Text

Description automatically generated

**School of Engineering and Information Technology**

**ASSIGNMENT COVER SHEET**

* Please complete and insert this form as the first page of EACH electronic assignment.
* Submit the assignment with attached coversheet electronically as per the instructions in the Assignment Question sheet.
* Please make sure you keep a copy of the assignment.

**Student Details**

| Surname | Ang | Given name | Jinwei |
| --- | --- | --- | --- |
| Student Number | 34792195 | Email | xsolsticegfx@gmail.com |

**Assignment details**

| Unit name | Principles of Computer Science | Unit Code | ICT167 |
| --- | --- | --- | --- |
| Unit Coordinator | Kevin Wong | Tutor/Tutorial time | Steven Loke  630PM |
| Due date/time | Saturday,25 March 2023, 11:59 PM | Submission date |  |
| Assignment title | Assignment 2 | | |
| Other information | Do clarify with me if there’s any questions on the program. | | |

***All forms of plagiarism, cheating and unauthorized collusion are regarded seriously by the University and could result in penalties including failure in the unit and possible exclusion from the University. If in doubt, please contact the Unit Coordinator.***

**Student's Declaration**

Please double click on all the check boxes.

☒ Except where I have indicated, the work I am submitting in this assignment is my own work and has not been submitted for assessment in another unit.

☒ This submission complies with Murdoch University policies regarding plagiarism, cheating and collusion.

☒ I have retained a copy of this assignment for my own records.

## Table of Contents

[Table of Contents 3](#_iz8fruccisho)

[Project Requirements/Specifications 4](#_fbc7o7bjsndm)

[User Guide 11](#_14elcxonh6s)

[Structure/Design/Algorithm 20](#_pqn8fp1p8a0l)

[Limitations 28](#_xufj4d4pcn6a)

[Testing 29](#_tm9eb0gf5ive)

[Source Program Listing 36](#_opaorwb8rta4)

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## Project Requirements/Specifications

The program will load information from a CSV file and store the information in an ArrayList. The program will then display a menu of several options for users to choose from (1,2,3,4,5,6,7,8).

There will be 2 parent classes (Student and Unit).

Student\_Research and Student\_Course class will be derived from Student class.

Unit\_Course and Research class will be derived from Unit class.

The ArrayList will store Student and Unit pairing 1-to-1. For example, index 0 will hold Student 1 and index 1 will hold Student 1’s unit.

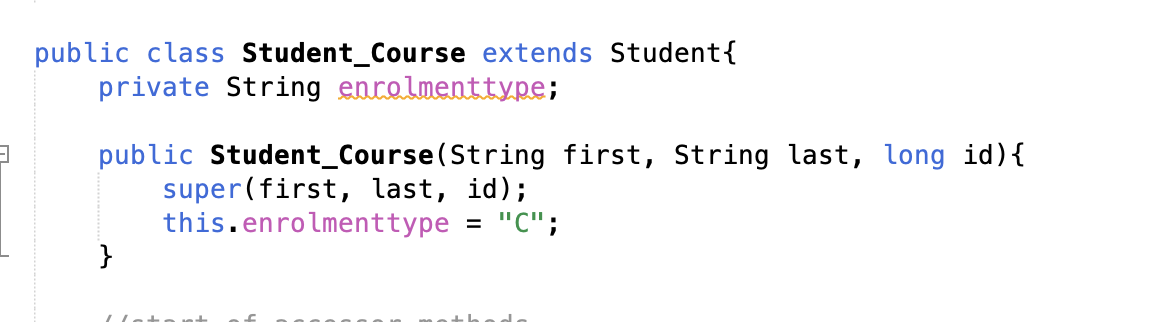
**Menu Options:**

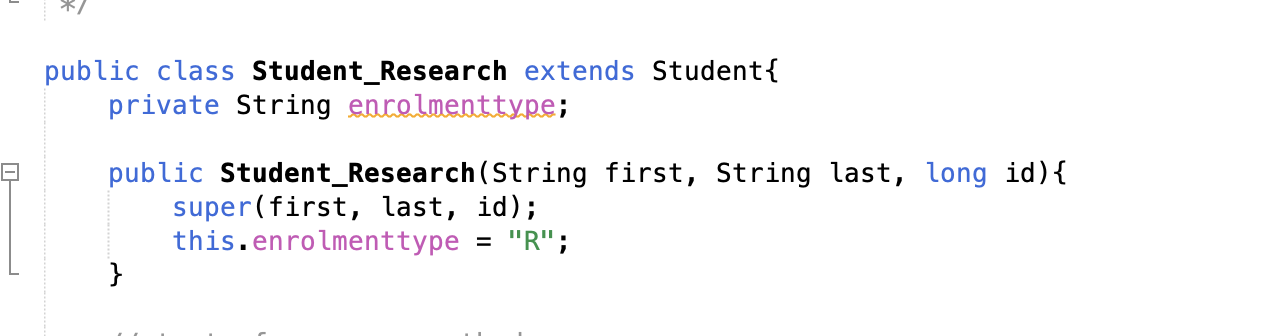
1. Quit
2. The program will add marks of students by reading it from another CSV file.
3. Remove a student from the ArrayList based on Student ID.
4. Output all details in the ArrayList.
5. Determine how many students in coursework scored below and above the average overall score.
6. Report grade of a student based on Student ID.
7. Sort the ArrayList in ascending order based on Student ID using bubble sort.
8. Output the sorted ArrayList to a CSV file.

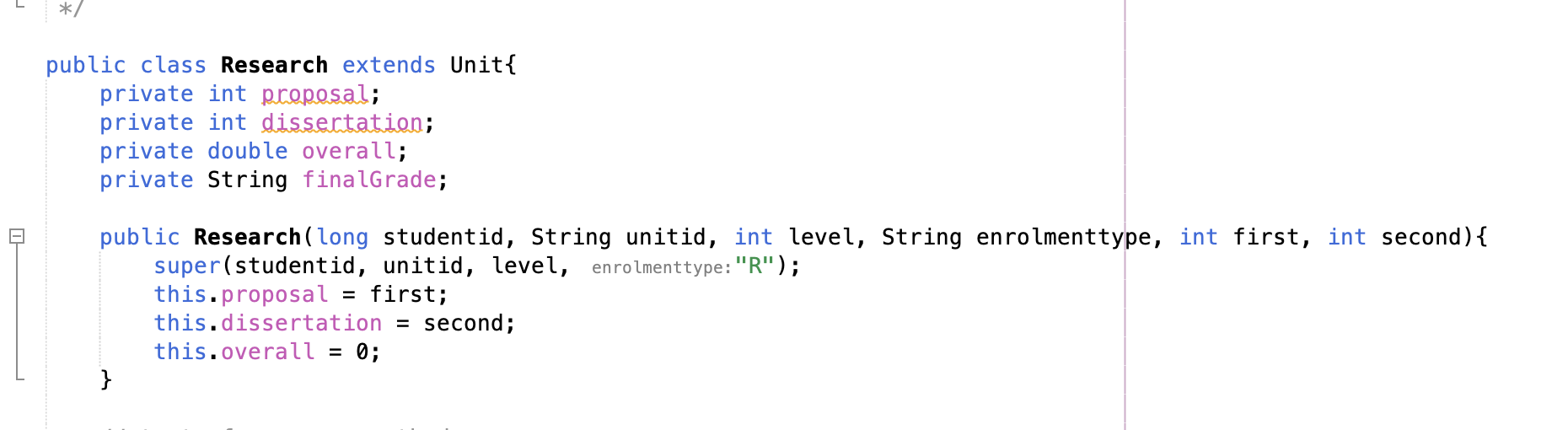
# **Inheritance:**

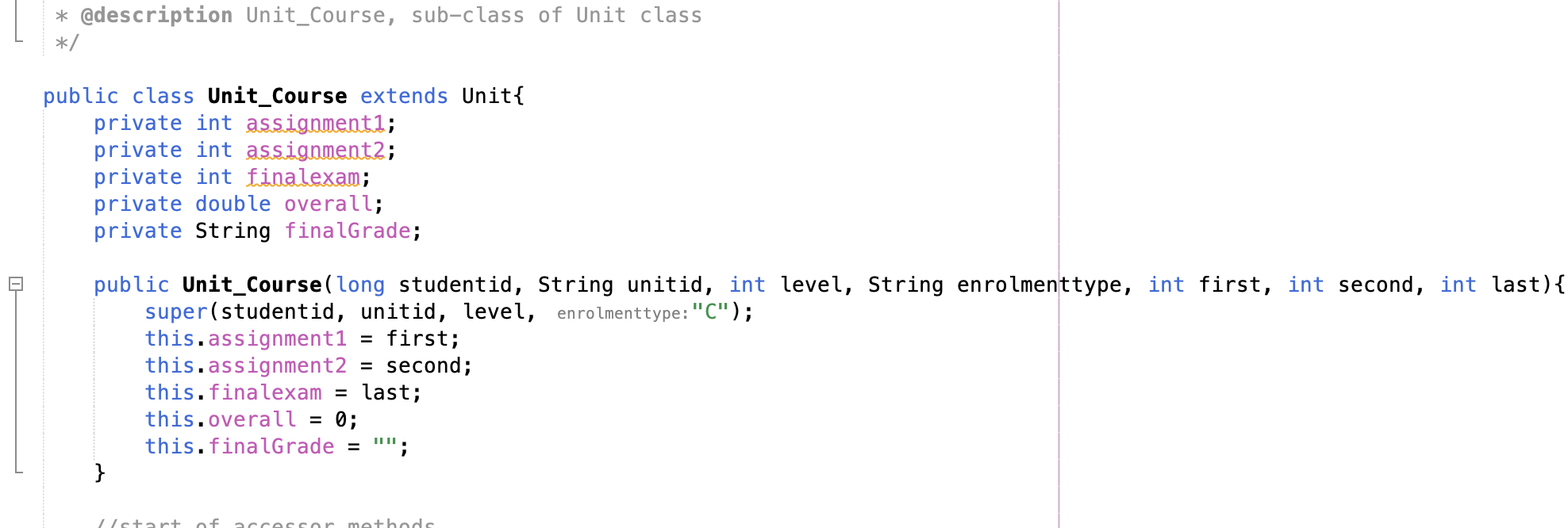
Student\_Research and Student\_Course class are inherited from the Student class.

Unit\_Course and Research class are inherited from the Unit class.









**Polymorphism:**

There will be different implementations of a method depending on the type of object that is passed to the method.

For example, when reportGrade() is called from the Student class, it will return “There is no grade here.”, but when the same method is called from the Student\_Research or Student\_Course class, it will then report the grade of the particular student.

Student.java



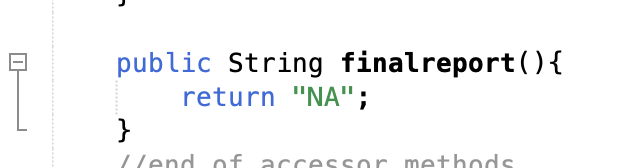
Student\_Course.java & Student\_Research.java



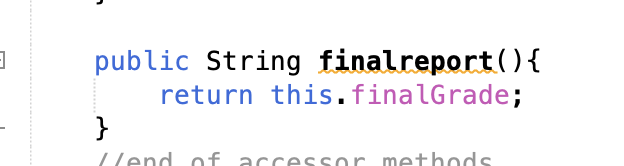
**Dynamic Binding:**

Similar to polymorphism, dynamic binding is achieved by having a method overridden when it is being called by a subclass. To achieve this, the name of the method must be the same in the parent class and the sub class, the parameters must be same and the relationship between the classes must be inherited. In the program, an example of dynamic binding is the finalreport() method in Unit class which returns “NA” when called by the parent class Unit, but when it is being called by the subclasses Unit\_Course or Research, it will then return the final grade of the student.

