Group 5

Bettervisor

Milestone 3

Class List

Team Members

Anyaegbunam, Chinaza E. (1158144)
Bhandari, Adhyayan (1135943)
Brener, Or (1140102)
Fraser, Ashlyn W. (1098724)
Haroon, Joudat (1146720)
Hung, Jerritt (1140292)
Kandage Wijewardhana, Maneesh B. (1125828)
Tayem, Nour A. (1093248)

TABLE OF CONTENTS

C1: Person

Class Name: Person
Instance Variables:

1) private String firstName
2) private String preferredName
3) private String lastName

Constructor:

1) Person(firstName, preferredName, lastName)
    a) Instantiates a Person object and assigns the arguments to the class' corresponding instance

Methods: N/A

C2: Student

Class Name: Student
Instance Variables:

1) private int studentID
2) private int currSemester
3) private String universityEmail
4) private float gpa
5) private String major
6) private String minor
7) private float balance
8) private boolean isDomestic
9) private float totalCredits
10) private PersonalInfo personalInfo
11) private ArrayList<FutureSemester> futureSchedule
12) private ArrayList<Course> completedCourses
13) private ArrayList<Course> registeredCourses

Constructor:

1) Student(firstName, preferredName, lastName, studentID, currSemester, universityEmail, gpa, major, minor, balance, isDomestic, totalCredits, personalInfo, futureSchedule, completedCourses, registeredCourses)
    a) Instantiates a Student object with the passed in arguments as attributes

Methods:

1) public String applyGraduation(Counselor)
    a) Asks for applicable grad app fields from user and calls GradApp Constructor
    b) Using the passed in Counselor object, call makeGradAppDecision with the newly created GradApp
    c) Returns string returned by makeGradAppDecision
2) public String applyMinor(Counselor, minor)
    a) Calls MinorApp Constructor
    b) Using the passed in Counselor object, call makeMinorAppDecision with the newly created MinorApp
    c) Returns the string created by makeMinorAppDecision

3) public String registerCourse(CourseCatalog, courseCode)
    a) Calls searchCatalog to obtain the course object that contains the given course code
    b) Calls selectSection to obtain the section that the student wants to register into
    c) Calls isFull to check whether the section chosen is full or not
    d) If section is full:
        i) Call addToWaitList to add the student to the course waitlist
    e) If section is not full:
        i) Call addToClassList to add the student to the course
        ii) Call updateSection to remove an empty seat from the section
        iii) Call updateBalance to add the course fee to the student's tuition
    f) A string containing a message of success or failure is returned
4) public String addCourseReview(courseCode)
    a) Call findCourse with the passed in courseCode to retrieve the Course Object
    b) If findCourse returns null, return a message to the user indicating that they can't review the course
    c) If findCourse returns a non-null Course Object call createReview with the retrieved Course Object
5) public String deregisterCourse(CourseCatalog, courseCode)
    a) Call searchRegisteredCourses to check if Student is trying to deregister from a registered course.
    b) Call searchClassList to find the section that the student is currently in
    c) Call removeFromClassList to delete the student entry from the class list and perform necessary updates
    d) Call updateBalance to deduct the course fee from the student that was removed from course's classList
6) public String updatePersonalInfo()
    a) Using the student's personalInfo instance variable, it prints it to stdout with numbered options for each field by calling displayPersonalInfo
    b) Student is prompted for an integer input corresponding to the field they would like to update
    c) If student's choice is from 1-4:
        i) Student is prompted for a new value for this field
        ii) Method verifies the format of the input
        iii) The appropriate setter method is called for the PersonalInfo class
    d) If student's choice is 5 meaning ecAddress:
        i) Student is prompted for a new value for each of the address fields and hits enter to move onto next field
        ii) Method verifies the format of the input
        iii) Call setEcAddress
    e) A string containing a message of success or failure is returned

7) public String updateBalance(Course, addMinusCost)
    a) Calls the getCost method in the given course object to obtain the course fee
    b) If addCost is true, add the cost
8) public String payBalance()
    a) Student is prompted for an amount to pay
    b) Using the inputted amount, the student's balance is decremented
    c) String is returned indicated success or failure with detailed information if anything went wrong
9) private Course findCourse(courseCode)
    a) Loops through Student's registeredCourses instance variable and tries to find a match for the courseCode that was passed in using getCourseCode()
    b) Returns a copy of the courseObject if a match was found, otherwise returns null
10) private void createReview(Course):
    a) Prompts user for rating and comment fields
    b) Validates input for profanity
    c) Calls CourseReview Constructor
    d) Calls publishReview using the passed in Course object
11) private Course searchRegisteredCourses(courseCode)
    a) Method will loop through the student's registeredCourses instance variable
    b) Once it finds the course code that was passed in, it will return that corresponding Course object
    c) If not found, returns null

