Assessment information for candidates

This assessment applies to the following Unit Outcomes and Assessment Standards:

Software Design and Development (Higher)

Outcome 1
The candidate will:

1 Explain how programs work, drawing on an understanding of advanced concepts in software development and computer architecture by:

1.1 Reading and explaining code
1.2 Describing the purpose of a range of programming constructs and how they work
1.3 Describing how a range of standard algorithms work
1.4 Describing how programs relate to low-level structures and operations

The range of programming constructs should include subprograms, parameters user-defined functions and sequential file operations. The range of standard algorithms should include input validation, linear search, finding minimum and maximum and counting occurrences.

Outcome 2
The candidate will:

2 Develop modular programs using one or more software development environments by:

2.1 Applying contemporary design and development methodologies
2.2 Selecting and using combinations of appropriate constructs
2.3 Selecting and using appropriate simple and structured data types, including 1-D arrays
2.4 Testing digital solutions systematically
2.5 Applying aspects of good programming technique — meaningful variable names, internal commentary, indentation

The range of programming constructs should include subprograms, parameters and sequential file operations. Programs should include at least two data types, including 1-D arrays.

To pass this assessment, you will have to show that you have met these Outcomes and Assessment Standards.

Your assessor will let you know how the assessment will be carried out and any required conditions for doing it.

Outcomes 1 and 2

Programming task

This assessment task combines designing and developing a program to meet the problem specification, and then using the internal commentary of your program, and a few additional questions, to meet Outcome 1.

To pass this assessment, you will have to complete all the steps, show your assessor your working program, and hand in all the hard copy evidence of the work you have completed.

You can document your achievement in the programming tasks checklist.

Evidence of sufficient programming tasks should be appropriately labelled with your name, date(s) of completion, and your assessor's comments (if any). Evidence may be kept in electronic or hard copy form.

Evidence for the successful completion of these Outcomes may be drawn from more than one programming task.

Problem specification

Your overall percentage grade in your Higher Computing course is devised by adding the coursework mark (out of 60) to the exam mark (out of 90) and then calculating the percentage.

Usually the grades are awarded by the following percentages:

A Grade is awarded for greater than or equal to 70%
B Grade is awarded to those between 60% and 69%
C Grade is awarded to those between 50% and 59%
D Grade is awarded to those between 45% and 49%
No Grade is given to those achieving less than 45%

Your task is to design, implement and test a program by completing the following steps:

Assessment Standards	Steps to achieve task
2.1	**Step 1** ♦ Write an algorithm using an appropriate design notation which will input one student's coursework mark and prelim mark and calculate and display their percentage and grade. ♦ Remember to show the data flow and ensure that the input is validated.
2.4	**Step 2** ♦ Create your own test plan for the program which will systematically test that the program works.
2.2 **2.3** **2.5**	**Step 3** ♦ Write a program based on your design. ♦ You should be using subprograms and parameter passing within the program. ♦ Remember to use meaningful variable names, internal commentary and indentation.
1.2	**Step 4** ♦ Use internal commentary to explain the purpose of using subprograms and parameter passing and how they work.
2.4	**Step 5** ♦ Test the program using your test plan. ♦ If necessary, make any amendments to your program.
1.1 **1.3**	**Step 6** ♦ Ask your assessor for four standard algorithm subprograms and insert them into your program. ♦ Use internal commentary to describe how each of the subprograms work.

1.2 **2.2** **2.3**	**Step 7** ♦ Your program only works for one candidate at present. ♦ Alter your program so that it will read the name, coursework mark and prelim mark for all the 15 students in your class from an external file (available from your assessor). ♦ Make sure that you add internal commentary to explain how this works.
	Step 8 ♦ Alter your program to enable it to find out how many 'A' passes are in the class.
	Step 9 ♦ Demonstrate your working program to your assessor. ♦ Hand in hard copy evidence of the work you have done, including your: — program design — program listing showing meaningful variable names, internal commentary, and indentation — completed test plan with evidence of final testing, data used and results
1.4	**Step 10** Answer the following questions: **a)** Once the program has been tested it is running in the Python environment. Describe the role of the interpreter in the development of a Python program. **b)** Describe the role of the processor in executing a Python instruction. **c)** Explain what will be in the computer's memory immediately after the Python code has been translated.