Unit.java



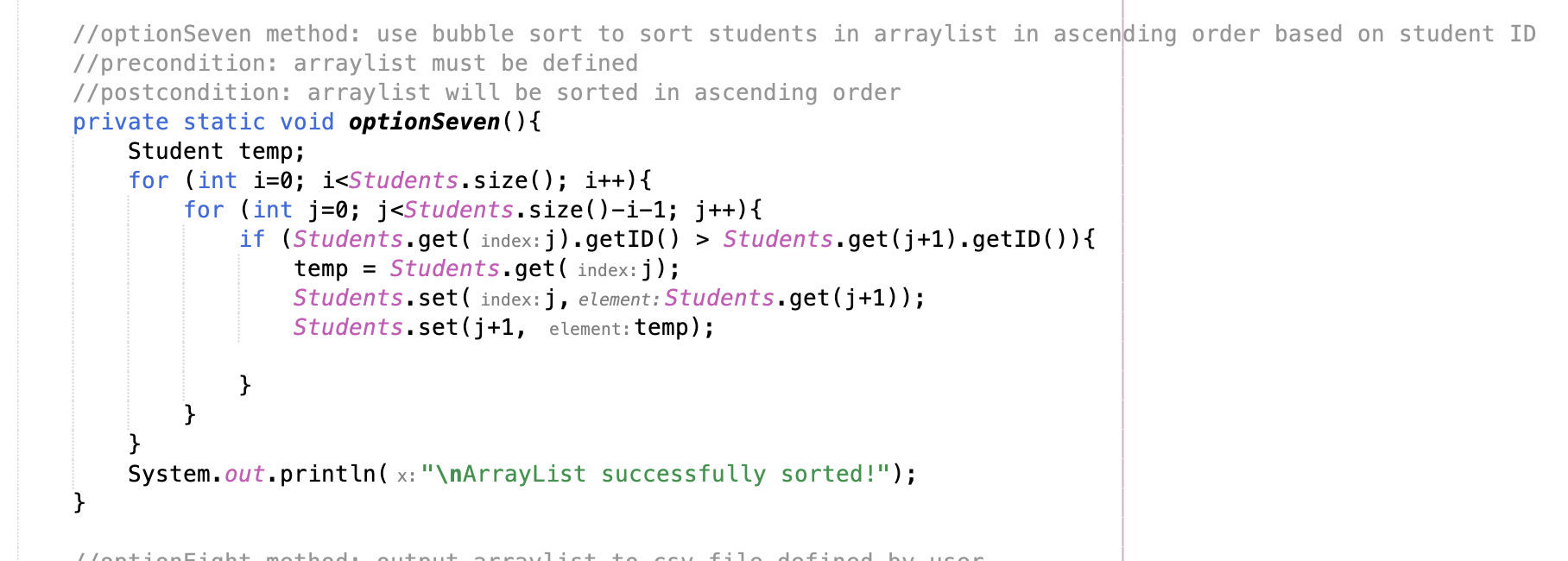
Unit\_Research.java & Unit\_Course.java



**Sorting Algorithm:**

Bubble Sort is used in menu option 7, which will run through the ArrayList and sort the data in an ascending order based on the Student ID. How this work is it will loop through the ArrayList and swap position of the particular object with the adjacent objects if the particular object is in the wrong position.

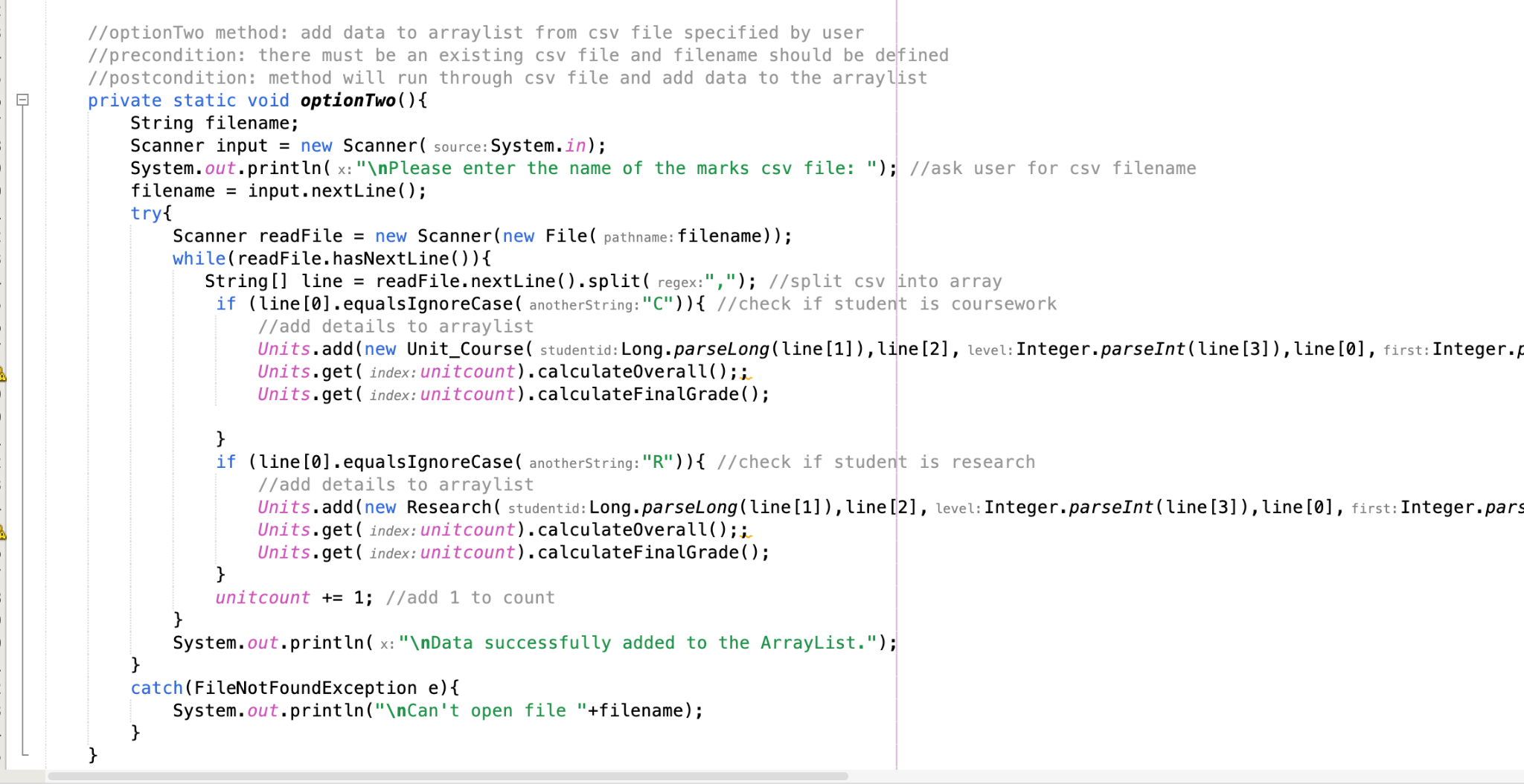
Bubble Sort Algorithm in Client.java



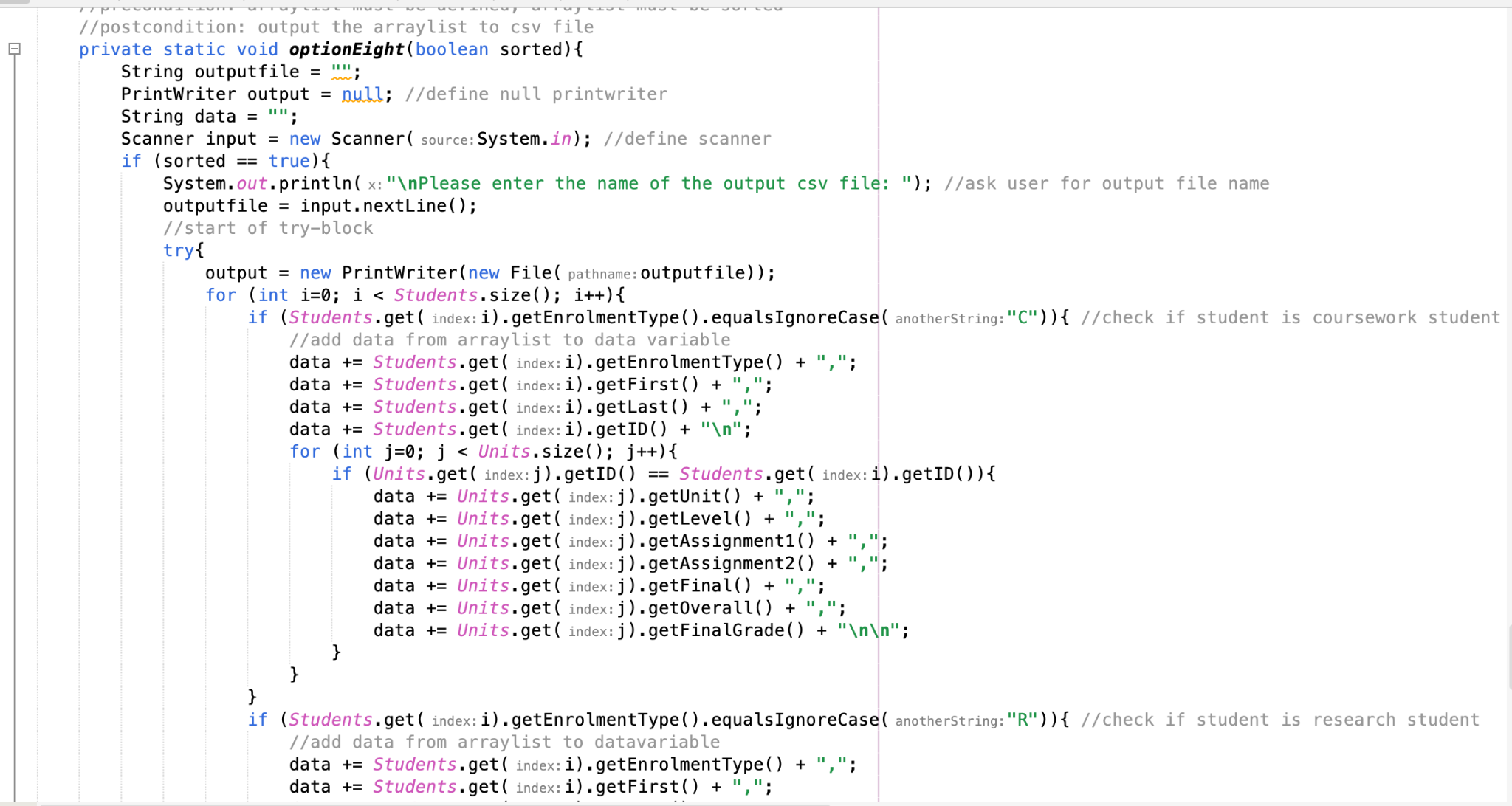
**Handling CSV files:**

The program uses Scanner to read the information from the CSV file and PrintWriter to output the information to the CSV file.





Scanner to import the data.





PrintWriter to output the data.

The Scanner will read a csv file from the project directory and import the data to the ArrayList.

The PrintWriter will write the data to a csv file and store the csv file inside the project directory.

## User Guide

**Apache Netbeans IDE 16** was used for this program.

**Client.java** - Client program to enter data from a CSV file and store them in an ArrayList. Also provides several methods in the form of a menu to perform analysis and queries on the data.

**Student.java** - Student class with a parametized constructor. Stores the firstname, lastname and studentid variables.

**Student\_Course.java** - Subclass of the Student class. Stores the enrolmenttype variable. enrolmenttype set to “C”.

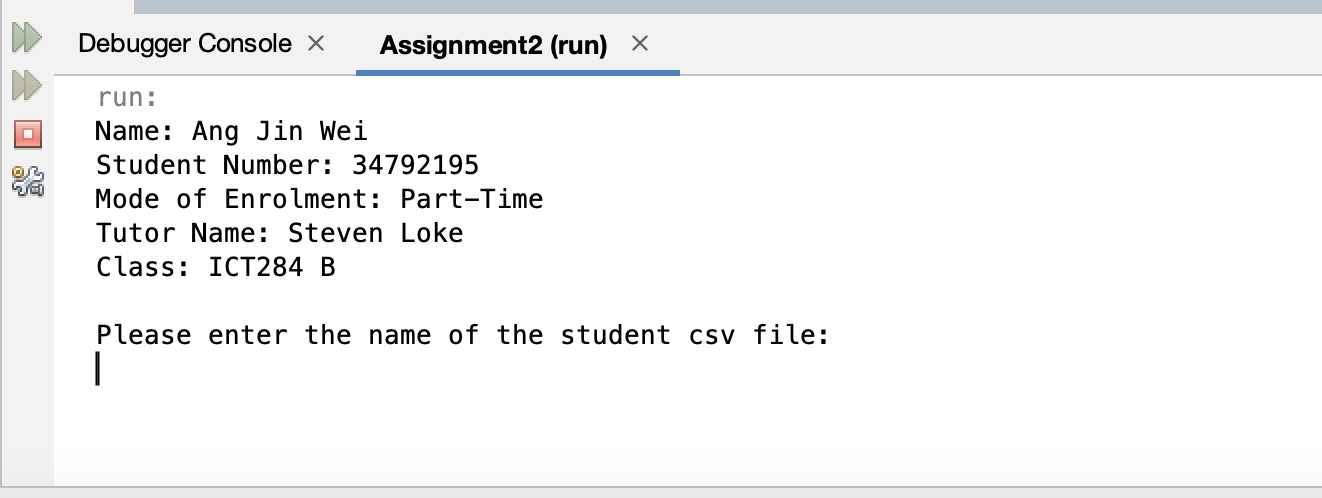
**Student\_Research.java** - Subclass of the Student class. Stores the enrolmenttype variable. enrolmenttype set to “R”.

