4COSC001W: Software Development I – Coursework specification (2023/24)	
Module leader	S Roberts
Weighting:	50%
Qualifying mark:	30%
Description:	Coursework
Learning Outcomes Covered in this Assignment:	The coursework rationale is: LO1 Analyse specific problems and design their solutions by applying appropriate algorithmic techniques; LO2 Apply programming concepts to implement solutions in the taught programming language; LO3 Implement and manipulate simple data structures; LO4 Use an integrated development environment to create programs to satisfy a simple specification.
Handed Out:	Friday 27 October 2023
Due Date:	**Coursework Deadline**: Thursday 23 November **BEFORE** 1:00 pm
Expected deliverables:	a) Submit your Python program code - **Important:** Submit your python code file created in IDLE using the name convention: "student_id.py", e.g. w1234567.py - **DO NOT** submit your code as a word, notepad or PDF document. b) Submit your test case results on the word version provided c) Demo (during a scheduled computer seminar (week 10 **or** week 11))
Method of Submission:	Submitted online via Blackboard
Type of Feedback and Due Date:	Written feedback and marks 15 working days (3 weeks) after the submission deadline. All marks will remain provisional until formally agreed by an Assessment Board.

Assessment regulations
Refer to the following for clarification on what constitutes plagiarism, collusion and penalties for late submissions.

This is an individual coursework. You should not share your coursework or parts of your coursework with another student as this can cause you both to receive an allegation of collusion:

https://www.westminster.ac.uk/current-students/guides-and-policies/academic-matters/academic misconduct/collusion
Clarification on what constitutes plagiarism:

https://www.westminster.ac.uk/current-students/guides-and-policies/academic-matters/academic misconduct/plagiarism

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website:
http://www.westminster.ac.uk/study/current-students/resources/academic-regulation

Coursework Description

General Notes

• Use user-defined functions in your solution as appropriate.

• Use descriptive names for your variables and user-defined functions.

• Reference within your code any code adapted from external or other sources.

A University requires a program to predict progression outcomes at the end of each academic year. You should write this program in Python using the data shown in Table 1.

	Volume of Credit at Each Level			Progression Outcome
	Pass	Defer	Fail	
1	120	0	0	Progress
2	100	20	0	Progress (module trailer)
3	100	0	20	Progress (module trailer)
4	80	40	0	Do not Progress – module retriever
5	80	20	20	Do not Progress – module retriever
6	80	0	40	Do not Progress – module retriever
7	60	60	0	Do not progress – module retriever
8	60	40	20	Do not progress – module retriever
9	60	20	40	Do not progress – module retriever
10	60	0	60	Do not progress – module retriever
11	40	80	0	Do not progress – module retriever
12	40	60	20	Do not progress – module retriever
13	40	40	40	Do not progress – module retriever
14	40	20	60	Do not progress – module retriever
15	40	0	80	Exclude
16	20	100	0	Do not progress – module retriever
17	20	80	20	Do not progress – module retriever
18	20	60	40	Do not progress – module retriever
19	20	40	60	Do not progress – module retriever
20	20	20	80	Exclude
21	20	0	100	Exclude
22	0	120	0	Do not progress – module retriever
23	0	100	20	Do not progress – module retriever
24	0	80	40	Do not progress – module retriever
25	0	60	60	Do not progress – module retriever
26	0	40	80	Exclude
27	0	20	100	Exclude
28	0	0	120	Exclude

Table 1: Progression outcomes as defined by the University regulations.

Part 1 - Main Version

A. Outcomes (28 marks)

• The program should allow students to predict their progression outcome at the end of each academic year. The program should prompt for the number of credits at pass, defer and fail and then display the appropriate progression outcome for an individual student (i.e., progress, trailing, module retriever or exclude).

B. Validation (12 marks)

- The program should display '**Integer required**' if a credit input is the wrong data type.
- The program should display '**Out of range**' if credits entered are not in the range 0, 20, 40, 60, 80, 100 and 120.
- The program should display '**Total incorrect**' if the total of the pass, defer and fail credits is not 120.
- A few marks will be allocated for the efficient use of conditional statements. For example, the program does not need 28 conditional statements for 28 outcomes.
- An example of the program running with user input (shown in bold):

```
Please enter your credits at pass: p
Integer required

Please enter your credits at pass: 140
Out of range.

Please enter your credits at pass: 100
Please enter your credit at defer: 40
Please enter your credit at fail: 20
Total incorrect.

Please enter your credits at pass: 100
Please enter your credit at defer: 20
Please enter your credit at fail: 0
Progress (module trailer)
```

C. Multiple Outcomes (12 marks)

- The program loops to allow a staff member to predict progression outcomes for multiple students.
- The program should prompt for credits at pass, defer and fail and display the appropriate progression for each individual student until the staff member enters **'q' to quit**. Optionally you can use an input of **'y' to continue.**
- See example of program run combined with Histogram below.

