### Higher Software Design and Development Unit (J27C 76)

### 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 an additional question, to meet Outcome 1 and 2.

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 SCQF level 6 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:**

|  |  |
| --- | --- |
| **Step 1** | **Assessment Standard 2.1** |

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.

1. **Get details from user**
2. **Calculate percentage**
3. **Calculate grade**
4. **Display results**
   1. **Get PupilName from KEYBOARD**
   2. **Start conditional loop**
   3. **Get CourseWorkMark from KEYBOARD**
   4. **End loop when condition is met**
   5. **Start conditionsl loop**
   6. **Get PrelimMark from KEYBOARD**
   7. **End loop when condition is met**

**2.1 TotalMark = PreilmMark + CourseWorkMark**

**2.2 percentage = TotalMark \* 100 \ 150**

**3.1 If Percentage >= 70 then set Grade to A**

**3.2 Else if Percentage >= 60 and < 69 then set Grade to B**

**3.3 Else if Percentage >= 50 and <59 then set Grade to C**

**3.4 Else if Percentage >= 45 and <49 then set Grade to D**

**3.5 Else if Percentage <45 then set Grade to No Grade**

**3.6 End If**

**4.1 Display PupilName**

**4.2 Display “The percentage is” Percentage**

**4.3 Display “The grade is” Grade**

|  |  |
| --- | --- |
| **Step 2** | **Assessment Standard 2.4** |

Create your own test plan for the program which will systematically test that the program

works.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Types** | **Coursework Mark** | **Prelim Mark** | **Expected Output** |
| **Normal** | **55** | **65** | **Accepted** |
| **Extreme** | **60** | **90** | **Accepted** |
| **Extreme** | **0** | **0** | **Accepted** |
| **Exceptional** | **64** | **98** | **Not Accepted** |

|  |  |
| --- | --- |
| **Step 3** | **Assessment Standards 2.2 and 2.5** |

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.

|  |  |
| --- | --- |
| **Step 4** | **Assessment Standards 1.1 and 1.2** |

Use internal commentary to explain the purpose of using subprograms and parameter passing and how they work.

|  |  |
| --- | --- |
| **Step 5** | **Assessment Standard 2.5** |

Test the program using your test plan. If necessary, make any amendments to your program.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Types** | **Coursework Mark** | **Prelim Mark** | **Output** |
| **Normal** | **55** | **65** | **Accepted** |
| **Extreme** | **60** | **90** | **Accepted** |
| **Extreme** | **0** | **0** | **Accepted** |
| **Exceptional** | **64** | **98** | **Not Accepted** |

|  |  |  |
| --- | --- | --- |
| **Coursework Mark** | **Prelim Mark** | **Output** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| **Step 6** | **Assessment Standards 1.1, 1.2, 2.2, 2.3** |

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 7** | **Assessment Standard 1.3** |

Alter your program to enable it to find out how many ‘A’ passes are in the class by using the

“Count Occurrences” standard algorithm.

Use internal commentary to describe how the “Count Occurrences” algorithm works.

Alter your program to find out who has the best percentage in the class by using the “Find

Max” standard algorithm.

Use internal commentary to describe how the “Finding Max” algorithm works.

|  |  |
| --- | --- |
| **Step 8** | **Assessment Standard 1.4** |

Explain how the computer processes the program using the fetch-execute cycle with reference to processor, memory and buses.

The processor sets up an address bus with the required address, to identify the memory location to be read from.

The processor activates the read line on the control bus, this tells the processor that it is to read an instruction from the identified memory location to the processor.

An instruction is fetched from the identified memory via the data bus and stored in the instruction register, an instruction is transferred to the instruction register in the processor from the identified memory location via the data bus.

The instruction in the instruction register is decoded and executed, the instruction is decoded and executed. This may involve several more fetches of data from memory.ssss

|  |  |
| --- | --- |
| **Step 9** | **Assessment Standard 1.4** |

Demonstrate your working program to your assessor.

Hand in hard copy evidence of the work you have done, includingyour:

* program design
* program listing showing meaningful variable names, internal commentary, and indentation completed test plan with evidence of final testing, data used and results