**Unit.java** - Unit class with a parametized constructor. Stores the unitid, level and enrolmenttype variable.

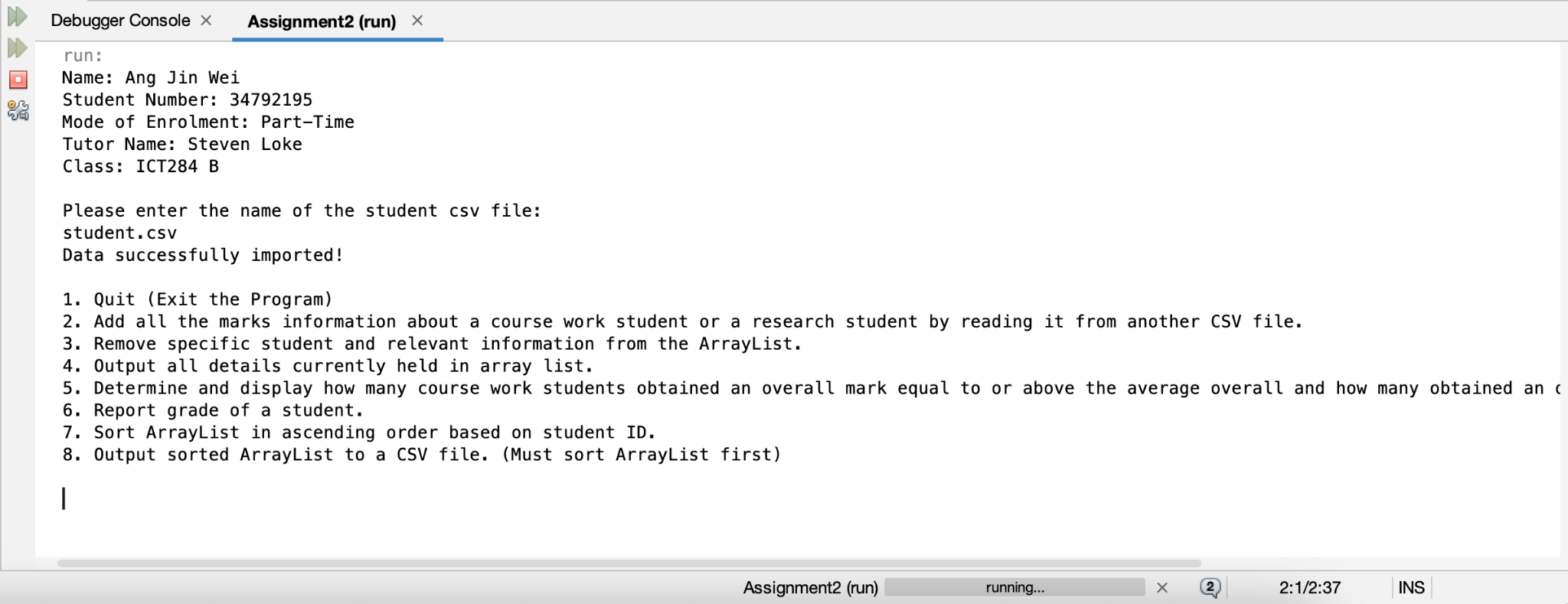
**Unit\_Course.java** - Subclass of the Unit class. Stores the assignment1, assignment2, finalexam, overall and finalGrade variables.

**Research.java** - Subclass of the Unit class. Stores the proposal, dissertation, overall and finalGrade variables.

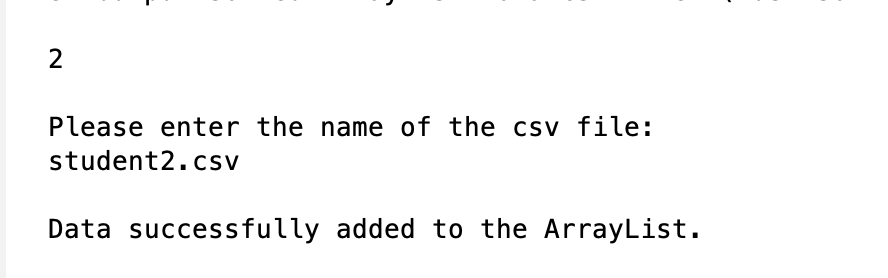
To run the program, start your IDE and open “Assignment2” source. Run the program (Client.java) and follow the instructions.



You will be prompted to enter the name of the csv file you want to import student data from.

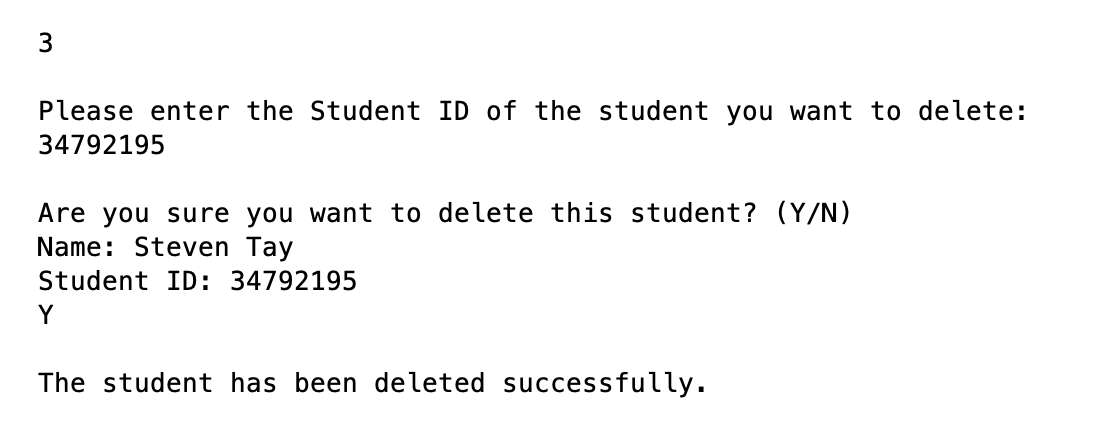


After entering the CSV filename, the data will be loaded to the ArrayList and you will be presented with a menu.

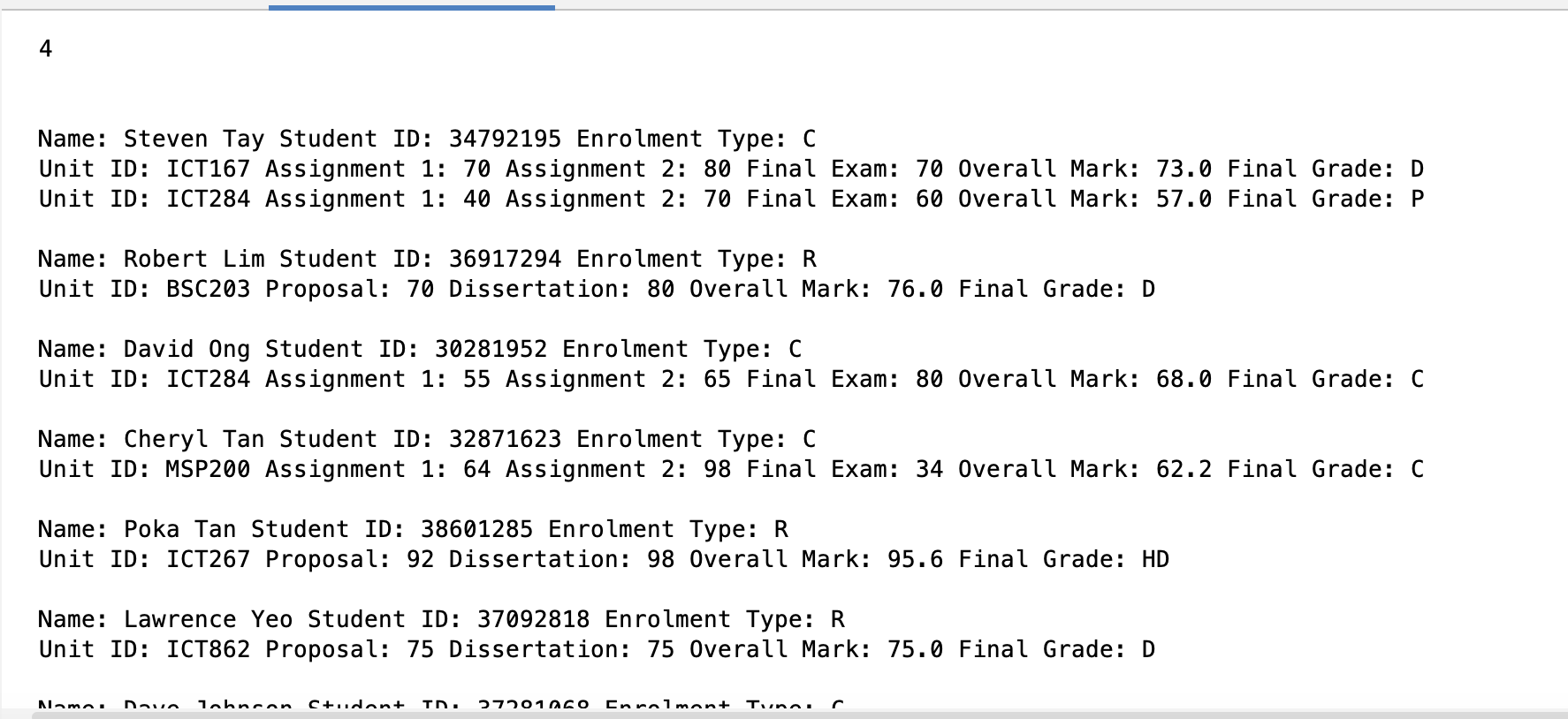


Option 2 will require you to enter the filename of the marks csv file you want to add to the ArrayList. The program will then print a message to show the data has been successfully added to the ArrayList.

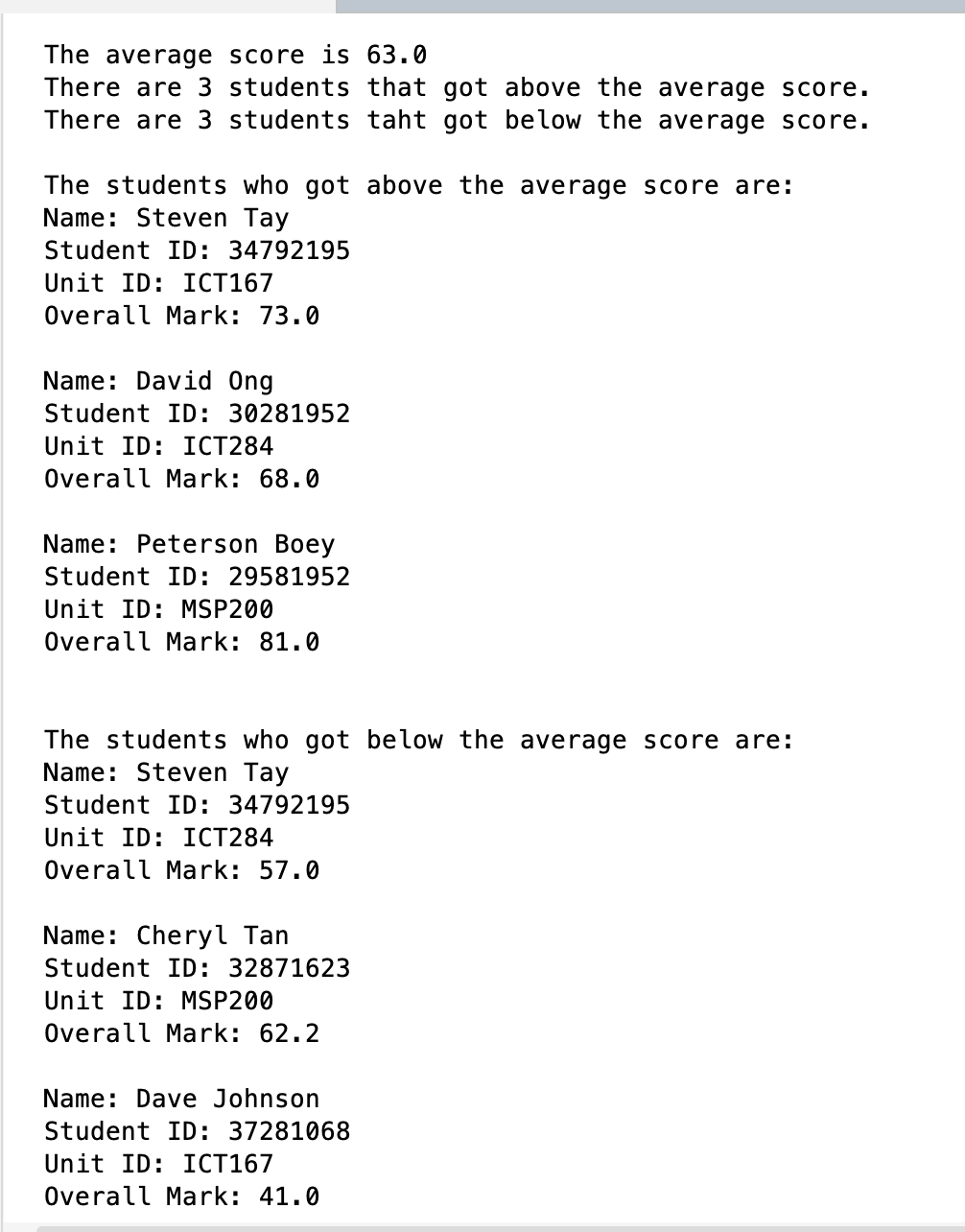
For Option 2, you will be required to follow the CSV format below at the end of the userguide.



