Assessment details COIT20245

Assessment item 2—JAVA Program using array of objects

Due date:	**Week 9 Friday (16 Sept 2022) 11:59 pm AEST**	**ASSESSMENT**
	Refer below for complete assessment item 2 requirements (**Assignment Two**)	
Weighting:	30%	
Length:	N/A	**2**

Objectives

This assessment item relates to the unit learning outcomes as stated in the Unit Profile.

Details

For this assignment, you are required to develop a **Menu Driven Console Java Program** to demonstrate you can use Java constructs including input/output via the console, Java primitive and built-in types, Java defined objects, arrays, selection and looping statements and various other Java commands. Your program must produce the correct results.
The code for the menu and option selection is supplied: GradingSystemMenu.java and is available on the unit website, you must write the underlying code to implement the program. The menu selections are linked to appropriate methods in the given code. Please spend a bit of time looking at the given code to familiarise yourself with it and where you have to complete the code. You will need to write comments in the supplied code as well as your own additions. You will submit your files by the due date using the "**Assignment 2**" link on the Moodle unit website under **Assessment … Assignment 2 Submission**.

What to submit for this assignment?

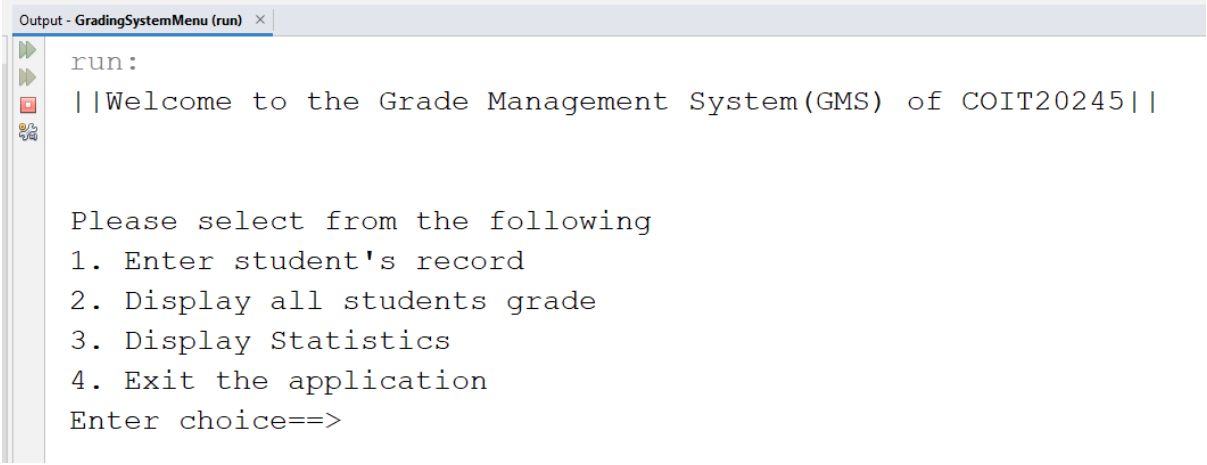
- It is a requirement that you use Netbeans for your final submission. Submit a zip of your Netbeans project folder as **Ass2.zip**
- **ReportAss2.docx -** A report including
    - Explain in detail each statement that interfaces Menu (getMenuItem), processing (processingGradeingSystem) and main method together. Show the linking using a flow chart.
    - Explain in detail each statement that interfaces between Student and GradingSystemMenu class. Show the linking using a flow chart.
    - An UML class diagram of your student class,
    - how long it took to create the whole program,
    - any problems encountered, and screenshots of the output produced including annotations.

(Use Alt-PrtScrn to capture just the application window and you can paste it into your Word document) You should test every possibility in the program.

Assignment Specification

You have completed the console program for processing grade of students for COIT20245. We are going to extend this application so the students name, student number, marks and grades can be stored in an array of objects, **do not use ArrayList**.

The program will run via a menu of options, the file **GradingSystemMenu.java** has been supplied (via the Moodle web site) which supplies the basic functionality of the menu system.



Look at the code supplied and trace the execution and you will see the menu is linked to blank methods (stubs) which you will implement the various choices in the menu.

Student class
First step is to create a class called Student (Student.java).