D. Histogram (9 marks)

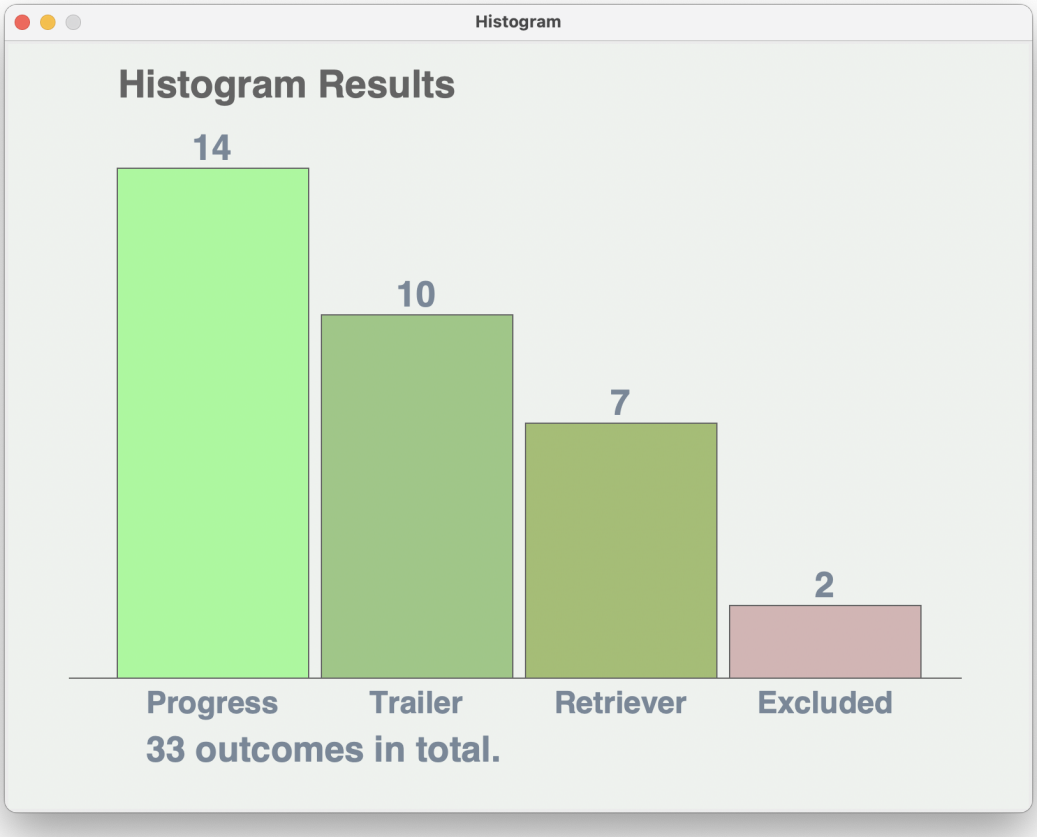
- When 'q' is entered, the program should use the **"graphics.py"** module to produce a 'histogram' representing the number of students who achieved a progress outcome in each category range: progress, trailing, module retriever and exclude. **The histogram should relate to the data input entered during the program run and work for any number of outcomes, it must use the graphics.py module**.
- Display the number of students for each progression category and the total number of students.
- Example of a program run and input (in bold). Note: program should exit on 'q' to quit and produce the histogram. 'y' to continue shown in the example is optional and depends on your program structure.

Example Output:

```
Enter your total PASS credits: 120
Enter your total DEFER credits: 0
Enter your total FAIL credits: 0
Progress

Would you like to enter another set of data?
Enter 'y' for yes or 'q' to quit and view results: y

Enter your total PASS credits: 100
Enter your total DEFER credits: 0
Enter your total FAIL credits: 20
Progress (module trailer)
```

```
Would you like to enter another set of data?
Enter 'y' for yes or 'q' to quit and view results: y

Enter your total PASS credits: 80
Enter your total DEFER credits: 20
Enter your total FAIL credits: 20
Module retriever

Would you like to enter another set of data?
Enter 'y' for yes or 'q' to quit and view results: y

Enter your total PASS credits: 60
Enter your total DEFER credits: 0
Enter your total FAIL credits: 60
Module retriever

Would you like to enter another set of data?
Enter 'y' for yes or 'q' to quit and view results: y

Enter your total PASS credits: 40
Enter your total DEFER credits: 0
Enter your total FAIL credits: 80
Exclude

Would you like to enter another set of data?
Enter 'y' for yes or 'q' to quit and view results: q
```

The program should now display a histogram of results using the graphics.py module. Example Window Below



Submit the completed part 1 test plan provided with your final part 1 solution.