Option 3 will require you to enter the specific Student ID you want to delete from the ArrayList. A confirmation prompt will be shown and upon entering “Y”, the student will be deleted from the ArrayList.



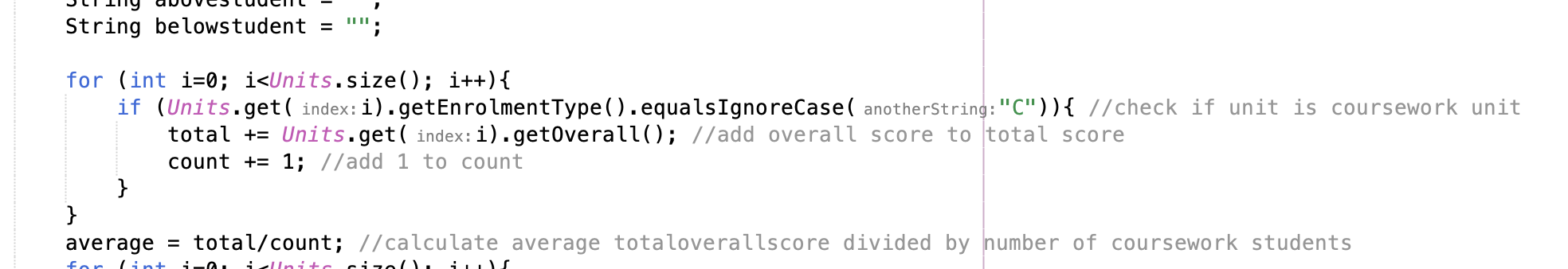
Option 4 will display all the information held in the ArrayList.

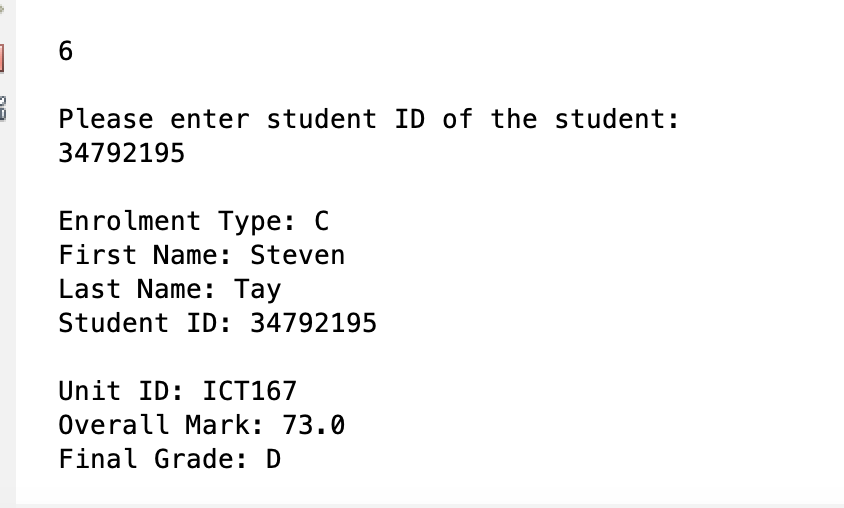


Option 5 will display the average overall score of all the coursework students and also display the amount of students that got above or below the average overall mark.

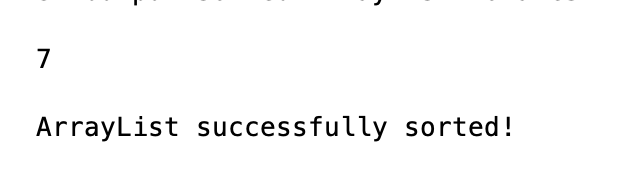
Formula for average overall mark:

1. Check if unit stored is coursework
2. If unit is coursework, add the overall to total, add one to count
3. average = total/count

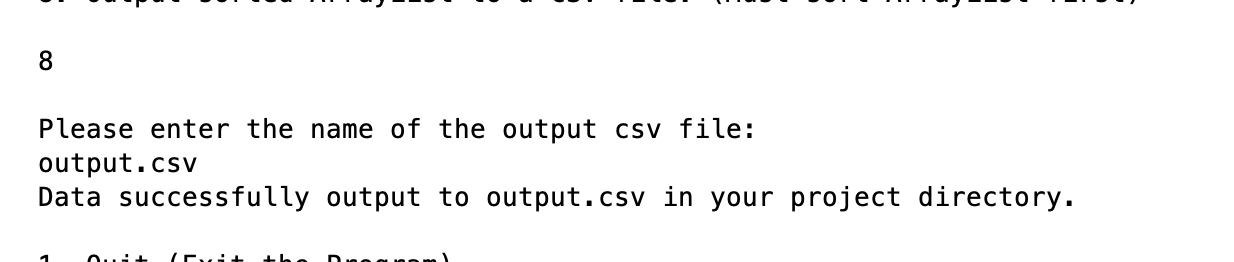




Option 6 will display the grades of a student of your choice by Student ID.



Option 7 will sort the ArrayList in ascending order based on Student ID.





Option 8 will ask you for a csv filename and output the data to the CSV file indicated.

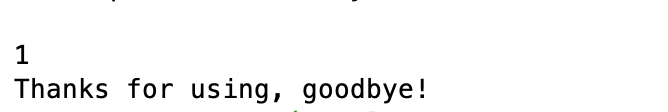
Output format will be: enrolmenttype,firstname,lastname,studentid

Below the student will be their taken units.

For coursework students: unitid,unitlevel,assignment1,assignment2,finalexam,overallmark,grade

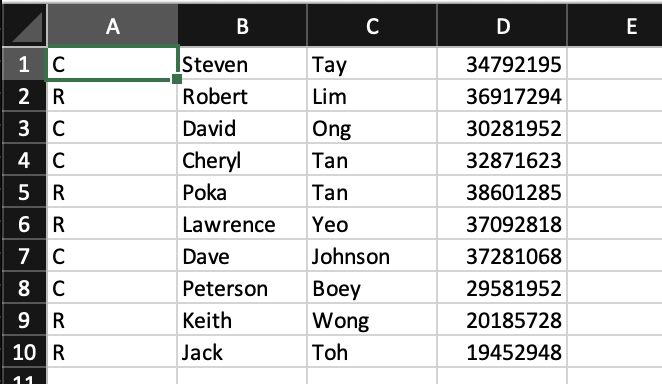
For research students:

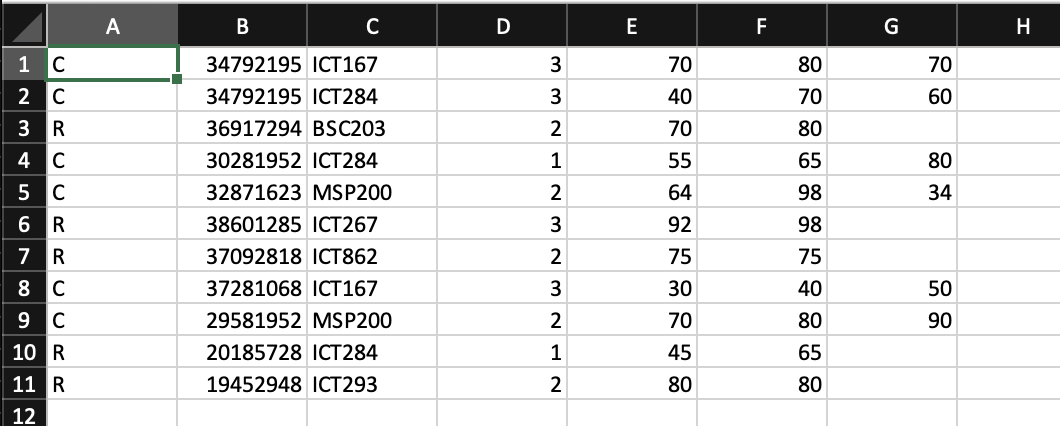
unitid,unitlevel,proposal,dissertation,overallmark,grade



Lastly, you can select Option 1 to exit the program.

**CSV Format for input:**

****

****

**CSV should follow the above formats.**

For initial import student.csv file, format should be:

**enrolmenttype,firstname,lastname,studentid**

(Example: C,David,Lim,12345678)

For student mark csv file(for option 2), format should be:

**Coursework**: **enrolmenttype,studentid,unitid,unitlevel,assignment1,assignment2,finalexam**

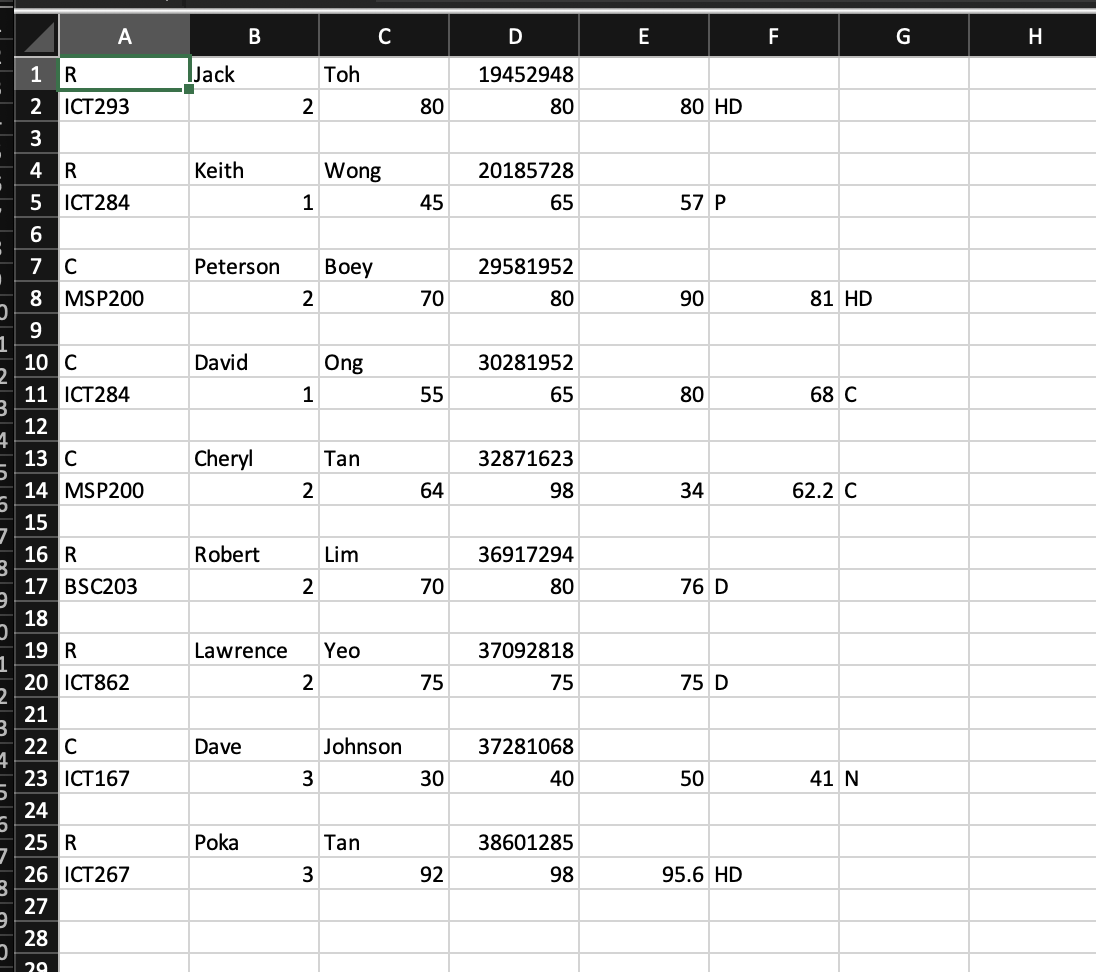
(Example: C,12345678,ICT123,2,50,50,50)

**Research**:

**enrolmenttype,studentid,unitid,unitlevel,proposal,dissertation**

(Example: R,12345678,ICT321,3,50,50)

**CSV Format for Output**

****

**enrolmenttype,firstname,lastname,studentid**

For coursework students:

**unitid,level,assignment1,assignment2,finalexam,overallmark,finalgrade**

For research students:

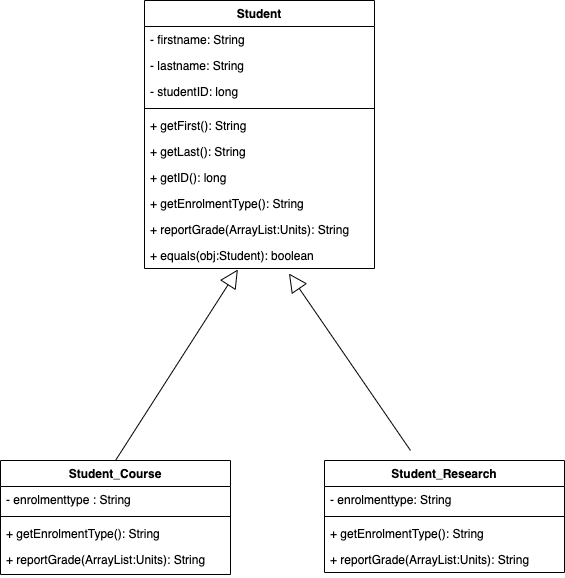
**unitid,level,proposal,dissertation,overallmark,finalgrade**