The Student class will be very simple it will contain seven private instance variables:

- studentName as a String
- studentID as a String
- assignmentOneMarks as double
- assignmentTwoMarks as double
- projectMark as double
- individualTotalMarks as double
- grade as a String

The numeric literal values, like P=50.00, HD=85.00 ***must*** be represented as **constants**.

The following public methods will have to be implemented:

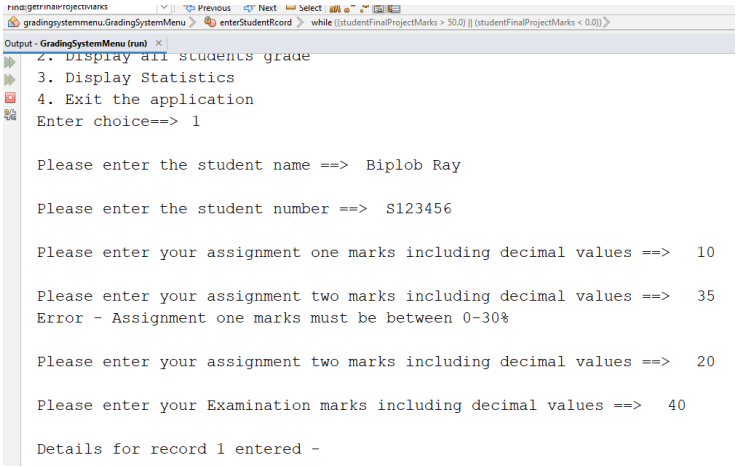
- A default constructor
- A parameterised constructor
- Five set methods (mutators)
- Five get methods (accessors)
- A method to calculate total marks and return student's total marks as double – calculateIndividualTotalMarks(). This calculation will be the same as in assignment one.
- A method to calculate grade and return the grade as String – calculateGrade(). This calculation will be the same as in assignment one. Use constants for all numeric literals.

Note: Following basic database principles, calculated values are not usually stored, so in this case we will not store the grade as a instance variable, but use the calculateGrade() method when we want to determine the grade.

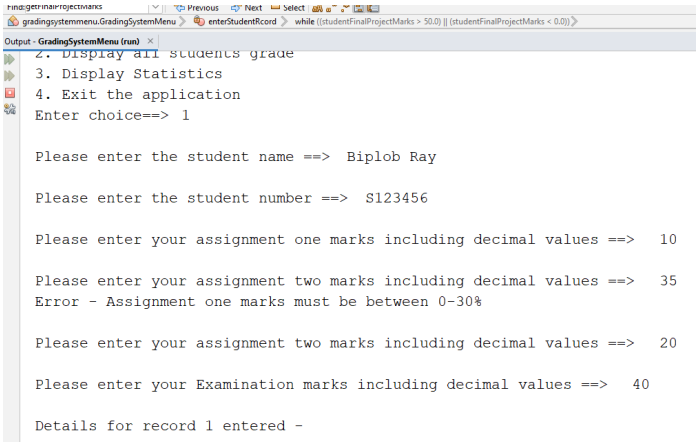
GradingSystemMenu class
Once the Student class is implemented and fully tested we can now start to implement the functionality of the menu system.

Data structures
For this assignment we are going to store the student's name, student number and assessment marks an array of Student objects.
Declare an array of Student objects as an instance variable of GradingSystemMenu class the array should hold **ten** students.
You will need another instance variable (integer) to keep track of the number of the students being entered and use this for the index into the array of Student objects.

Menu options
1. **Enter students name, student number and assessment marks**: `enterStudentRcord()`
You will read in the student's name, student number and assessment marks as you did in assignment one.

