GROUP 10: MILESTONE 3

Detailed Class List

CampusCentral

Group Members:

Name: Karanvir Basson
Student ID: 1129746
Email: kbasson@uoguelph.ca

Name: Andrew Chow
Student ID: 1088114
Email: achow04@uoguelph.ca

Name: Fee Kim Ah-Poa
Student ID: 1159794
Email: fahpoa@uoguelph.ca

Name: Mazen Bahgat
Student ID: 1157821
Email: mbahgat@uoguelph.ca

Name: Muhammad Talha Sadaqat
Student ID: 1145327
Email: msadaqat@uoguelph.ca

Name: Hadi Rana
Student ID: 1107555
Email: hrana04@uoguelph.ca

Name: Joshua Jones
Student ID: 1125364
Email: jjones21@uoguelph.ca

Name: Cavaari Taylor
Student ID: 1159034
Email: cavaari@uoguelph.ca

Table of Contents

Class **Main**

Constructors
`public Main()`

Methods	Descriptions
`public static void main(String[] args)`	The main program responsible for setting up the program.

Class **TextUI**

Constructors
public TextUI()

Methods	Descriptions
public void printAddPlannedCoursePrompt()	Prints a prompt that allows the student to add a planned course to their schedule.
public void printDropRegisteredCourseStudent()	Prints a prompt that allows the student to drop a registered course from their schedule.
public void printEnterStudentToBecomeTAPrompt()	This method prints a prompt message for the user to enter the ID of the student they want to make a TA for a course.
public void printInvalidCourse()	Prints an error message if the user enters an invalid courseCode.
public void printInvalidSection()	Prints an error message if the user enters an invalid section code.
public void printInvalidStudentID()	This method prints an error message indicating that the student ID entered by the user doesn't exist in the class.
public void printLoginPrompt(String currentSemester)	Prints the login prompt for the user, including the current semester and options to log in as a student or professor.
public void printMenuProfessor()	This method prints the menu options available for the Professor user.
public void printMenuStudent()	Prints the menu options for a Student user, including options to add, remove, register, and drop courses from their schedule, as well as view their current, past, and future courses, and available TA positions.
public void printRemovePlannedCourseStudent()	Prints a prompt that allows the student to remove a planned course from their schedule.
public void printRegisterPlannedCourseStudent()	Prints a prompt that allows the student to register for a planned course on their schedule.
public void printReviewTAApplicationsPrompt(Professor professor, String currentSemester)	Prints a prompt to review TA applications for a particular course in the current semester.
public void printSectionPrompt()	Prints a prompt that allows the student to input a chosen section, when planning a new course.
public void printTAPositions(CourseCatalog courseCatalog)	Prints the TA positions for each course in the given course catalog.
public void printViewClassListPrompt()	Prints a prompt that allows the professor to choose a course to view their classlist.

public void **printWelcomeMessage**(AcademicAssociate academicAssociate)	Displays a welcome message to the user with information about the current user and the application.
public void **startUI**(CourseCatalog courseCatalog)	This method is responsible for starting the User Interface (UI) for the course catalog system.
public void **viewClassList**(CourseCatalog courseCatalog, String fullCourseCode)	Prints the class list of a specific course offering in the CourseCatalog.
public void **viewCurrentCourses**(AcademicAssociate academicAssociate, String currentSemester)	Displays the current courses for the given academic associate in the current semester.
public void **viewFutureCourses**(AcademicAssociate academicAssociate, String currentSemester)	Displays the future courses for the given academic associate.
public void **viewPastCourses**(AcademicAssociate academicAssociate, String currentSemester)	Displays the past courses for the given academic associate.
private void **printInvalidUserRole**()	Prints a message to the console indicating that an invalid user role has been entered and prompts the user to enter either 'S' to log in as a student or 'P' to log in as a professor.
private String **getCurrentSemester**()	This method returns the current semester based on the current date.

Class **AcademicAssociate**

Instance Variables
`private String firstName;`
`private String middleName;`
`private String lastName;`
`private String email;`
`private int id;`
`private ArrayList<CourseAttempt> courses;`

Constructors
`public AcademicAssociate(CourseCatalog courseCatalog)`
`public AcademicAssociate(String firstName, String middleName, String lastName, String email, int id, ArrayList<CourseAttempt> courses)`

