz2
--Question Start--
TF
tfquiz2correctanswer
tfquiz2question
--Quiz End--

Answer File Format: Example in Repository Filename = "AnswerExample.txt"

A // the correct answer

Submission List File Format: Example in Repository Filename = "SubmissionList.txt"

- This is one file that guarantees that the data persists (DO NOT EDIT)

stud1Example Quiz 1.txt
stud1Midterm1.txt

Course List File Format: Example in Repository Filename = “CourseList.txt”

- This is one file that guarantees that the data persists (DO NOT EDIT)

Example Course
CS18000

User_info.txt File Format: Example in Repository Filename - “User_info.txt”

- This is one file that guarantees that the data persists (DO NOT EDIT)

Test Teacher,teach1,password,true
Test Student,stud1,password,false

This begins the description and methods of each class:

Course.java

Description:

This class is a representation of a course. It includes an array list of quizzes and a name. It is the center of the building and editing of a lot of courses and quiz methods.

Relationship to other classes

A course is made up of an array list of quizzes, which in turn are made up of an array list of questions. The session.java main method implements the course class.

Testing:

Testing of this class is included in TestCourse.java and in SessionTest.java

Fields

Field Name	Type	Access Modifier	Other modifiers	Description
name	String	private		The name of the course and course.txt file.
course	ArrayList <Quiz>	private		An arraylist of quizzes making up the course
quizStartString	String	private	static final	String used for course.txt file formatting
questionStartString	String	private	static final	String used for course.txt file formatting
quizEndString	String	private	static final	String used for course.txt file formatting
courseEndString	String	private	static final	String used for course.txt file

				formatting

Constructors

Parameters	**Access Modifiers**
String name*	public
String name, ArrayList<Quiz> quizzes	public

*This constructor reads from a course.txt file to create the course, using the name to identify the file.

Methods

Method Name	**Return Type**	**Access Modifier**	**Input Parameters**	**Description**
getName	String	public		Returns name of course
setName	void	public	String name	Sets parameter as name of course
getCourse	ArrayList<Quiz>	public		Returns ArrayList of course's quizzes.
setCourse	void	public	ArrayList<Quiz> course	Sets parameter as ArrayList of course's quizzes
saveCourse	void	public		Creates a file which saves a course and all of its information according to a specific format
removeQuiz	boolean	public	String quizName	Removes the quiz of specified name from the course

Question.java

Description:

The class is a singular instance of a question. It included the question itself, the type of question, the correct answer, and the options. It allows many other classes to work and provides a solid basis of a wide variety of information. The toString() method of the Question class is the groundwork for printing and taking a quiz.

Relationship to other classes:

This class is the basis of almost everything in this assignment. The question class allows the building of the quiz (an array list of questions) in Quiz.java. A course (Course.java) is made up of one or more quizzes. All of these are implemented and used in the Session.java class.

Testing:

Testing of this class is included in TestQuestion.java. This tests the methods of this class. I also used a separate main method to guarantee that everyone worked properly in this class. The testing of the Session.java class also tests many different aspects of Question.java.

Fields

Field Name	Type	Access Modifier	Description
type	String		The type of question (MC, TF, or FIB)
correctAnswer	String		The correct answer
question	String		The question or statement
optionA	String		If multiple choice, option A. Else, null
optionB	String		If multiple choice, option B. Else, null
optionC	String		If multiple choice, option C. Else, null
optionD	String		If multiple choice, option D. Else, null

Constructors

Parameters	Access Modifier(s)
String type, String correctAnswer, String question, String optionA, String optionB, String optionC, String optionD	public

Methods

Method Name	Return Type	Access Modifier	Input Parameters	Description
getType()	String	public	None	Returns the type of the question
setType()	void	public	String type	Set the type to the new string you input
getCorrectAnswer()	String	public	None	Returns the correct answer of the question
setCorrectAnswer()	void	public	String correctAnswer	Set the correct answer to the new string you input
getQuestion()	String	public	None	Returns the question/statement of the question
setQuestion()	void	public	String Question	Set the question/statement to the new string you input
getOptionA()	String	public	None	Returns the option A of the question
setOptionA()	void	public	String optionA	Set the option A to the new string you input
getOptionB()	String	public	None	Returns the option B of the question
setOptionB()	void	public	String optionB	Set the option B to the new string you input
getOptionC()	String	public	None	Returns the option C of the question
setOptionC()	void	public	String optionC	Set the option C to the new string you input
getOptionD()	String	public	None	Returns the option D of the question
setOptionD()	void	public	String optionD	Set the option D to the new string you input
toString()	String	public	None	Return a string of the following format (used for printing):