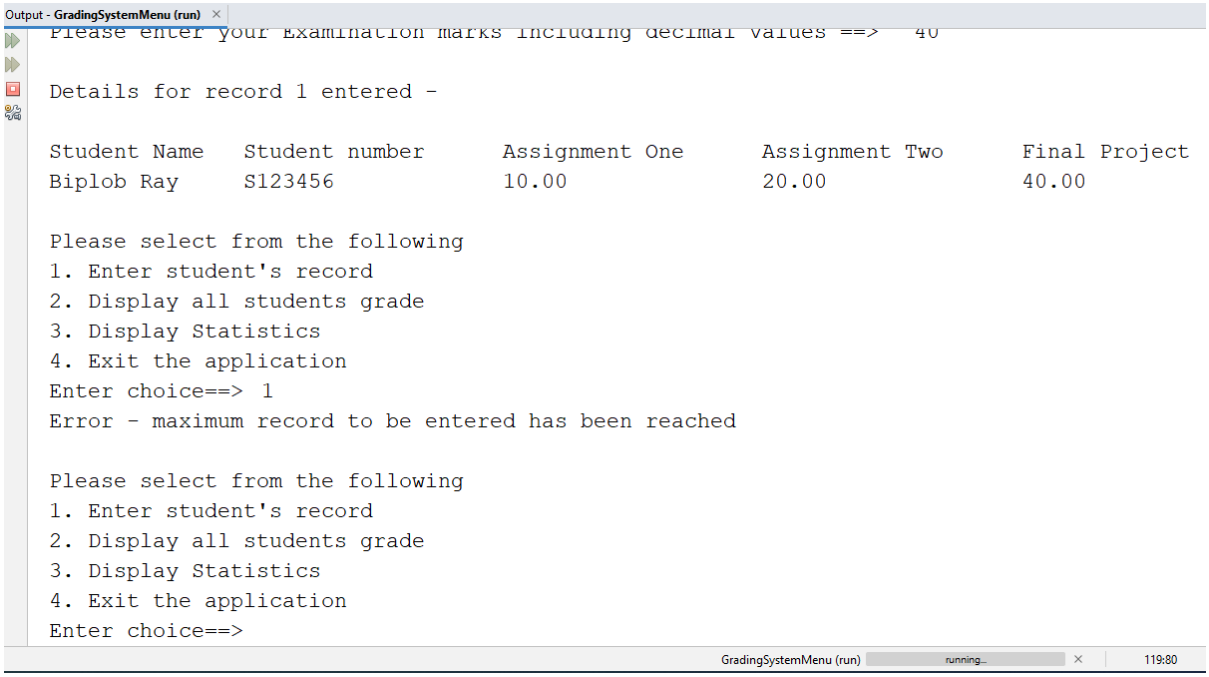


Data validation (you can implement this after you have got the basic functionality implemented)
You will need to validate the user input using a **validation loop**.
The student's name and student number cannot be blank i.e. not null and the assessments marks needs to be within the range of (0-assessment weighting), the same as assignment one.

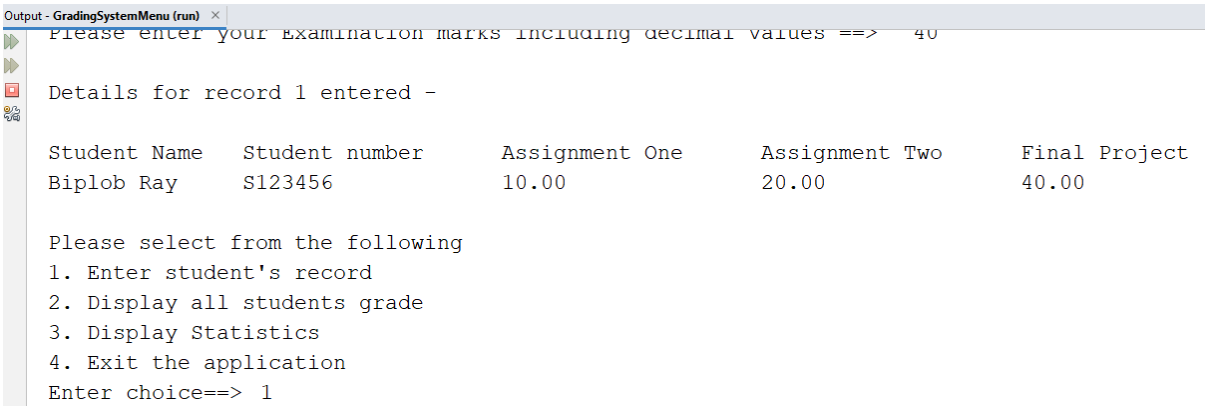


When entering record of student's name, student number and assessments marks, the student have been entered successfully into five local variables you will need to add these values into the student object array, you will also need to increment a counter to keep track of the number of students you have entered and the position in the array of the next student to be entered.