Methods	Descriptions
`public boolean courseExists(CourseAttempt courseAttempt)`	Returns a boolean value indicating if the course attempt already exists in the students schedule.
`public ArrayList<Student> generateTAs(CourseCatalog courseCatalog, int numTas)`	Returns a list of student objects who will be teaching assistants for the courses present in the course catalog.
`public ArrayList<CourseAttempt> getCourses()`	This method returns an array list of course attempts a student has taken before.
`public String getEmail()`	Returns the email attribute of a student.
`public String getFirstName()`	Returns the first name attribute of a student.
`public String getFullName()`	Returns the full name of a student.
`public ArrayList<CourseAttempt> getHardCodedCourses(CourseCatalog)`	Returns an array of hard coded courses for demo purposes that is assigned to the student.
`public boolean getHasPHD()`	Returns a boolean value representing if the academic associate has a PHD or not.
`public int getID()`	Returns the ID number attribute of a student.
`public String getLastName()`	Returns the last name attribute of a student.
`public String getMiddleName()`	Returns the middle name attribute of a student.
`public String toString()`	Returns a String for an academicAssociate.

Class **Professor**

Instance Variables
`private boolean hasPHD;`

Constructors
`public Professor(String firstName, String middleName,String lastName, String email, int id, ArrayList<CourseAttempt> courses, boolean hasPHD)`
`public Professor(AcademicAssociate academicAssociate, boolean hasPHD)`

Methods	Descriptions
`public boolean approveTAApplicant(CourseOffering courseOffering, Student newTA)`	Approves a student as a teaching assistant for a course offering, removing their application from the list of teaching assistant applications and adding them to the list of teaching assistants.
`public boolean getHasPHD()`	Returns whether the professor has a PhD.

Class **Student**

Instance Variables
`private String degree;`
`private String major;`
`private String minor;`

Constructors
`public Student(AcademicAssociate academicAssociate, String degree, String major, String minor)`
`public Student(CourseCatalog courseCatalog,AcademicAssociate academicAssociate)`

Methods	Descriptions
`public boolean dropCourse(String fullCourseCode, CourseCatalog courseCatalog)`	This method attempts to drop a course with the specified fullCourseCode and courseCatalog.
`public String getDegree()`	Returns the degree of the Student object.
`public String getMajor()`	Returns the major of the student.
`public String getMinor()`	Returns the minor of the student.
`public boolean planCourse(CourseAttempt courseAttempt)`	Plans a course attempt for the student.
`public boolean registerCourse(String fullCourseCode, CourseCatalog courseCatalog)`	Registers a student for a course by updating the corresponding course attempt's status to "Registered".
`public boolean removePlannedCourse(String fullCourseCode, CourseCatalog courseCatalog)`	Removes a planned course from the student's list of courses if it exists.
`public boolean dropCourse(String fullCourseCode, CourseCatalog courseCatalog)`	This method attempts to drop a course with the specified fullCourseCode and courseCatalog.
`public String toString()`	Returns a string representation of the Student object, including its degree, major, and minor.

Class **Course**

Instance Variables
`private String courseID;`
`private int courseCode;`
`private String courseName;`

Constructors
`public Course(String courseID, int courseCode, String courseName)`
`public Course(Course course)`

Methods	Descriptions
`public boolean equals(Course course)`	Returns true if the current course object equals the passed in course object, according to the internally specified conditions for equality.
`public boolean equals(CourseCatalog courseCatalog, String fullCourseCode)`	Returns true if the courseCode of the current course object equals the passed in courseCode.
`public int getCourseCode()`	Returns the course code for a specific course.
`public String getCourseID()`	Returns the course ID for a specific course.
`public String getCourseName()`	Returns the course name for a specific course.
`public String toString()`	Returns the string representation for the course class.

Class **CourseOffering**

Instance Variables
`private String semester;`
`private ArrayList<String> sections;`
`private ArrayList<Student> students;`
`private ArrayList<Student> teachingAssistantApplications;`
`private ArrayList<Student> teachingAssistants;`

Constructors
`public CourseOffering(Course course, String semester, ArrayList<String> sections, ArrayList<Student> students, ArrayList<Student> teachingAssistantApplications, ArrayList<Student> teachingAssistants)`
`public CourseOffering(String courseID, int courseCode, String courseName, String semester)`
`public CourseOffering(String courseID, int courseCode, String courseName, String semester, ArrayList<String> sections, ArrayList<Student> students)`
`public CourseOffering(String courseID, int courseCode, String courseName, String semester, ArrayList<String> sections, ArrayList<Student> students, ArrayList<Student> teachingAssistantApplications, ArrayList<Student> teachingAssistants)`

Methods	Descriptions
`public boolean addStudent(Student student)`	Adds a student to the class.
`public boolean addTeachingAssistant(Student newTA)`	Adds a new teaching assistant to the class if they have previously applied and are not already a teaching assistant.
`public boolean addTeachingAssistantApplication(Student newTAApplication)`	Adds a new teaching assistant application to the list of teaching assistant applications.
`public void removeTAApplication(int studentID)`	Removes the teaching assistant application of a student with the specified ID from the list of teaching assistant applications for this course, if it exists.
`public ArrayList<String> getSections()`	Returns the sections of the courseOffering
`public String getSemester()`	Returns the semester of the course offering.