				If MC: question + "\n" + "A. " + optionA + "\n" + "B. " + optionB + "\n" + "C. " + optionC + "\n" + "D. " + optionD + "\n"; If TF: question + "\n"; If FIB: question + "\n";
toString2()	String	public	None	Return a string of the following format (used for reading and writing a file): If MC: type + "\n" + correctAnswer + "\n" + question + "\n" + "A. " + optionA + "\n" + "B. " + optionB + "\n" + "C. " + optionC + "\n" + "D. " + optionD + "\n"; If TF: type + "\n" + correctAnswer + "\n" + question + "\n"; If FIB: type + "\n" + correctAnswer + "\n" + question + "\n";

Quiz.java

Description:

The class is a singular instance of a quiz. It is made up of an array list of questions and a name. This class however does much more than this. This class allows you to modify a quiz, read submissions, and save submissions.

Relationship to other classes:

Quiz is made up of an array list of one or more questions (Question.java). A course (Course.java) is made up of one or more quizzes. All of these are implemented and used in the Session.java class. Many of the methods below are used in the Session.java class for final implementation and printing purposes as well.

Testing:

Testing of this class is included in TestQuestion.java. This tests the methods of this class. I also used a separate main method to guarantee that everyone worked properly in this class. The testing of the Session.java class also tests many different aspects of Question.java.

Fields

Field Name	Type	Access Modifier	Description
quiz	ArrayList<Question>		An array list of questions that make up the quiz
name	String		The name of the quiz for a variety of internal purposes and identification

Constructors

Parameters	Access Modifier(s)
ArrayList<Question> quiz	public

Method Name	Return Type	Access Modifier	Input Parameters	Description
getQuiz()	ArrayList<Question> quiz	public	None	Returns the array list that represents the quiz
setQuiz()	void	public	ArrayList<Question> quiz	Set thequiz to the new array list you input
getName()	String	public	None	Returns the string of the name of the quiz
setName()	void	public	String name	Sets the name of the quiz to the input
saveSubmission()	void	public	String username, ArrayList<String> answers,	Saves the submission of the current quiz with the answers in the array list, the username, and the quizTime. It allows for the teacher to look back at

			double quizTime	submissions and writes the files of submissions.
readSubmission()	public	String	filename	Reads the submission given as the filename input. It returns the string representation of it in response to the teacher requesting to see it.
modifyQuiz()	void	public	int questionNumber, String type, String correctAnswer, String question, String optionA, String optionB, String optionC, String optionD	Sets a question object to the specified parameters. Allows quiz modification.
toString()	String	public	none	Returns a string representation of a quiz. It calls the question toString() method to do so.
modifyGrade()	void	public	double gradeValue, String fileName	Updates the grade of the student with the parameter gradeValue within the submission file given by fileName

Session.java

Description: This class contains all the information necessary to create a session that contains the current user and a list of courses that the user can interact with. It also contains a main method to guide the user through logging in, viewing and editing courses, and taking / creating quizzes.

Relationships to other classes: Since this class contains the main method, it interacts with almost every other class. It uses the User.java class to help store user data and allow the user to login. Then, it uses the Course, Quiz, and Question class to help keep track of the user's actions when it comes to modifying courses and taking quizzes.

Testing: The testing of this class is handled via the main method included in Session.java along with TestSession.java. A lot of the output can be manually tested by simply using the program. However, we feel that this isn't sufficient so we included a separate class, TestSession.java to provide test cases for the class.

Fields

Field Name	**Type**	**Access Modifier**	**Description**
user	User	private	The user in the session
courses	ArrayList<Courses>	private	ArrayList containing all available courses
currentCourse	Course	private	The current course in the session
currentQuiz	Quiz	private	The current quiz in the session

Constructors

Parameters	**Access Modifier(s)**	**Description**
User user	public	Instantiates the user field with the given parameter, sets the value of courses by calling readCourseListFile() method, sets Current Course to null

Methods

Method Name	**Return**	**Access**	**Input**	**Description**

	Type	**Modifier**	**Parameters**	
getCourses()	ArrayList<Course>	public	None	Returns the array list containing all available courses
getUser()	User	public	None	Returns the current user of the session
getCurrentCourse()	Course	public	None	Returns the current course
getCurrentQuiz()	Quiz	public	None	Returns the current quiz
		public	ArrayList<Course> courses	Updates the courses field with a new array list of available courses.
setUser()	void	public	User user	Updates the user field with the given parameter
setCurrentCourse()	void	public	Course currentCourse	Updates the currentCourse field with the given parameter
setCurrentQuiz()	void	public	Quiz currentQuiz	Updates the currentQuiz field with the given parameter
readCourseListFile()	ArrayList<Course>	public	None	Reads CourseList.txt and creates a list of Course names. The method uses this list of course names to construct an array list of Course objects read from a separate file corresponding to each Course in the list of course names. The courses are constructed using the course constructor and passing each course name from the list
selectCourse()	void	public	Scanner scanner	Allows the user to select an available course and updates the currentCourse field with the appropriate course