**The format will always be student following by the units the student has taken.**

**The output csv file will be saved in your project directory!**

## Structure/Design/Algorithm

**Student.java/Student\_Course.java/Student\_Research.java UML**

****

**Student.java**

* Student class
* firstname: String
* lastname: String
* studentID: long
* getFirst() // returns firstname
* getLast() // returns lastname
* getID() // returns studentID
* getEnrolmentType() // returns “NA”
* reportGrade() // returns “There is no grade here.”
* equals() //check if student obj is equal to another obj, returns true/false

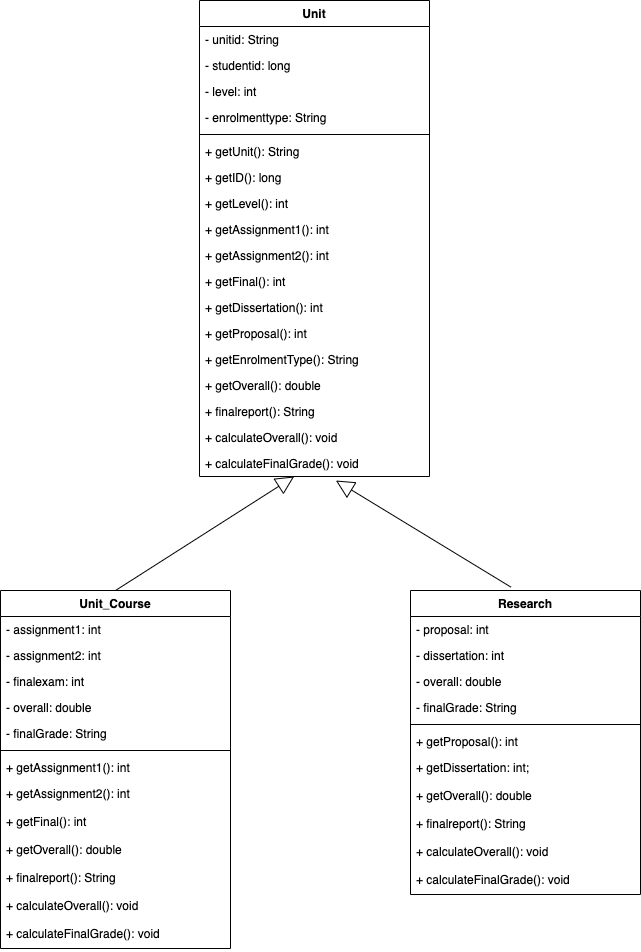
**Student\_Course.java**

* Student\_Course subclass of Student class
* enrolmenttype: String
* getEnrolmentType() // returns enrolmenttype
* reportGrade() // display student and grades

**Student\_Research.java**

* Student\_Research subclass of Student class
* enrolmenttype: String
* getEnrolmentType() // returns enrolmenttype
* reportGrade() // display student and grades

**Unit.java/Unit\_Course.java/Research.java UML**

****

**Unit.java**

* Unit class
* unitid: String
* studentid: long
* level: int
* enrolmenttype: String
* getUnit() // returns unitid
* getLevel() // returns level
* getAssignment1() // returns 0
* getAssignment2() // returns 0
* getFinal() // returns 0
* getDissertation() // returns 0
* getProposal() // returns 0
* getEnrolmentType() // returns enrolmenttype
* getOverall() // returns 0
* finalreport() // returns “NA”
* calculateOverall()
* calculateFinalGrade()

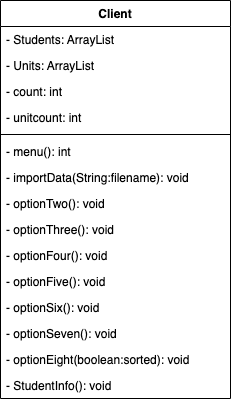
**Unit\_Course.java**

* Unit\_Course subclass of Unit class
* assignment1: int
* assignment2: int
* finalexam: int
* overall: double
* finalGrade: String
* getAssignment1() // returns assignment1
* getAssignment2() // returns assignment2
* getFinal() // returns finalexam
* getOverall() // returns overall
* finalreport() // returns finalGrade
* calculateOverall() // calculates overall mark
* calculateFinalGrade() // calculates final grade

**Research.java**

* Research subclass of Unit class
* proposal: int
* dissertation: int
* overall: double
* finalGrade: String
* getProposal() // returns proposal
* getDissertation() // returns dissertation
* getOverall() // returns overall
* finalreport() // returns finalGrade
* calculateOverall() // calculates overall mark
* calculateFinalGrade() // calculates final grade

**Client.java UML**



**Client.java**

* Client class
* Students: ArrayList
* Units: ArrayList
* count: int
* unitcount: int
* menu() // prints menu and returns choice of user as int
* importData() // read students from csv file and add to arraylist
* optionTwo() // read marks from csv file and add to arraylist
* optionThree() // delete student from arraylist based on student ID
* optionFour() // display all details in arraylist
* optionFive() // determine and display average of coursework students
* optionSix() // report grade of particular student with student ID
* optionSeven() // sort the array in ascending order based on student ID
* optionEight() // output the details in arraylist to csv

**Pseudocodes for Methods**

calculateOverall()

1. define assignment and result as new double
2. assignment = 30% of assignment1 and 30% of assignment2
3. result = assignment + 40% of finalexam
4. round up result to nearest decimal
5. set overall to result

calculateFinalGrade()

1. if overall score is 80 or above, set finalgrade to HD
2. if overall score is 70 or above and below 80, set finalgrade to D
3. if overall score is 60 or above and below 70, set finalgrade to C
4. if overall score is 50 or above and below 60, set finalgrade to P
5. if overall score is below 50, set finalgrade to N

reportGrade()

1. define msg as new String
2. add enrolment type, first name, last name and student id to msg
3. loop through arraylist to find units belonging to the particular student
4. if unit belong to student, add unit id, overall mark and final grade to msg
5. return msg

menu()

1. define scanner and int choice
2. print out menu
3. return user choice

importData()

1. start try block
2. open csv file using scanner
3. catch filenotfoundexception if csv file not found
4. check if csv file has next line
5. split line into array
6. check if student is coursework or research
7. add details of student into arraylist
8. add 1 to count
9. display message showing data has been imported successfully

optionTwo()

1. define filename as String and scanner
2. ask user for input on filename
3. start try block
4. open csv file using scanner
5. catch filenotfoundexception if csv file not found
6. check if csv file has next line
7. split line into array
8. check if unit is coursework or research
9. add details of unit into arraylist
10. add 1 to unitcount
11. display message showing data has been imported successfully

optionThree()

1. initialize boolean found = false, String choice = “”, long delete and scanner
2. ask user for student id
3. loop through array and check if student id equal to user’s input
4. if yes, ask for confirmation from user whether to delete the student, showing the student’s information
5. if user confirm to delete the student, delete all information and marks linked to the student
6. if student id isn’t found in the array, display student id not found

optionFour()

1. define msg as new String
2. loop through arraylist
3. add each student info and units to msg
4. return msg

optionFive()

1. intialize variables total, count, average, above, below, abovestudent, belowstudent
2. loop through array and add overallscore to total, adding 1 to count each time, making sure unit is coursework
3. get average by dividing the total overallscore by the number of units
4. loop through array again and check if each coursework student has above or below the average score
5. add the student’s info and grades to abovestudent or belowstudent and add 1 to above if the student has above the average score, 1 to below if the student has below the average score
6. display average score
7. display number of students with above average score
8. display number of students with below average score
9. display students with above average score
10. display students with below average score

optionSix()

1. intialize variables id, found
2. ask user for student id
3. loop through arraylist to search for user defined student id
4. if student id is found, display the grade of the student
5. if student id is not found, display message student id not found

optionSeven()

1. intialize new Student object temp
2. loop through arraylist
3. loop through the arraylist again for each loop
4. compare studentid of object with adjacent objects
5. if studentid > adjacent object studentid, swap their positions (bubble sort)

optionEight()

1. initialize outputfile, data variables
2. initialize new PrintWriter object = null
3. check if arraylist is sorted
4. start try block
5. catch filenotfoundexception if can’t open csv file
6. if arraylist isn’t sorted, print message asking user to sort arraylist
7. if arraylist is sorted, ask user for output file name
8. loop through arraylist to get student and unit info
9. add the info to data
10. write the data to csv using printwriter
11. close printwriter

**Client.java Pseudocode**

1. Define as main method
2. initialize variables choice, sorted=false, csvfile and new Scanner object input
3. call StudentInfo()
4. ask user for student csv file
5. call importData() with user input as parameter
6. loop until choice equals 1
7. call menu()
8. if 2 is returned, call optionTwo()
9. if 3 is returned, call optionThree()
10. if 4 is returned, call optionFour()
11. if 5 is returned, call optionFive()
12. if 6 is returned, call optionSix()
13. if 7 is returned, call optionSeven()
14. if 8 is returned, call optionEight()
15. if 1 is returned, display farewell message and exit program

## Limitations

1. Program can only add marks to existing students in the arraylist (from the initial import data) but not add more students to the arraylist.
2. Other than that, I have implemented all functions as specified in the assignment.

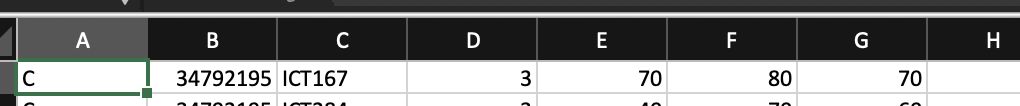
## Testing

| **Test ID** | **Test description/justification – what is the test for and why this particular test.** | **Actual data for this test** | **Expected output** | **Actual desk check result when desk check is carried out** | **Desk check outcome – Pass/Fail** |
| --- | --- | --- | --- | --- | --- |
| 1 | Test if overall mark is calculated correctly for coursework students. The calculation of overall mark is crucial as there is a set formula for the calculation. | Assignment1 70  Assignment2 80  FinalExam 70 | 73 | 73 | Pass |
| 2 | Test if overall mark is calculated correctly for research students. The calculation of overall mark is crucial as there is a set formula for the calculation. | Proposal 70  Dissertation 80 | 76 | 76 | Pass |
| 3 | Test if optionSeven() successfully sort the ArrayList. Without sorting the ArrayList, program will not allow outputting of the information to csv file. |  | Should sort according to student id in ascending order. | Successfully sorted in ascending order. | Pass |
| 4 | Test if optionSix() reports grades of student. It is part of the program’s requirement to be able to display a student’s grade based on student ID. | 34792195 | Steven Tay | Steven Tay | Pass |
| 5 | Test if optionTwo() deletes student and unit info from arraylist. It is part of the program’s requirement to be able to remove a student information based on student ID. | 34792195 | Unit and Student linked to 34792195 should be removed. | Units and Student is removed. | Pass |
| 6 | Test if optionEight() successfully outputs arraylist information to csv file. It is part of the program’s requirement to be able to output the details in the arraylist to a csv file. | output to output.csv | All details should be output | All details are output | Pass |

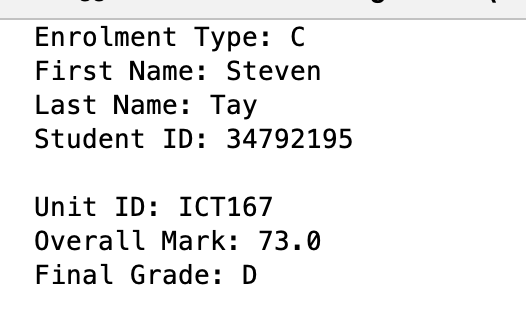
**Test ID 1**

Test if overall mark for coursework students are correctly calculated.

Formula = (assignment1\*0.3)+(assignment2\*0.3)+(finalexam\*0.4)



Student with mark of 70 for Assignment1, 80 for Assignment2, 70 for FinalExam is imported.



Program successfully calculated overall mark as 73.

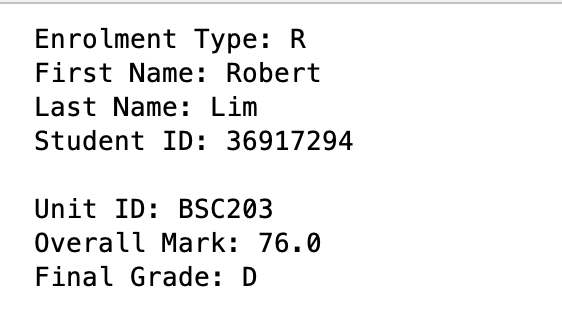
**Test ID 2**

Test if overall mark for research students are correctly calculated.

Formula = (proposal\*0.4)+(dissertation\*0.6)



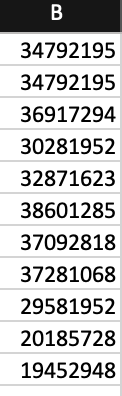
Student with 70 for Proposal and 80 for Dissertation is imported.