12) public String requestRefund(amount)
    a) Calls isEligible to determine whether the student is eligible for a refund or not
    b) If student is eligible:
        i) If student is not domestic, call deductFee to deduct foreign transaction fees from the refund
        ii) Call getBankDetails to obtain the student's financial information
        iii) Call submitRequest to submit a request to the university for a refund
13) public boolean isEligible()
    a) Student is not eligible and returns false if the following criteria is met:
        i) Refund amount requested > balance
        ii) isDomestic == false and curSemester < 2
    b) Otherwise, return true indicating that the student is eligible for a refund
14) private float deductFee(amount)
    a) Deducts a set fee from the given amount and returns the result
15) public String getBankDetails()
    a) Returns the student's banking information
16) public String submitRequest()
    a) Submits a request to the student for a refund, returning a message indicating success or failure with detailed information if anything went wrong

17) public void setMinor(minor)
    a) Sets the student minor instance variable with the passed in argument
18) public String payTuition()
    a) Iterate through registeredCourses instance variable and generate a bill using the getCost method for each course
    b) Call isPastDeadline to ensure the student is still capable of paying their fee
    c) If isPastDeadline returns true, loops through registeredCourses, calls getCourseCode for each course and passes it into deregisterCourse
    d) Otherwise, call payBalance
19) public String createFutureSemester(courseCatalog,year, season)
    a) Call the FutureSemester Constructor
    b) Following is looped until the student exits or has planned 5 courses
        i) The student is asked to enter in the course code they want to plan, and is passed into searchCatalog to retrieve the correct Course object
        ii) Call isRestricted to ensure that the student isn't restricted from taking the course
        iii) Call getNumPlanned to ensure that they haven't planned 5 courses already
        iv) Call addToCoursePlan with the previously retrieved Course Object
    c) After the student is done planning, call addFutureSemester
20) public void addFutureSemester(FutureSemester)
    a) Adds the given future semester to the student's created future semester list

C3: Counselor

Class Name: Counselor
Instance Variables:

1) private int employeeID
2) private String faculty
3) private String primaryEmail

Constructor:

1) Counselor(firstName, preferredName, lastName, employeeID, faculty, primaryEmail)
    a) Instantiates a MinorApp object with the employeeID, faculty and primaryEmail

Methods:

1) public String addMeeting(Meeting)
    a) Method Description
2) public String makeGradAppDecision(GradApp)
    a) Calls getStudent to obtain the student that submitted the graduation application
    b) Verifies that student object within GradApp meets criteria in order to graduate:
        i) currSemester=8
        ii) balance=0
    c) Calls setIsApproved to set the approval status of the GradApp object
    d) String containing a Success or Failure message is returned depending on approval or denial of grad application
3) public String makeMinorAppDecision(MinorApp)
    a) Calls getStudent to obtain the student that submitted the minor application
    b) Verifies that the student object within MinorApp meets criteria to declare a minor:
        i) currSemester!=8
        ii) gpa >= 60
    c) Calls setIsApproved to set the approval status of the MinorApp object
    d) Calls setMinor to update the student's minor with the new changes
    e) String containing a Success or Failure message is returned depending on approval or denial of minor declaration

C4: Course

Class Name: Course
Instance Variables:

1) private String courseCode
2) private HashMap<Student, Section> classList
3) private ArrayList<Section> sectionList
4) private ArrayList<Course> prerequisites
5) private Date examTime
6) private float cost
7) private String term
8) private float weight
9) private String description
10) private ArrayList<Course> restrictions
11) private ArrayList<CourseReview> reviews

Constructor:

1) Course(courseCode, classList, sectionList, prerequisites, examTime, cost, term, weight, description, restrictions, reviews)
    a) Instantiates a Course object with the corresponding arguments as instance variables

Methods:

1) public Section selectSection()
    a) Lists the course's sections by looping through the course's section list and calling the toString method in each section object, printing out the returned string
    b) Prompts the user to select a section, returning the selected section
2) public String addToClassList(Student, Section)
    a) Adds the given student to the course's class list, mapping the student to the section they were added to
3) public void removeFromClassList(Student, Section)
    a) Method removes the student key from the course's classList instance variable which subsequently removes the student->section mapping entirely
    b) Call updateSection
    c) Call isFull
    d) If the section is not full:
        i) Call popFromWaitlist to retrieve a student from the waitlist
        ii) Call addToClassList with the popped student
        iii) Update popped student's balance by calling updateBalance

4) public String toString()
    a) Returns a string containing the course code, description, weight, prerequisites, and restrictions
5) public String publishCourseReview(CourseReview)
    a) Appends the courseReview object to a course's reviews instance variables
6) public float getCost()
    a) Returns the cost of a course
7) public Section searchClassList(Student)
    a) Using the course's classListinstance variable, method checks if student is contained within it
    b) If student is found, returns the value corresponding with the student key which is the section they are enrolled in
    c) If student is not found, returns null
8) public int getCourseCode()
    a) Returns the course's courseCode
9) public boolean isRestricted(student)
    a) Returns true if a course in the given student's registered or completed courses is part of the course's restrictions, and returns false otherwise

C5: Section

Class Name: Section
Instance Variables:

1) private int sectionID: int
2) private int emptySeats: int
3) private int maxSeats: int
4) private ArrayList<Student> waitlist
5) private Date lectureTime
6) private Date labTime