				Lists all the course names and asks the user to input the name of the course they would like to select (ignoring caps), uses input validation
selectQuiz()	void	public	Scanner scanner	Allows the user to select a quiz from a list of quizzes within a course. Updates the currentQuiz field with the selected quiz List all the quiz names and asks the user to input the quiz they would like to select (ignoring caps), uses input validation
constructQuiz()	void	public	Scanner scanner	Allows the user to either manually or automatically create a quiz. Gives the user the option to either create the quiz manually or import the quiz from a file. If the user decides to create the quiz manually, the method will allow the user to input text for the quiz name, then calls the createQuestion() method to create questions for the quiz. The user can continue to add as many questions as they want until they decide to stop. The current quiz field is updated with the newly created quiz If the user decides to import the quiz from a file, they are prompted for a file name and the method importQuizFromFile() will be called
importQuizFromFile()	void	public	String fileName	Reads quiz data from files in the same format as Course files. Each line represents an element

				of the question, and questions are delimited by –Question Start–, the end of the quiz is signified by –Quiz End–. The first line of the .txt file is the quiz name
createQuestion()	Question	public	Scanner scanner	Asks the user to input text for the question, question type, correct answer, and if the question type is multiple choice, the user may input 4 options. Creates and returns a new Question object with the input
modifyQuestion()	void	public	Scanner scanner	Asks the user to input the number of the question they would like to modify. Uses input validation based on the number of questions that exist in the given quiz. After selecting a valid question, the createQuestion() method is called and the given question is replaced by the newly created question by calling the modifyQuiz() method
takeQuiz()	void	public	Scanner scanner	Allows a student to take the selected quiz. Outputs all questions in the currentQuiz field and records the students answers in an ArrayList<String>. Also records the time spent taking the quiz Saves the submission by calling the Quiz.saveSubmission() method
createNewUser()	User	public	Scanner scanner, ArrayList<User>	Prompts the user to create new Login information.

			userList	Searches the current list of users in the system to check if the information entered is unique or not. If the new user info is unique, it will return a new user. If the info is a duplicate, then the user is offered more chances to input different information.

User.java

Description:

The class is a singular instance of a user. It includes the user's name, user's username, and user password, and checks if the user is a student or a teacher. It allows many other classes to work and provides a solid basis for a wide variety of information.

Relationship to other classes:

This class is heavily in the Session class in implementing all the features surrounding the user.

Testing:

Testing of this class is included in TestUsers.java. This is a class that contains the main method which will allow the user to log in as either a teacher or student. Teachers will be able to share and edit quizzes through the terminal, and students will be able to access and take quizzes.

Fields

Field Name	Type	Access Modifier	Description
name	String	private	The type of question (MC, TF, or FIB)
username	String	private	The correct answer
isTeacher	boolean	private	The question or statement
password	String	private	If multiple choice, option A. Else, null

Constructors

Parameters	Access Modifier(s)
String name, String username, String password, boolean isTeacher	public

Methods

Method Name	Return Type	Access Modifier	Input Parameters	Description
getName()	String	public	None	Returns the name of the user
setName()	void	public	String name	Set the name of the user
getUsername()	String	public	None	Returns the username of the user
setUsername()	void	public	String username	Set the username of the user
getPassword()	String	public	None	Returns the password of the user
setPassword()	void	public	String password	Set the password of the user
getIsTeacher()	boolean	public	None	Returns the boolean for if the user is teacher
checkUserExists()	boolean	public	None	Returns true if the user already exists and catches any IOException.
createNewUser()	void	public	None	Creates a new user and catches any IOException while reading from the UserList.txt file.
writeUserData()	void	public	None	Writes user data using a print writer.
getUserID()	String	public	None	Makes user ID based on the variables in the class
deleteUserData()	void	public	None	Modify the user information based on the details and new input

Our Test Case Files:
SessionTest.java

- This is the most complete and necessary test class. It tests the session class, which in turn tests almost every method we have created. All of the classes we have are tested by this class. It is the largest test of our quiz project.

TestingQuestion.java

- This class is a main method that tests the question class and the methods. It ensures that the methods work.

TestingQuiz.java

- This class has a main method that tests the quiz class and the methods. It ensures that the methods work and prints errors if there are any.

TestCourse.java

- This class has a main method that tests the course class and the methods. It ensures that the methods work and prints errors if there are any. It tests the following:
    - course constructor from file name
    - quiz remover
    - course saver to file

TestSessionMainClass

- Code that tests to make sure data persists even if you close data