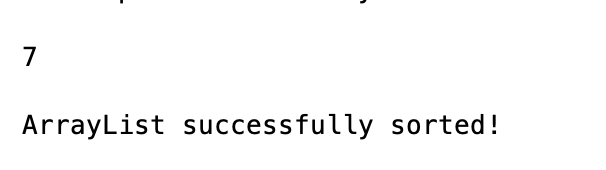
Program successfully calculated overall mark as 76.

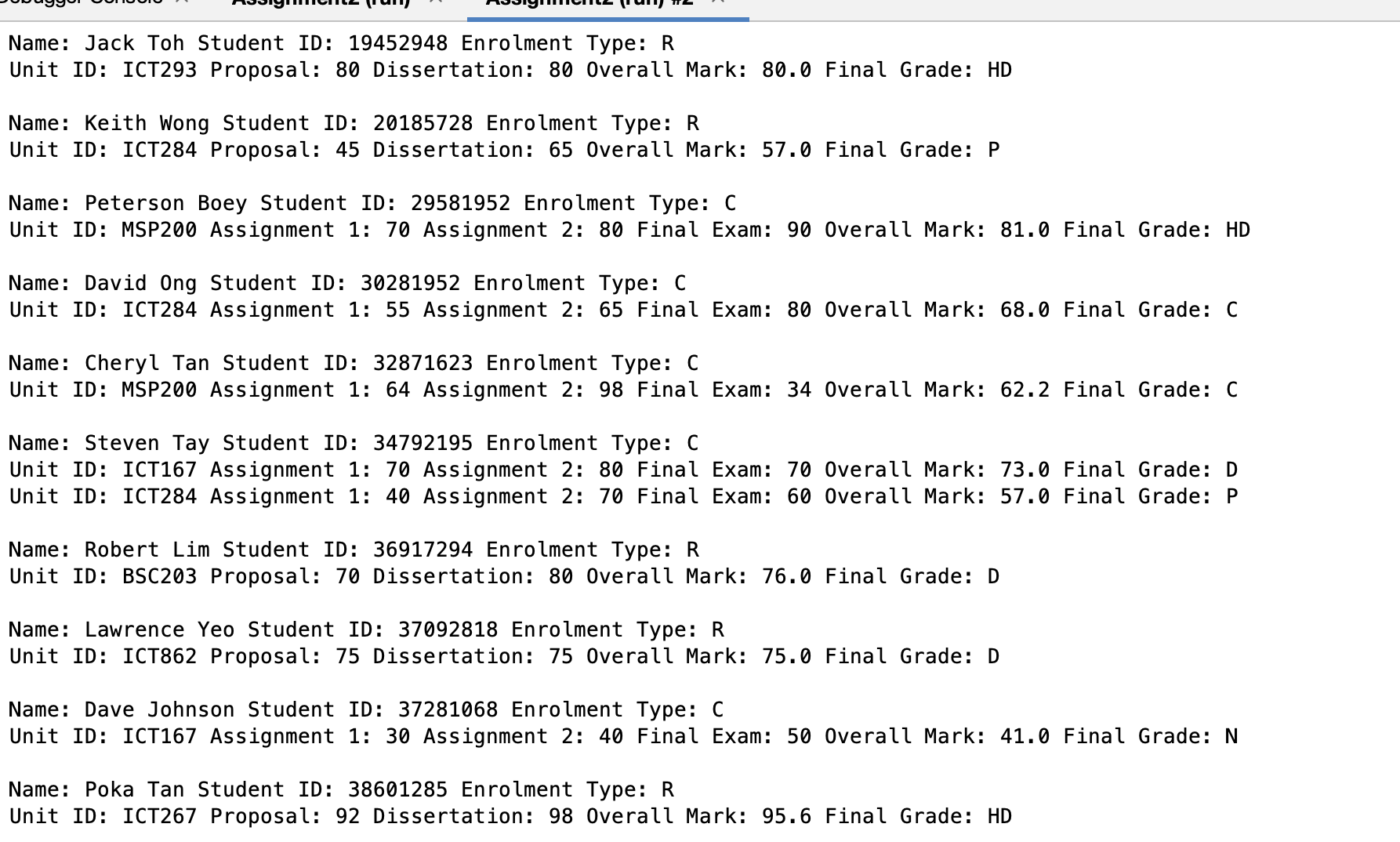
**Test ID 3**

Test if OptionSeven() successfully sorts Student ID.



Student with Student ID in this order has been imported.

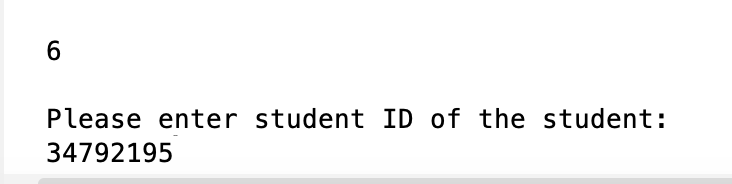




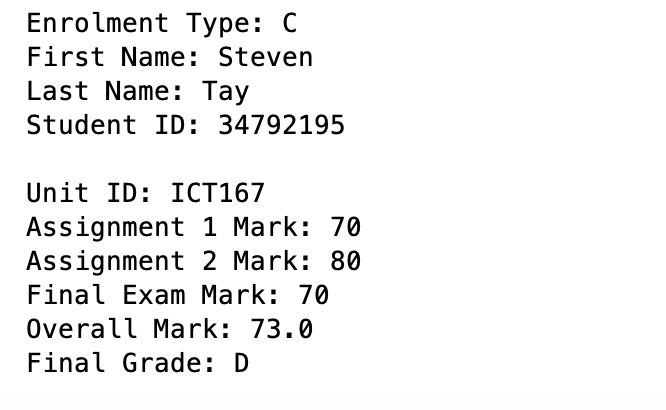
Students successfully sorted in ascending order of Student ID.

**Test ID 4**

Test if OptionSix() successfully display student’s grades based on student ID.



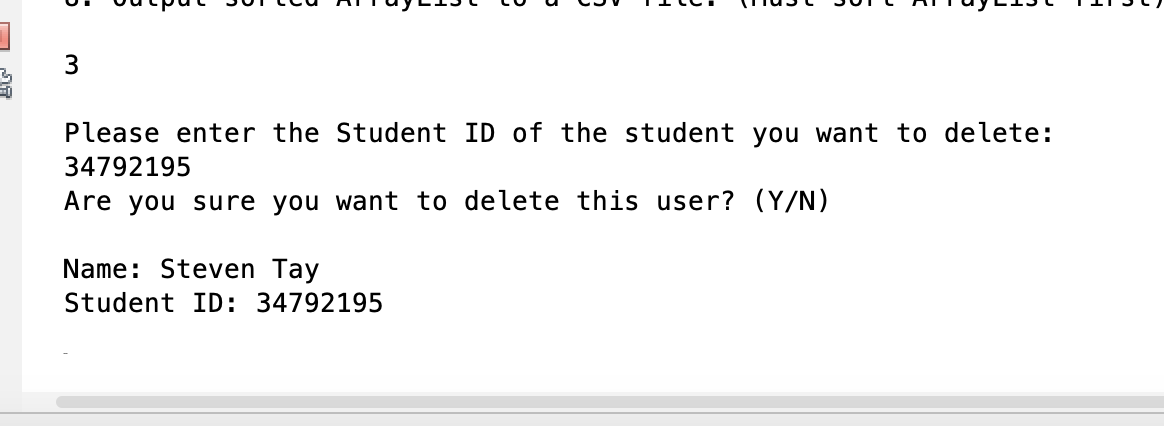
34792195 is entered as student ID.



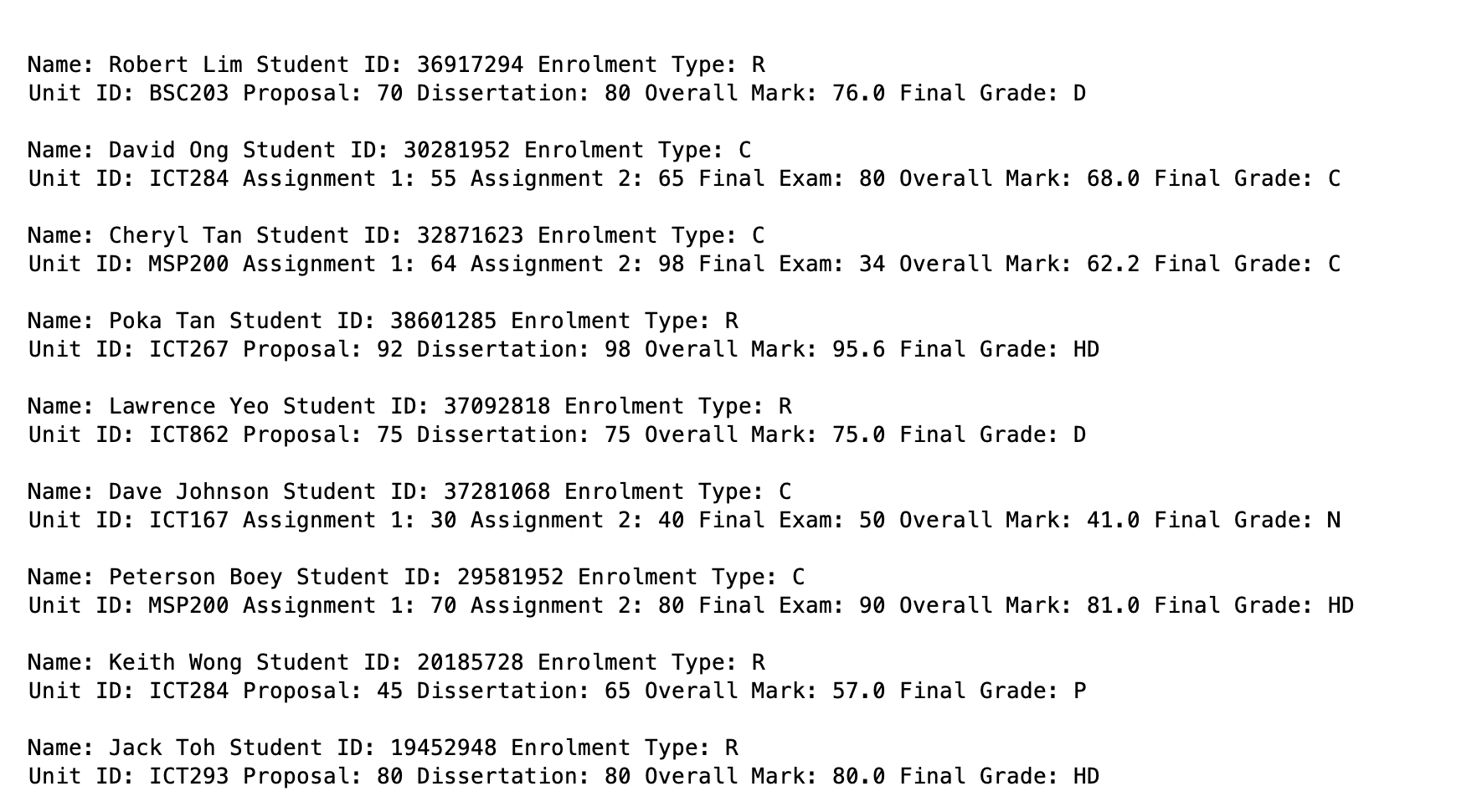
Student’s grades successfully displayed.

**Test ID 5**

Test if optionThree() successfully deletes students and relevant unit info from arraylist.



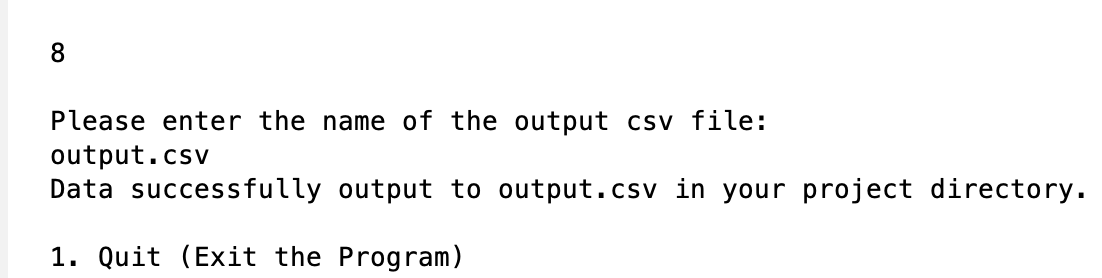
Program successfully searches for student based on student ID.



