4COSC006C: Software Development I – Coursework specification (2021/22)

Module leader	Saman Hettiarachchi
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:	Thursday 24th February 2022
Due Date:	Monday 28th March 2022 at 1:00 pm

Expected deliverables:	a) Submit your Python program code and test results document as a **single compressed file**
	    - **Important:** Submit your compressed file created using the name convention: "student_id.zip", e.g. w1234567.zip
	    - **DO NOT** submit your code as word, notepad or a PDF document (only as .py file(s)).
	    - **You have the option to create 4 different python scripts or one single python scripts with a menu with 4 options**
	b) Submit your test case results on the word version provided
	c) Demo and viva (during a scheduled computer seminar – week 10 **or** week 12 **NOTE: It is mandatory to attend viva. Not attending viva will cause your mark to be capped at 30.**

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 section 4 of the "How you study" guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

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-regulations**

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.

The 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.

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

	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

Part 1 - Main Version
Outcomes

1. 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).

2. Validation
   - 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)
   ```

3. Multiple Outcomes & Histogram
   - 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 user enters 'q' to quit. Optionally you can use an input of **'y' to continue.**
   - When 'q' is entered, the program should produce a 'histogram' where each star represents a student who achieved a progress outcome in the category range: progress, trailing, module retriever and exclude. **The histogram should relate to the data input entered by the staff member during the program run and work for any number of outcomes**.
   - 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. 'y' to continue shown in the example is optional and depends on your program structure.

   ```
   Staff Version with Histogram

   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


----------------------------------------------------------------
Horizontal Histogram
Progress 1  : *
Trailer 1   : *
Retriever 2 : **
Excluded 1  : *

5 outcomes in total.
----------------------------------------------------------------
```

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

Part 2 - Vertical Histogram (extension)

Extend your program to add a vertical histogram (stars in a category should go downwards), e.g.;

```
    Progress  Trailing  Retriever  Excluded
       *         *          *          *
                            *
```

• If attempted, the code for **both** staff versions (Part 1 and Part 2) must be in your program and submitted for marking.
• Submit the completed test plan provided with your final part 2 solution.

Part 3 - List/Tuple/Directory (extension)

Extend your solution so that the program uses Python to save the input progression data to a list, tuple or directory. Then access the stored data from the list, tuple, directory and print the data in the following format below.

Output: The following should display after the histogram(s)

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

Submit the completed test plan provided with your final part 3 solution.

Part 4 - Text File

For this part you could create an additional Part 4 program or extending your original version. Use python to save input progression data to a text file. Later in the program, access the stored data and print out as shown below. Example output (with data from text file):

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

Submit the completed test plan provided with your final part 4 solution.

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: ……………………….…

Coursework Demo

- 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 you 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.

APPENDIX 1 - Marking scheme

Test Plan	Criteria	Marks	Subtotal
	Part 1 – Main Version		
Tests 1 -10	Outcomes: • Predicts **progress** outcome correctly (4) with one condition statement (1) • Predicts **module trailer** outcome correctly (4) with one condition statement (1) • Predicts **module retriever** outcome correctly (4) with one condition statement (1) • Predicts **exclude** outcome correctly (4) with one condition statement (1) Submitted Test Plan PASS/FAIL correct at demo. Zero if test plan results incorrect or FAIL shown for all tests 1-10.	5 5 5 5 5	/25
Tests 11-13	Validation: • Identifies input that is wrong data type • Credits outside range, 20, 40, 60, 80, 100, 120 • Identifies credit total not 120 Submitted Test Plan PASS/FAIL correct at demo. Zero if test plan results incorrect or FAIL shown for all test 11-13.	3 3 3 3	/12 (37)
Tests 14-15	Loops: • Predicts progression outcomes for multiple students • User enters 'q' to quit Submitted Test Plan PASS/FAIL correct at demo. Zero if test plan results incorrect or FAIL shown for tests 14-15.	8 2 2	/12 (49)
Tests 16	Histogram: • 'Histogram' outcomes correct • Category and overall totals correct Submitted Test Plan PASS/FAIL correct at demo. Zero if test plan results incorrect or FAIL shown for test 16.	4 2 1	/7 (56)
Test 17	Part 2 - Vertical histogram: Submitted Test Plan PASS & correct at demo	5 1	/6 (62)
Test 18	Part 3: • Data stored in list, tuple or dictionary • Data from list, tuple or dictionary is displayed Submitted Test Plan PASS & correct at demo	4 4 2	**/10** (7**2**)
Test 19	Part 4: Text File • Input data saved to text file • Stored data is accessed and displayed correctly Submitted Test Plan PASS & correct at demo	4 4 2	**/10** (8**2**)
	User-Defined Functions	5	(87)
	Programming style - descriptive variable/function names	3	(**90**)
	Demo in scheduled computer seminar (Week 10 or 12)	10	
	Totals		**/100**

APPENDIX 2 - TEST PLANS (see also separate word version)

Student Name:			Student ID:	
TEST PLAN for Part 1				
Submit completed test plan with your code solution				
Test No.	Test Input	Expected Result	Actual Result (or state 'not attempted')	Pass / Fail ('Actual Result' matches 'Expected Result = Pass')
1	Pass = 120 Defer = 0 Fail = 0	'Progress' is displayed		
2	Pass = 100 Defer = 20 Fail = 0	'Progress (module trailer)' displayed		
3	Pass = 100 Defer = 0 Fail = 20	'Progress (module trailer)' displayed		
4	Pass = 80 Defer = 20 Fail = 20	'Module retriever' is displayed		
5	Pass = 60 Defer = 40 Fail = 20	'Module retriever' is displayed		
6	Pass = 40 Defer = 40 Fail = 40	'Module retriever' is displayed		
7	Pass = 20 Defer = 40 Fail = 60	'Module retriever' is displayed		
8	Pass = 20 Defer = 20 Fail = 80	'Exclude' displayed		
9	Pass = 20 Defer = 0 Fail = 100	'Exclude' displayed		
10	Pass = 0 Defer = 0 Fail = 120	'Exclude' displayed		
Validation				
11	Pass = a	'Integer required' displayed		
12	Pass = 5	'Out of range' displayed		
13	Pass = 100 Defer = 40 Fail = 0	'Total incorrect' displayed		
Displaying Histogram				
14	Program loops	Predicts multiple progression outcomes		

| 15 | Enter 'q' to quit | Exits loop | | |
| 16 | Exit loop | Progress 1 : *
Trailer 2   : **
Retriever 4 : ****

Excluded 1 : *

8 outcomes in total. | | |

TEST PLAN for **Part 2, 3 and 4**			Actual Result (or state 'not attempted')	Pass/Fail
17	Vertical Histogram	Progress 1 | Trailer 2 | Retriever 4 | Exclude 1 * * * * * * * * 8 outcomes in total.		
18	Data stored & retrieved from list, tuple or dictionary	Output uses data from list, tuple or dictionary		
19	Data stored & retrieved from text file	Output uses data from text file		

TEST PLAN for **Part 2, 3 and 4**			Actual Result (or state 'not attempted')	Pass/Fail
17	Vertical Histogram	Progress 1 | Trailer 2 | Retriever 4 | Exclude 1 * * * * * * * * 8 outcomes in total.		
18	Solution uses list, tuple or dictionary	Output uses data stored to and retrieved from list, tuple or dictionary		
19	Solution uses text file	Output uses data stored to and retrieved from text file		