COMP101: Introduction to Programming 2019-20

Assignment-04

Issue Date: **Tuesday 29th October 2019**

Submission Date: **Wednesday 6th November (17:00)**

Summary:
Assignment-04 is worth 13% of the total marks for COMP101.
The assignment uses sequencing, selection, iteration, function definitions and I/O control of strings and numbers.

You must submit an attempt at this assignment else a fail grade for the module will be awarded.

Submission details: Two files are required:
Submit one .py file AND one .pdf file

1) .py filename format:
familyName_givenName-CA04.py e.g. **Smith_John-CA04.py**

This file contains:
i) Your Python code edited in IDLE (NOT the Shell window)
The first 3 lines should be comment lines as follows:
#Your University id followed by Smith_John-CA04.py
#Month and Year of coding
#Brief description of the problem solved

AND

2) .pdf filename format:
familyName_givenName-CA04.pdf e.g. **Smith_John-CA04.pdf**

This file contains:
i) A test table (use the test table template for evidence of testing)
ii) Pseudocode

Deadline Detail:

By the deadline indicated:

Your documents are to be submitted electronically via the department submission server at https://sam.csc.liv.ac.uk/COMP/Submissions.pl

Earlier submission is possible, but any submission after the deadline attracts the standard lateness penalties - see http://www.csc.liv.ac.uk/department/regulations/practical.html

Plagiarism and collusion guidelines will apply throughout the assignment submission

COMP101 Assignment-04 2019-20

Assessment Information

Assignment Number	04 (of 07)
Weighting	13%
Assignment Circulated	As on front page
Deadline	As on front page
Submission Mode	e-submission
Learning outcome assessed	LO1: Identify principles and practice of using high level programming constructs to solve a problem LO3: Produce documentation in support of a programmed solution LO4: Use a suitable Integrated Development Environment to carry out implementation, interpretation/compilation, testing and execution LO6: Design and apply effective test cases
Purpose of assessment Marking criteria	Assessment of using sequence, selection, iteration constructs and defined functions to control I/O of strings and numbers in successful calculations for a given problem. Modularisation of code. Total marks over seven questions as a percentage
Submission necessary in order to satisfy module requirements?	Yes Assignments are not marked anonymously
Late Submission Penalty	Standard UoL Policy.

Problem Specification:

Start the program with a call to ‘main( )’

1. Present the user with a menu coded as a function ‘main()’

The menu should present three options:

- An option to enter assessment grades for assignments submitted late;
- An option to enter assessment grades for assignments submitted on time;
- An option to exit the program

2. Create functions that can be called from the three options on the menu

3. Call a function from the first option to allow the user to enter grades for all students who have submitted their work late. We do not know how many students have submitted late.
When the user has completed their entries, return them to the Main Menu

4. Call a function from the first option to display a stub indicating this option is under development.
Return the user to the main menu.

5. Entry and exit to/from the program is via the Main Menu

6. Validate all user input:
If invalid data is input, this should be disallowed and the user asked to re-input the value until it is correct

Requirements for assignments submitted late:
Design, implement and test a program that calculates and displays information about penalties for late submission for a piece of coursework as described below:

For work submitted late, calculate the penalties that apply:
The deadline is midnight on the submission day. Any coursework submitted after the midnight deadline is considered late and is subject to a late penalty.

The coursework is marked as it is and given an original mark (the 'raw grade') – an integer in range 0 to 100 inclusive

The pass-mark for the piece of coursework is 40%.

A coursework given a grade of 0 is an academic fail, i.e. it has been graded and has been awarded a score of zero. It does not indicate non-submission.

Calculating late penalties:

The raw grade loses 5 marks every day it is late up to a maximum of 7 days
After 7 days, the work automatically fails

If the raw grade after late deduction drops below 40 no further late deductions are made and the mark awarded will be 40

If the raw grade is less than or equal to 40 then the mark stays at its original value – no late penalties apply.

Write a function to input a mark and number of days late

a) The mark (‘raw mark’) should be an integer between 0 and 100 (inclusive).

b) The number of days should be an integer between 0 and 7 (inclusive). It is possible that coursework submitted late has been excused, hence 0 days late is entered (no penalty).

c) If input is outside these ranges, ask the user asked to re-input the value until the value is within the correct range.

d) Report if the assignment will be capped at 40

e) Output the resulting mark for each day between 0 and the number of days late

Example – sample o/p for raw grade > 40
Raw grade: 55
Number of days late: 4

Days late	Mark to award
0	55
1	50
2	45
3	40
4	35

This assignment will be capped at 40

Enter another mark? Y/N:

Notes:
Good program control should allow one entry and one exit: Entry /exit for the program should only be via the Main Menu

The user should be returned to the Main Menu from any called function.

There is no need to store the individual grades anywhere: Each mark can be displayed on screen (similar to the example) and then the process repeated for the next input

Mark scheme
Analysis and Design 20%

Implementation 65%

Testing 15%

The mark scheme looks for:
A sequence, selection, iteration and modularisation (i.e. defining functions) and the program starting with a call to main().

Efficient use of variables to handle the input data and the use of these variables to make the code clear and readable, thus aiding maintenance and debugging

Appropriate use of output techniques that are of benefit to the user

In-line comments used effectively

Consider design, clarity, accuracy and appropriate use of code, documentation and appropriate testing to determine accuracy and/or problems (which may be documented in the test table comments column)

Because submission is handled electronically any file submitted past the deadline will be considered one day late. Late submissions are subject to the standard University Policy.