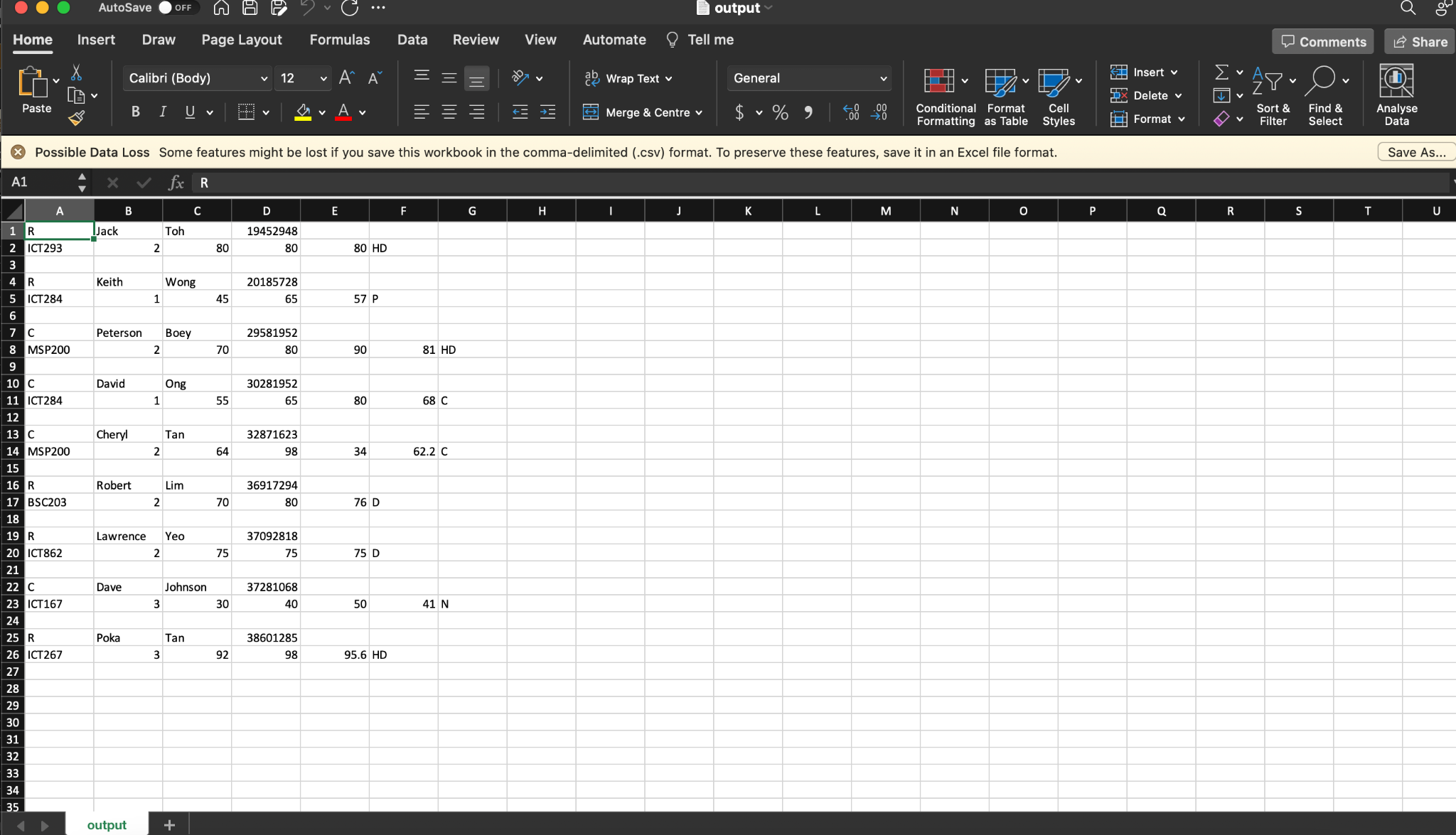
Student and Units belonging to the student successfully deleted from arraylist.

**Test ID 6**

Test if optionEight() successfully output the information in the arraylist to a csv file.



output.csv defined as the output file name.



Data successfully output to output.csv.

## Source Program Listing

**Student.java**

package assignment2;

import java.util.\*;

import java.io.\*;

/\*\*

\*

\* @author Ang Jin wei

\* @date 16 Mar 2023

\* @file Student.java

\* @description Student Class

\*/

public class Student {

private String firstname;

private String lastname;

private long studentid;

//default constructor

public Student(){

this.firstname = "NA";

this.lastname = "NA";

this.studentid = 0;

}

//parametized constructor

public Student(String first, String last, long id){

this.firstname = first;

this.lastname = last;

this.studentid = id;

}

//start of accessor methods

public String getFirst(){

return this.firstname;

}

public String getLast(){

return this.lastname;

}

public long getID(){

return this.studentid;

}

public String getEnrolmentType(){

return "NA";

}

//end of accessor methods

public String reportGrade(ArrayList Units){

return "There is no grade here.";

}

public boolean equals(Student obj){

boolean equal;

if (this.studentid == obj.getID()){

equal = true;

}

else{

equal = false;

}

if (obj == this){

equal = false;

}

return equal;

}

}

**Student\_Course.java**

package assignment2;

import java.util.\*;

import java.io.\*;

/\*\*

\*

\* @author Ang Jin wei

\* @date 16 Mar 2023

\* @file Student\_Course.java

\* @description Student\_Course, sub-class of Student Class

\*/

public class Student\_Course extends Student{

private String enrolmenttype;

public Student\_Course(String first, String last, long id){

super(first, last, id);

this.enrolmenttype = "C";

}

//start of accessor methods

public String getEnrolmentType(){

return this.enrolmenttype;

}

//end of accessor methods

//display information and grade of student

public String reportGrade(ArrayList Units){

String msg = "";

msg += "Enrolment Type: " + this.getEnrolmentType() + "\n";

msg += "First Name: " + this.getFirst() + "\n";

msg += "Last Name: " + this.getLast() + "\n";

msg += "Student ID: " + this.getID() + "\n";

for (int i=0; i<Units.size(); i++){

Unit temp = (Unit) Units.get(i); //cast temp as Unit type object

if (this.getID() == temp.getID()){

msg += "\nUnit ID: " + temp.getUnit() + "\n";

msg += "Assignment 1 Mark: " + temp.getAssignment1() + "\n";

msg += "Assignment 2 Mark: " + temp.getAssignment2() + "\n";

msg += "Final Exam Mark: " + temp.getFinal() + "\n";

msg += "Overall Mark: " + temp.getOverall() + "\n";

msg += "Final Grade: " + temp.finalreport() + "\n";

}

}

return msg;

}

}

**Student\_Research.java**

package assignment2;

import java.util.\*;

import java.io.\*;

/\*\*

\*

\* @author Ang Jin wei

\* @date 16 Mar 2023

\* @file Student\_Research.java

\* @description Student\_Research, sub-class of Student Class

\*/

public class Student\_Research extends Student{

private String enrolmenttype;

public Student\_Research(String first, String last, long id){

super(first, last, id);

this.enrolmenttype = "R";

}

//start of accessor methods

public String getEnrolmentType(){

return this.enrolmenttype;

}

//end of accessor methods

//display information and grade of student

public String reportGrade(ArrayList Units){

String msg = "";

msg += "Enrolment Type: " + this.getEnrolmentType() + "\n";

msg += "First Name: " + this.getFirst() + "\n";

msg += "Last Name: " + this.getLast() + "\n";

msg += "Student ID: " + this.getID() + "\n";

for (int i=0; i<Units.size(); i++){

Unit temp = (Unit) Units.get(i); //cast temp as Unit type object

if (this.getID() == temp.getID()){

msg += "\nUnit ID: " + temp.getUnit() + "\n";

msg += "Proposal: " + temp.getProposal() + "\n";

msg += "Dissertation: " + temp.getDissertation() + "\n";

msg += "Overall Mark: " + temp.getOverall() + "\n";

msg += "Final Grade: " + temp.finalreport() + "\n";

}

}

return msg;

}

}

**Unit.java**

package assignment2;

/\*\*

\*

\* @author Ang Jin wei

\* @date 16 Mar 2023

\* @file Unit.java

\* @description Unit Class

\*/

public class Unit {

private String unitid;

private long studentid;

private int level;

private String enrolmenttype;

//default constructor

public Unit(){

this.unitid = "NA";

this.studentid = 0;

this.level = 0;

this.enrolmenttype = "NA";

}

//parametized constructor

public Unit(long studentid, String unitid, int level, String enrolmenttype){

this.studentid = studentid;

this.unitid = unitid;

this.level = level;

this.enrolmenttype = enrolmenttype;

}

//start of accessor methods

public String getUnit(){

return this.unitid;

}

public long getID(){

return this.studentid;

}

public int getLevel(){

return this.level;

}

public int getAssignment1(){

return 0;

}

public int getAssignment2(){

return 0;

}

public int getFinal(){

return 0;

}

public int getDissertation(){

return 0;

}

public int getProposal(){

return 0;

}

public String getEnrolmentType(){

return this.enrolmenttype;

}

public double getOverall(){

return 0;

}

public String finalreport(){

return "NA";

}

//end of accessor methods

public void calculateOverall(){

}

public void calculateFinalGrade(){

}

}

**Unit\_Course.java**

package assignment2;

/\*\*

\*

\* @author Ang Jin wei

\* @date 16 Mar 2023

\* @file Unit\_Course.java

\* @description Unit\_Course, sub-class of Unit class

\*/

public class Unit\_Course extends Unit{

private int assignment1;

private int assignment2;

private int finalexam;

private double overall;

private String finalGrade;

public Unit\_Course(long studentid, String unitid, int level, String enrolmenttype, int first, int second, int last){

super(studentid, unitid, level, "C");

this.assignment1 = first;

this.assignment2 = second;

this.finalexam = last;

this.overall = 0;

this.finalGrade = "";

}

//start of accessor methods

public int getAssignment1(){

return this.assignment1;

}

public int getAssignment2(){

return this.assignment2;

}

public int getFinal(){

return this.finalexam;

}

public double getOverall(){

return this.overall;

}

public String finalreport(){

return this.finalGrade;

}

//end of accessor methods

public void calculateOverall(){

double assignment;

double result;

assignment = (this.assignment1\*0.3)+(this.assignment2\*0.3); //calculate assignment mark 30% each of overall

result = assignment + (this.finalexam\*0.4); //calculate final exam mark 40% of overall

result = Math.round(result\*100.0)/100.0; //round up to nearest decimal

this.overall = result;

}

//calculate final grade

// HD = 80 or above

// D = 70 or above and below 80

// C = 60 or above and below 70

// P = 50 or above and below 60

// N = below 50

public void calculateFinalGrade(){

if (this.getOverall() >= 80){

this.finalGrade = "HD";

}

if (this.getOverall() >= 70 && this.getOverall() < 80){

this.finalGrade = "D";

}

if (this.getOverall() >= 60 && this.getOverall() < 70){

this.finalGrade = "C";

}

if (this.getOverall() >= 50 && this.getOverall() < 60){

this.finalGrade = "P";

}

if (this.getOverall() < 50){

this.finalGrade = "N";

}

}

}

**Research.java**

package assignment2;

/\*\*

\*

\* @author Ang Jin wei

\* @date 16 Mar 2023

\* @file Research.java

\* @description Research, sub-class of Unit Class

\*/

public class Research extends Unit{

private int proposal;

private int dissertation;

private double overall;

private String finalGrade;

public Research(long studentid, String unitid, int level, String enrolmenttype, int first, int second){

super(studentid, unitid, level, "R");

this.proposal = first;

this.dissertation = second;

this.overall = 0;

}

//start of accessor methods

public int getProposal(){

return this.proposal;

}

public int getDissertation(){

return this.dissertation;

}

public double getOverall(){

return this.overall;

}

public String finalreport(){

return this.finalGrade;

}

//end of accessor methods

public void calculateOverall(){

double result;

result = (this.proposal\*0.4) + (this.dissertation\*0.6); //calculate proposal mark 40% and disseration 60%

result = Math.round(result\*100.0)/100.0; //round up to nearest decimal

this.overall = result;

}

//calculate final grade

// HD = 80 or above

// D = 70 or above and below 80

// C = 60 or above and below 70

// P = 50 or above and below 60

// N = below 50

public void calculateFinalGrade(){

if (this.getOverall() >= 80){

this.finalGrade = "HD";

}

if (this.getOverall() >= 70 && this.getOverall() < 80){

this.finalGrade = "D";

}

if (this.getOverall() >= 60 && this.getOverall() < 70){

this.finalGrade = "C";

}

if (this.getOverall() >= 50 && this.getOverall() < 60){

this.finalGrade = "P";

}

if (this.getOverall() < 50){

this.finalGrade = "N";

}

}

}

**Client.java**

package assignment2;

import java.util.\*;

import java.io.\*;

/\*\*

\*

\* @author Ang Jin wei

\* @date 16 Mar 2023

\* @file Client.java

\* @description Allow entry of data for several students into ArrayList and perform analysis and queries.