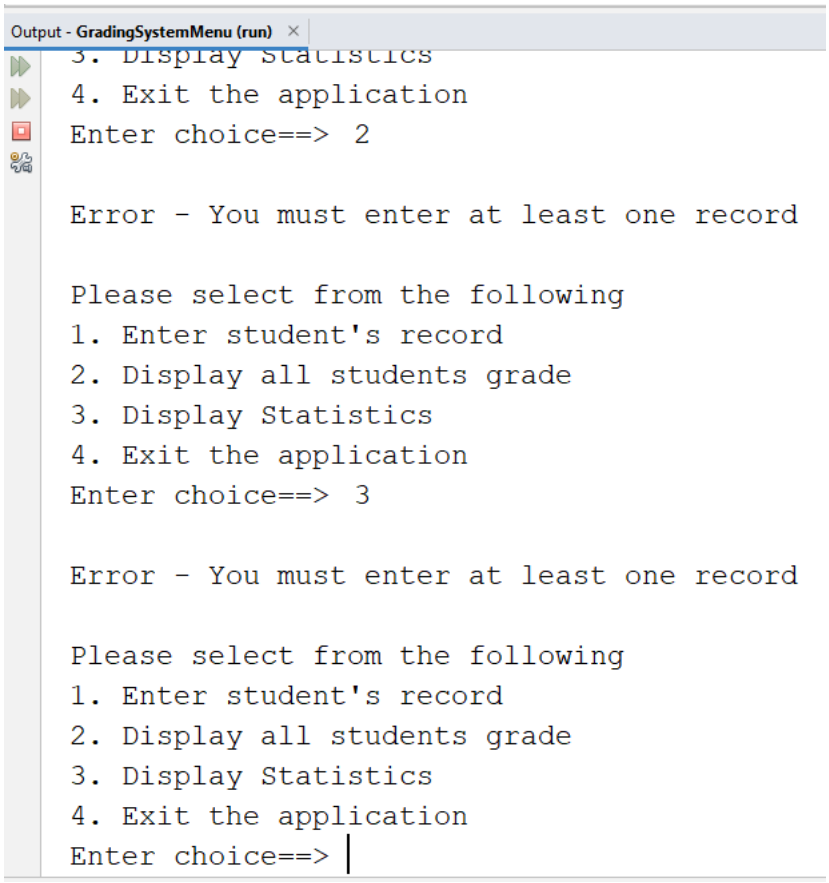
When the maximum number of students record is reached do not attempt to add any more student's record and give the following error message:



When the student details have been successfully entered, display the details of the student and the charge as follows:



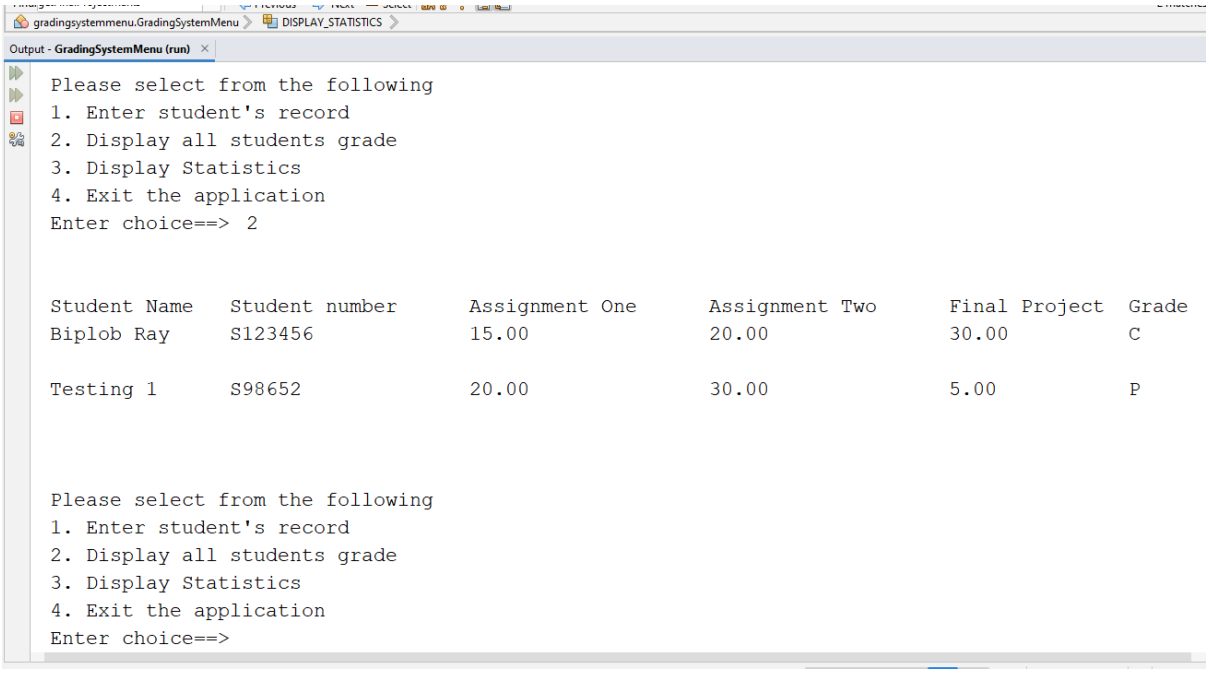
Note: For the next two options, display all and statistics you should ensure at least one student's record has been entered and give an appropriate error message if it there are no students record entered.