Part 2 – List (extension) (6 marks)

For Part 1, most of the solutions would use variables to store the input data. For Part 2, extend your solution so that the program saves the input progression data to a list or nested list. Then access the stored data from the list and print the data in the following format below. Test plan not required.
Example Output: The following should display after the histogram

```
Part 2:
Progress - 120, 0, 0
Progress (module trailer) - 100, 0, 20
Module retriever - 80, 20, 20
Module retriever - 60, 0, 60
Exclude – 40, 0, 80
```

Part 3 - Text File (extension) (6 marks)

For this part you could create an additional Part 3 program or extending your original version. Use python to save any inputted progression data to a text file. Later in the program, access the stored data and print out as shown below. Test plan not required. Example output (with data from text file):

```
Part 3:
Progress - 120, 0, 0
Progress (module trailer) - 100, 0, 20
Module retriever - 80, 20, 20
Module retriever - 60, 0, 60
Exclude – 40, 0, 80
```

Use of User-Defined Functions (5 marks)

Use user-defined functions in your solution as appropriate.

Use of Descriptive Variable/Function names (3 marks)

Use descriptive names for your variables and user-defined functions.

Coursework Demo (14 marks)

- You are expected to demonstrate your working solution to your tutor during a scheduled online tutorial as shown on the weekly schedule. **NOTE: If you do not attend your demo only your solutions for Part 1 will be marked.**
- Demo marks are allocated for your ability to answer questions and demonstrate understanding of your solutions.
- If you cannot explain your code and are unable to point to a reference within your code of where this code was found (i.e., in a textbook or on the internet) then no marks will be given for the demo of that component.

References

- Reference any code taken from other sources in your program code with python program comments.
- Include the following at the top of your program(s).

```
# I declare that my work contains no examples of misconduct, such as plagiarism, or
collusion.
# Any code taken from other sources is referenced within my code solution.

# Student ID: ……..…

# Date: ……..…
```

Marks per Component

Component		Marks
Part 1	• A) Outcomes (conditions) ○ Predicts **progress** correctly (6) ○ Predicts **module trailer** correctly (5-6) ○ Predicts **module retriever** correctly (5-6) ○ Predicts **exclude** correctly (5-6) • Test Plan	**24** **4**
	• B) Validation ○ Identifies input that is wrong data type (3) ○ Credits outside range, 20, 40, 60, 80, 100, 120 (3) ○ Identifies credit total not 120 (3) • Test Plan	**9** **3**
	• C) Multiple Outcomes (loop) ○ Loops to predicts progression outcomes for multiple inputs (8) ○ User enters 'q' to quit (2) • Test Plan	**10** **2**
	• D) Histogram ○ 'Histogram' displays (3) ○ 'Histogram' outcomes correct (3) ○ Category and overall totals correct (3) • Test Plan	**12** **2**
	Part 1 - Subtotal	**66**
Part 2: List	• Input data stored in list (3) • Output retrieved from list (3)	**6**
Part 3: Text File	• Input data saved to text file (3) • Output retrieved from text file (3)	**6**
User-Defined Functions		**5**
Descriptive Variable/Function names		**3**
Coursework Demo (1-to-1 with module tutor during your usual seminar (Week 10 or 11)) • Did not attend demo (0) – Note: Only Part 1 will be marked • Explanations for Part 1 (0-5) • Explanations Part 2(0-2) • Explanations for Part 3 (0-2), • Explanations for Part 4 (0-2) • Excellent explanations & all parts completed (12+)		**14**
	Total	**100**

Pass requirements

Pass (40 marks or above) To pass this assessment you should aim to complete Part 1 - A (Outcomes), **either** Part 1 - B (Validation) **or** Part 1 - C (Loop) and complete the test plan.

First-Class (70 marks or above**)** To achieve a first-class mark for this assessment you should aim to complete Part 1 and at **least two** from Part 2 (List), Part 3 (Text File), Part 4 (Dictionary) or User Defined Functions. You should attend your coursework demonstration