\*/

public class Client {

private static ArrayList<Student> Students = new ArrayList<>();

private static ArrayList<Unit> Units = new ArrayList<>();

private static int count;

private static int unitcount;

public static void main(String[] args) {

int choice = 0;

boolean sorted = false;

String csvfile;

Scanner input = new Scanner(System.in);

StudentInfo(); //print student info

System.out.println("Please enter the name of the student csv file: ");

csvfile = input.nextLine();

importData(csvfile); //import data from csvfile

//display menu and loop till user chooses 1

while (choice != 1){

choice = menu();

if (choice == 2){

optionTwo();

}

if (choice == 3){

optionThree();

}

if (choice == 4){

optionFour();

}

if (choice == 5){

optionFive();

}

if (choice == 6){

optionSix();

}

if (choice == 7){

optionSeven();

sorted = true;

}

if (choice == 8){

optionEight(sorted);

}

if (choice == 1){

System.out.println("Thanks for using, goodbye!");

System.exit(0); //terminate program

}

}

}

//menu method, print out menu and return choice of user's input

private static int menu(){

int choice;

Scanner input = new Scanner(System.in); //define scanner

System.out.println("\n1. Quit (Exit the Program)");

System.out.println("2. Add all the marks information about a course work student or a research student by reading it from another CSV file. ");

System.out.println("3. Remove specific student and relevant information from the ArrayList.");

System.out.println("4. Output all details currently held in array list.");

System.out.println("5. Determine and display how many course work students obtained an overall mark equal to or above the average overall and how many obtained an overall mark below the average overall mark.");

System.out.println("6. Report grade of a student.");

System.out.println("7. Sort ArrayList in ascending order based on student ID.");

System.out.println("8. Output sorted ArrayList to a CSV file. (Must sort ArrayList first)\n");

choice = input.nextInt();

input.nextLine();

return choice; //return user choice

}

//import data from csv file

//precondition: filename must be defined

//postcondition: method will run through csv file and add data to arraylist

private static void importData(String filename){

try{

Scanner readFile = new Scanner(new File(filename));

while(readFile.hasNextLine()){ //check if file has next line

String[] line = readFile.nextLine().split(",");

if (line[0].equalsIgnoreCase("C")){

Students.add(new Student\_Course(line[1],line[2],Long.parseLong(line[3])));

}

if (line[0].equalsIgnoreCase("R")){

Students.add(new Student\_Research(line[1],line[2],Long.parseLong(line[3]))); //add student object to arraylist

}

count+=1; //add 1 to count

}

System.out.println("Data successfully imported!");

}

catch(FileNotFoundException e){ //catch filenotfound exception

System.out.println("Can't open file "+filename+", please try again, the program will exit now.");

System.exit(0);

}

}

//optionTwo method: add data to arraylist from csv file specified by user

//precondition: there must be an existing csv file and filename should be defined

//postcondition: method will run through csv file and add data to the arraylist

private static void optionTwo(){

String filename;

Scanner input = new Scanner(System.in);

System.out.println("\nPlease enter the name of the marks csv file: "); //ask user for csv filename

filename = input.nextLine();

try{

Scanner readFile = new Scanner(new File(filename));

while(readFile.hasNextLine()){

String[] line = readFile.nextLine().split(","); //split csv into array

if (line[0].equalsIgnoreCase("C")){ //check if student is coursework

//add details to arraylist

Units.add(new Unit\_Course(Long.parseLong(line[1]),line[2],Integer.parseInt(line[3]),line[0],Integer.parseInt(line[4]),Integer.parseInt(line[5]),Integer.parseInt(line[6])));

Units.get(unitcount).calculateOverall();;

Units.get(unitcount).calculateFinalGrade();

}

if (line[0].equalsIgnoreCase("R")){ //check if student is research

//add details to arraylist

Units.add(new Research(Long.parseLong(line[1]),line[2],Integer.parseInt(line[3]),line[0],Integer.parseInt(line[4]),Integer.parseInt(line[5])));

Units.get(unitcount).calculateOverall();;

Units.get(unitcount).calculateFinalGrade();

}

unitcount += 1; //add 1 to count

}

System.out.println("\nData successfully added to the ArrayList.");

}

catch(FileNotFoundException e){

System.out.println("\nCan't open file "+filename);

}

}

//optionThree method: get user input for student ID and delete the object with the specific student ID

//precondition: studentid should exist in the arraylist

//postcondition: run through arraylist to see if student id exist, if so ask user for confirmation and delete the object

private static void optionThree(){

boolean found = false;

long delete;

String choice = "";

Scanner input = new Scanner(System.in);

System.out.println("\nPlease enter the Student ID of the student you want to delete: "); //ask user for student id

delete = input.nextLong();

input.nextLine();

for (int i=0; i<Students.size(); i++){

if (Students.get(i).getID() == delete){ //check if student id equals to user specified id

found = true;

System.out.println("Are you sure you want to delete this user? (Y/N)\n"); //ask user for confirmation

System.out.println("Name: " + Students.get(i).getFirst() + " " + Students.get(i).getLast());

System.out.println("Student ID: " + Students.get(i).getID()+"\n");

choice = input.nextLine();

if (choice.equalsIgnoreCase("Y")){ //check if user entered Y

Students.remove(i); //remove student

count -= 1;

for (int j=0; j<Units.size(); j++){

if (Units.get(j).getID() == delete){ //check if unit belongs to particular student

Units.remove(j); //remove unit

unitcount -= 1;

j-=1;

}

}

}

}

}

if (found == false){

System.out.println("Student ID not found!");

}

}

//optionFour method: display details held in array list

//precondition: arraylist should be defined

//postcondition: run through arraylist and display details in arraylist

private static void optionFour(){

String msg = "";

for (int i=0; i<Students.size(); i++){ //loop through arraylist

msg += "\n\nName: " + Students.get(i).getFirst() + " " + Students.get(i).getLast();

msg += " Student ID: " + Students.get(i).getID();

msg += " Enrolment Type: " + Students.get(i).getEnrolmentType();

if (Students.get(i).getEnrolmentType().equalsIgnoreCase("C")){ //check if student is coursework

for (int j=0; j<Units.size(); j++){

if (Units.get(j).getID() == Students.get(i).getID()){ //check if student id equal

//add details to msg

msg += "\nUnit ID: " + Units.get(j).getUnit();

msg += " Assignment 1: " + Units.get(j).getAssignment1();

msg += " Assignment 2: " + Units.get(j).getAssignment2();

msg += " Final Exam: " + Units.get(j).getFinal();

msg += " Overall Mark: " + Units.get(j).getOverall();

msg += " Final Grade: " + Units.get(j).finalreport();

}

}

}

if (Students.get(i).getEnrolmentType().equalsIgnoreCase("R")){ //check if student is research

for (int j=0; j<Units.size(); j++){

if (Units.get(j).getID() == Students.get(i).getID()){ //check if student id equal

//add details to msg

msg += "\nUnit ID: " + Units.get(j).getUnit();

msg += " Proposal: " + Units.get(j).getProposal();

msg += " Dissertation: " + Units.get(j).getDissertation();

msg += " Overall Mark: " + Units.get(j).getOverall();

msg += " Final Grade: " + Units.get(j).finalreport();

}

}

}

}

System.out.println(msg);

}

//optionFive method: run through arraylist and determine amount of student with higher or lower average mark for coursework students

//precondition: arraylist must be defined, student must be coursework student

//postcondition: display number of students with overall mark lower and above the average overall mark

private static void optionFive(){

int total = 0;

int count = 0;

double average = 0;

int above = 0;

int below = 0;

String abovestudent = "";

String belowstudent = "";

for (int i=0; i<Units.size(); i++){

if (Units.get(i).getEnrolmentType().equalsIgnoreCase("C")){ //check if unit is coursework unit

total += Units.get(i).getOverall(); //add overall score to total score

count += 1; //add 1 to count

}

}

average = total/count; //calculate average totaloverallscore divided by number of coursework students

for (int i=0; i<Units.size(); i++){

if (Units.get(i).getOverall() > average && Units.get(i).getEnrolmentType().equalsIgnoreCase("C")){ //check if overall score higher than average

above += 1; //add 1 to above average

for (int j=0; j<Students.size(); j++){

if (Units.get(i).getID() == Students.get(j).getID()){

//add particular student details

abovestudent += "Name: " + Students.get(j).getFirst() + " " + Students.get(j).getLast() + "\n";

abovestudent += "Student ID: " + Students.get(j).getID() + "\n";

abovestudent += "Unit ID: " + Units.get(i).getUnit() + "\n";

abovestudent += "Overall Mark: " + Units.get(i).getOverall() + "\n\n";

}

}

}

if (Units.get(i).getOverall() < average && Units.get(i).getEnrolmentType().equalsIgnoreCase("C")){ //check if overall score below than average

below += 1; //add 1 to below average

for (int j=0; j<Students.size(); j++){

if (Units.get(i).getID() == Students.get(j).getID()){

//add particular student details

belowstudent += "Name: " + Students.get(j).getFirst() + " " + Students.get(j).getLast() + "\n";

belowstudent += "Student ID: " + Students.get(j).getID() + "\n";

belowstudent += "Unit ID: " + Units.get(i).getUnit() + "\n";

belowstudent += "Overall Mark: " + Units.get(i).getOverall() + "\n\n";

}

}

}

}

System.out.println("\nThe average score is " + average);

System.out.println("There are " + above + " students that got above the average score.");

System.out.println("There are " + below + " students taht got below the average score.\n");

System.out.println("The students who got above the average score are: \n" + abovestudent);

System.out.println("The students who got below the average score are: \n" + belowstudent);

}

//optionSix method: ask user for input on student ID and report grade of the particular student

//precondition: arraylist must be defined and student ID must exist in arraylist

//postcondition: grades of the particular student will be displayed

private static void optionSix(){

long id;

boolean found = false;

Scanner input = new Scanner(System.in);

System.out.println("\nPlease enter student ID of the student: "); //ask user for student ID

id = input.nextLong();

for (int i=0; i<Students.size(); i++){

if (Students.get(i).getID() == id){ //loop through arraylist and check if student id is found

System.out.println("\n"+Students.get(i).reportGrade(Units)); //report grade of the particular student

found = true;

}

}

if (found == false){

System.out.println("\nStudent ID not found."); //if student id isn't found, display not found message

}

}

//optionSeven method: use bubble sort to sort students in arraylist in ascending order based on student ID

//precondition: arraylist must be defined

//postcondition: arraylist will be sorted in ascending order

private static void optionSeven(){

Student temp;

for (int i=0; i<Students.size(); i++){

for (int j=0; j<Students.size()-i-1; j++){

if (Students.get(j).getID() > Students.get(j+1).getID()){

temp = Students.get(j);

Students.set(j,Students.get(j+1));

Students.set(j+1, temp);

}

}

}

System.out.println("\nArrayList successfully sorted!");

}

//optionEight method: output arraylist to csv file defined by user

//precondition: arraylist must be defined, arraylist must be sorted

//postcondition: output the arraylist to csv file

private static void optionEight(boolean sorted){

String outputfile = "";

PrintWriter output = null; //define null printwriter

String data = "";

Scanner input = new Scanner(System.in); //define scanner

if (sorted == true){

System.out.println("\nPlease enter the name of the output csv file: "); //ask user for output file name

outputfile = input.nextLine();

//start of try-block

try{

output = new PrintWriter(new File(outputfile));

for (int i=0; i < Students.size(); i++){

if (Students.get(i).getEnrolmentType().equalsIgnoreCase("C")){ //check if student is coursework student

//add data from arraylist to data variable

data += Students.get(i).getEnrolmentType() + ",";

data += Students.get(i).getFirst() + ",";

data += Students.get(i).getLast() + ",";

data += Students.get(i).getID() + "\n";

for (int j=0; j < Units.size(); j++){

if (Units.get(j).getID() == Students.get(i).getID()){

data += Units.get(j).getUnit() + ",";

data += Units.get(j).getLevel() + ",";

data += Units.get(j).getAssignment1() + ",";

data += Units.get(j).getAssignment2() + ",";

data += Units.get(j).getFinal() + ",";

data += Units.get(j).getOverall() + ",";

data += Units.get(j).finalreport() + "\n\n";

}

}

}

if (Students.get(i).getEnrolmentType().equalsIgnoreCase("R")){ //check if student is research student

//add data from arraylist to datavariable

data += Students.get(i).getEnrolmentType() + ",";

data += Students.get(i).getFirst() + ",";

data += Students.get(i).getLast() + ",";

data += Students.get(i).getID() + "\n";

for (int j=0; j < Units.size(); j++){

if (Units.get(j).getID() == Students.get(i).getID()){

data += Units.get(j).getUnit() + ",";

data += Units.get(j).getLevel() + ",";

data += Units.get(j).getProposal() + ",";

data += Units.get(j).getDissertation() + ",";

data += Units.get(j).getOverall() + ",";

data += Units.get(j).finalreport() + "\n\n";

}

}

}

}

output.write(data); //write data to csv file

System.out.println("Data successfully output to "+outputfile+" in your project directory.");

output.close();

}

//end of try-block

catch(FileNotFoundException e){

System.out.println("Can't open file "+outputfile);

}

}

else{

System.out.println("Please sort the ArrayList first!");

}

}

//StudentInfo method: Print out student's information

public static void StudentInfo(){

String msg = "";

msg += "Name: Ang Jin Wei";

msg += "\nStudent Number: 34792195";

msg += "\nMode of Enrolment: Part-Time";

msg += "\nTutor Name: Steven Loke";

msg += "\nClass: ICT284 B";

System.out.println(msg+"\n");

}

}