`public ArrayList<Student> getStudents()`	Returns the students registered for the course offering.
`public ArrayList<Student> getTeachingAssistantApplications()`	Returns the list of teaching assistant applications for this course offering.
`public ArrayList<Student> getTeachingAssistants()`	Returns the list of teaching assistants for this course offering.
`public boolean isFutureCourse(String currentSemester)`	Checks whether the course offering is a future course offering based on the current semester.
`public boolean isPastCourse(String currentSemester)`	Checks whether the course offering is a past course offering based on the current semester.
`public boolean removeStudent(Student student)`	Removes the specified registered student from the course offering.
`public Student retrieveTAApplicant(int studentID)`	Retrieves the teaching assistant applicant with the given student ID from the list of teaching assistant applications for this course.
`public void setSections(ArrayList<String> sections)`	Sets the list of sections for the course offering.
`public void setStudents(ArrayList<Student> students)`	Sets the list of students registered in the course offering.
`public void setTeachingAssistantApplications (ArrayList<Student> teachingAssistantApplications)`	Sets the list of teaching assistant applications.
`public void setTeachingAssistants (ArrayList<Student> teachingAssistantApplications)`	Set the list of teaching assistants for the course offering.
`public boolean studentInClass(int id)`	Checks if a student with the given ID is enrolled in this course offering.
`public boolean teachingAssistantApplicationInClass (int id)`	Checks whether a student with the given ID has applied for a teaching assistant position in this course offering.
`public boolean teachingAssistantApplications (int id)`	Checks whether a student with the given ID is a teaching assistant in the course.
`public String toString()`	Returns the String representation of a courseOffering.

Class **CourseAttempt**

Instance Variables
`private String section;`
`private String userRole;`
`private String courseProgress;`
`private CourseOffering courseOffering;`

Constructors
`public CourseAttempt(CourseOffering courseOffering, String section,String userRole)`

Methods	Descriptions
`public String getSection()`	Returns the section of the course being attempted.
`public String getUserRole()`	Returns the role of the user attempting the course.
`public String getCourseProgress()`	Returns the progress the user has made in the course.
`public CourseOffering getCourseOffering()`	Returns the `CourseOffering` being attempted.
`public void setCourseProgress(String courseProgress)`	Sets the progress the user has made in the course.
`public void setSection(String section)`	Sets the section of the course attempt.
`public String toString()`	Returns the String representation of the courseAttempt object.

Class **CourseCatalog**

Instance Variables
`private HashMap<String, CourseOffering> courseOfferings;`

Constructors
`public CourseCatalog(String semester)`

Methods	Descriptions
`public String generateEmail()`	Generates an email address using the provided first name and last name.
`public Course getCourse(String courseID, int courseCode)`	Returns a Course object corresponding to the given course ID and course code.
`public String HashMap<String,CourseOffering> getCourseCatalogOfferings(String semester)`	Returns a HashMap containing the course offerings for the specified semester. Used to initialize the course catalog for demo purposes.
`public String getCourseName(String courseID, int courseCode)`	Returns the name of the course corresponding to the given course ID and course code.
`public String HashMap<String,CourseOffering> getCourseOfferings()`	Returns the hashmap that is used to access a courseOffering with a given key.
`public String getFullCourseID(Course course)`	Returns the full ID of the given Course.
`public String getFullCourseID(String courseID, int courseCode)`	Returns the full ID of the course with the given ID and code. The full ID is a combination of the course ID and course code.
`public String getMiddleName(int index)`	Returns a randomly generated middle name from a list of names.
`public HashMap<Integer, Student> getStudents()`	Returns a mapping from studentID's to Student objects. This is used to populate the hard coded courses for demo purposes.
`public String toString()`	Returns the string representation for the courseCatalog class.
`private int[] getCisCourseCodes()`	Returns a list of CIS course codes for a range of CIS courses

`private String[] getCisCourseNames()`	Returns a list of CIS course names for a range of CIS courses
`private int[] getMathCourseCodes()`	Returns a list of MATH course codes for a range of MATH courses
`private String[] getMathCourseNames()`	Returns a list of MATH course names for a range of MATH courses
`private String[] getOtherCourseIDs()`	Returns a list of any other course ID's.
`private int[] getOtherCourseCodes()`	Returns a list of other course codes for a range of other courses.
`private String[] getOtherCourseNames()`	Returns a list of other course names for a range of other courses.