Constructor:

1) Section(sectionID, emptySeats, maxSeats, waitlist, sectionList, lectureTime, labTime)
    a) Instantiates a Section object with the given section ID, number of empty and max seats,

Methods:

1) public String addToWaitlist(Student)
    a) Adds the given student to the section's waitlist
2) public Student popFromWaitlist()
    a) Performs a FIFO removal of a student in the section's waitlist and returns the removed student
3) public boolean isFull()
    a) Returns true if the section's empty seats is <= 0, and false if the section's empty seats is > 0
4) public String updateSection(numSeats)
    a) Adds a given number of seats to the section's empty seats
    b) Can be positive to add empty seats, or negative to remove empty seats
5) public String toString()
    a) Returns a string containing the section ID, number of empty seats vs max seats, lecture time, and lab time

C6: CourseCatalog

Class Name: CourseCatalog
Instance Variables:

1) private ArrayList<Course> courseList
2) private Date paymentDeadline

Constructor:

1) CourseCatalog(paymentDeadline)
    a) Instantiates a CourseCatalog object with the given payment deadline

Methods:

1) public Course searchCatalog(courseCode)
    a) Searches through the course list and returns the course that has the given course code
2) public void addCourse(Course course)
    a) Adds a given course to the course list of the course catalog
3) public String listCourses()
    a) Loops through the course list and calls the toString method in each course object, printing out the returned string
4) public boolean isPastDeadline(Date)
    a) Returns true if the given date comes before the payment deadline, and false if the given date comes after the payment deadline

C7: CourseReview

Class Name: CourseReview
Instance Variables:

1) private String courseCode
2) private int rating
3) private String comment
4) private Date reviewDate

Constructor:

1) CourseReview(courseCode, rating, comment)
    a) Instantiates a CourseReview object with the given courseCode, rating and comment
    b) Sets reviewDate equal to current date

Methods: N/A

C8: PersonalInfo

Class Name: PersonalInfo
Instance Variables:

1) private String ecFirstName
2) private String ecNumber
3) private String ecEmail
4) private Address ecAddress
5) private String personalNumber

Constructor:

1) PersonalInfo(ecFirstName, ecNumber, ecEmail, ecAddress, personalNumber)
    a) Instantiates a PersonalInfo object with the ecFirstName, ecNumber, ecEmail, ecAddress and personalNumber

Methods:

1) public void displayPersonalInfo()
    a) displays the personal info of the student after it's been verified.
2) public void setEcFirstName(ecFirstName)
    a) Sets the emergency contact's first name
3) public void setEcNumber(ecNumber)
    a) Sets the emergency contact's phone number
4) public void setEcEmail(ecEmail)
    a) Sets the emergency contact's email
5) public void setEcAddress(country, province, streetName, streetNumber, postalCode)
    a) Call all the appropriate setter methods in Address class
6) public void setPersonalNumber(personalNumber)
    a) Sets the student's personal phone number

C9: Address

Class Name: Address
Instance Variables:

1) private String country
2) private String province
3) private String streetName
4) private String streetNumber
5) private String postalCode

Constructor:

1) Address(country, province, streetName, streetNumber, postalCode)
    a) Instantiates a Address object with the country, province, streetName, streetNumber and postalCode

Methods:

1) public void setCountry(country)
    a) Sets the student's country
2) public void setProvince(province)
    a) Sets the student's province
3) public void setStreetName(streetName)
    a) Sets the student's street name
4) public void setStreetNumber(streetNumber)
    a) Sets the student's street number
5) public void setPostalCode(postalCode)
    a) Sets the student's postal code

C10: FutureSemester

Class Name: FutureSemester
Instance Variables:

1) private ArrayList<Course> courseList
2) private int year
3) private String season

Constructor:

1) FutureSemester(year, season)
    a) Instantiates a FutureSemester object that represents a student's schedule for the given year and season

Methods:

1) public String addCourseToPlan(course): String
    a) Adds a course to the plan of the given year and season
2) public int getNumPlanned()
    b) Returns the number of planned courses in the future semester

C11: Application

Class Name: Application(Abstract)
Instance Variables:

1) private boolean isApproved

Methods:

1) public void setIsApproved(boolean)
    a) Sets the isApproved variable to the boolean argument indicating whether or not the application has been approved

C12: GradApp

Class Name: GradApp
Instance Variables:

1) private Student student
2) private String major
3) private String institution
4) private String acquisitionType
5) private boolean distinction

Constructor:

1) GradApp(student, major, institution, acquisitionType, distinction)
    a) Instantiates a GradApp object with the given student, major, institution, acquisition type, and distinction

Methods:

1) public String printGradApp()
    a) Displays the student's first and last name, major, institution, acquisition type and whether a distinction was achieved to stdout
2) public Student getStudent()
    a) Returns a copy of the student object

C13: MinorApp

Class Name: MinorApp
Instance Variables:

1) private Student student
2) private String minor

Constructor:

1) MinorApp(student, minor)
    a) Instantiates a MinorApp object with the given student and minor

Methods:

1) public Student getStudent()
    a) Returns a copy of the student object