2. **Display all student's name, student number, assessment marks and grade**:

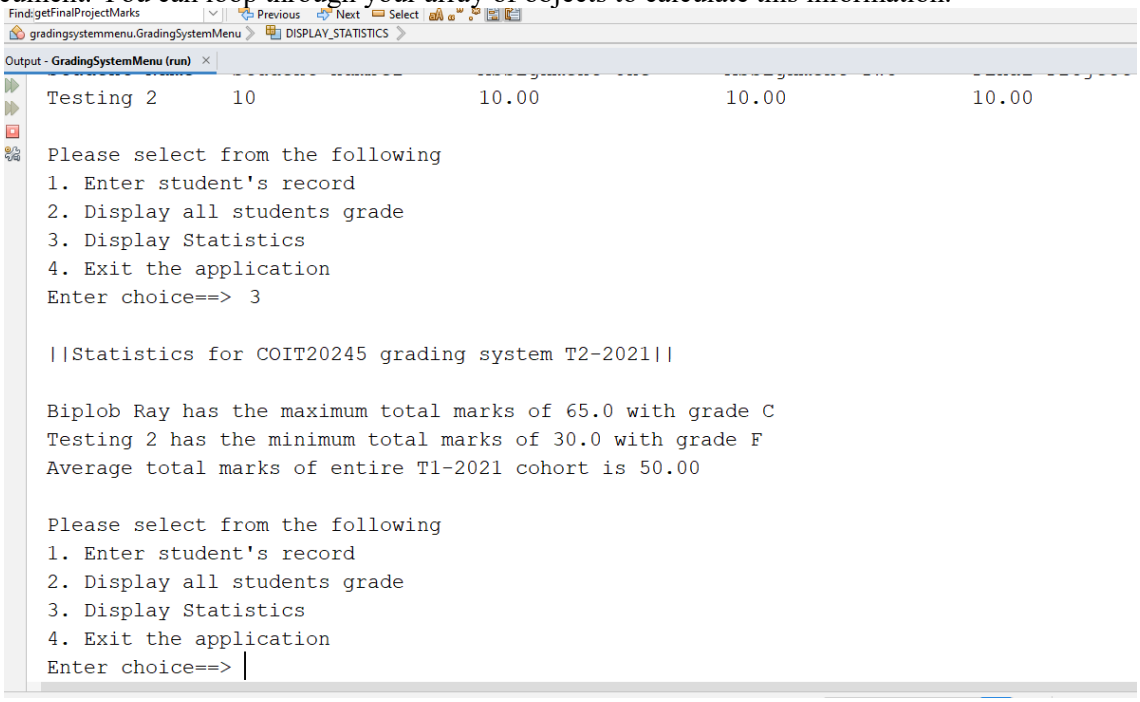
`displayAllRecordsWithGrade()`

When this option is selected display all the records with grade which have been entered so far.

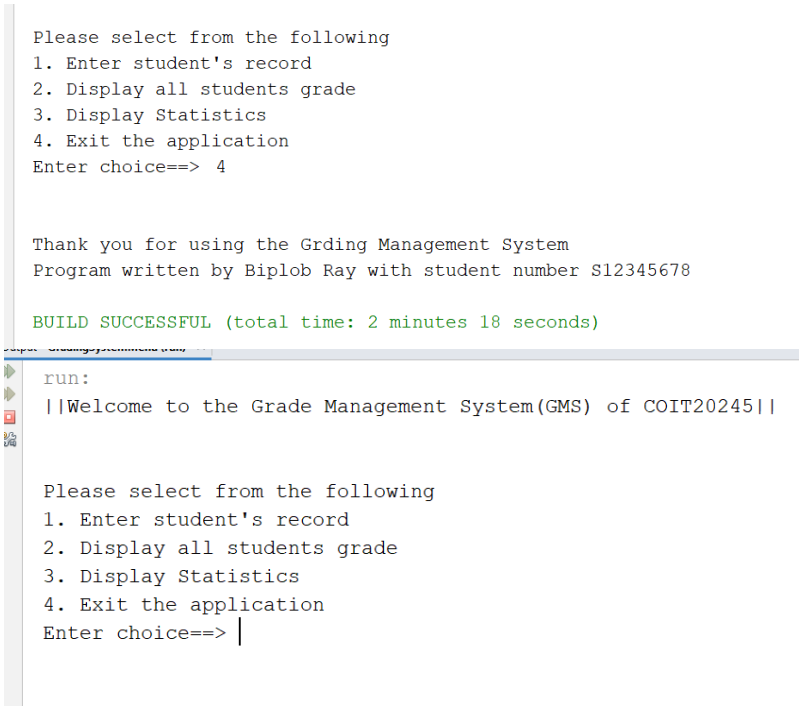


3. **Display statistics**: `displayStatistics()`

When this option is selected you will display the statistics as per detailed in assignment one document. You can loop through your array of objects to calculate this information.



Remember the welcome and exit messages as per assignment one.



Extra Hints

Your program should be well laid out, commented and uses appropriate and consistent names (camel notation) for all variables, methods and objects.
Make sure you have no repeated code (even writing headings in the output)
Constants must be used for all numbers (numeric literals) in your code.
Look at the marking criteria to ensure you have completed all of the necessary items.
Refer to a Java reference textbook and the unit and lecture material (available on the unit web site) for further information about the Java programming topics required to complete this assignment.
Check output, check code and add all of your comments, complete report and the UML class diagram.

Supplied Code

Download, compile and run the supplied code available from the unit web site.
You will see the menu interface has been implemented and you have to implement the underlying code, use the supplied method stubs and add your own methods.
Follow the `// TODO` comments in the supplied code.
Again no code should be repeated in your program.

If you just submit the supplied code you will receive zero marks.

Good luck! Dr Biplob Ray Unit Coordinator T222 COIT20245

b.ray@cqu.edu.au

Marking scheme is on the following page.

Marking Criteria	Marks allocated	Marks Received	Comments
Variables, constants, and types			
Variables have meaningful names and use camel notation	0.5		
Variables are the correct type and constants are used	0.5		
Array of objects is used	1		
Code in general			
Code is indented and aligned correctly	1		
Code is easy to read (use of vertical whitespace)	0.5		
Code has header comment which includes name, student ID, date, file name and purpose of the class	0.5		
Code is fully commented including all variables and methods	0.5		
No repeated code	1		
Student class			
Instance variables are correct and private	0.5		
Default and parameterised constructors are correct	0.5		
Method for calculating grade is correct	1		
Get and set methods are correct	1		
GradingSystemMenu class – Enter student's record			
Student name and student number is read correctly	0.5		
Assessment marks are read correctly	0.5		
Data is added to the object array correctly	2		
Output resembles the specification (two decimal points)	1		
Grades calculated correctly	0.5		
Error when maximum records is reached	0.5		
Error when student name and student number not entered	0.5		
Error when assessment marks are out of range	0.75		
Data validation loops are correct	0.25		
Display all student's record			
All records displayed	2		
Output resembles the specification	1		
Display statistics			
Maximum and minimum are correct	0.5		
Average is correct	0.5		
Total marks and grade calculation are correct	0.5		
Output resembles the specification (two decimal points)	0.5		
General			
Welcome and exit message (as per assignment one)	0.5		
No records entered is handled	0.5		

correctly			
Netbeans project folder submitted as a zip	0.5		
Report			
Explain the Menu (getMenuItem) and how it interfaces with main method (processingGradeingSystem)	3		
Explain interface between Student and GradingSystemMenu class	3		
UML class diagram of Student class is correct	1		
Screen shot(s) of testing and annotations	1		
Report presentation and comments including how long it took and any problems encountered	0.5		
Total	**30**		