Gratia Christian College
宏恩基督教學院

Examination

Programme : Higher Diploma in Transformative Business Management
Academic Year (Semester) : 2025 - 2026 (Semester 1)
Course Code : HTM115
Course Title : Generic Coding Skills for Web Applications
Date : 18 December 2025
Time : 2:30 – 4:30 pm
Time Allowed : 2 Hours
Weighting : 50% of the Overall Marks

Student ID No. : _____

This paper has **8** pages (including this cover page).

Instructions to candidates:

1. This question paper consists of two sections, answer ALL questions in both Sections.
2. Section A (40 marks) is multiple choices questions, choose ONE of the four options, otherwise, it will be deemed to be wrong.
3. Section B (60 marks) is short questions.
4. Answer all questions in the answer book provided.
5. This paper carries 100 marks which are allocated to each question as indicated.
6. This is a closed-book examination.

No materials or aids are allowed during the whole test/examination. If any unauthorized materials or aids are found on a candidate during the test/examination, the candidate will be subject to disciplinary action.

Do not turn this page until you are told to do so.

版權所有©
All Rights Reserved.

版權所有©All Rights Reserved.

版權所有©All Rights Reserved.

版權所有©
All Rights Reserved.

版權所有©All Rights Reserved.

版權所有©
All Rights Reserved.

版權所有©All Rights Reserved.

版權所有©
All Rights Reserved.

版權所有©All Rights Reserved.

版權所有©
All Rights Reserved.

版權所有©All Rights Reserved.

版權所有©
All Rights Reserved.

Section B (60 marks)

Question 1: The Nested Decision (Grading System) (30 marks)

Part A: Flowchart Design (15 marks)

Design a flowchart that checks if a student passed a class based on two criteria: attendance and exam score.

1. **Start** the program.
2. **Input** the student's Attendance Percentage (number) and Exam Score (number).
3. **Decision (First Check):** Check if the Attendance Percentage is **greater than or equal to 80**.
   - **NO Path (False):**
     - **Output:** Display **"Failed (Poor Attendance)."**
     - **End** the program.
   - **YES Path (True):**
     - **Decision (Nested Check):** Check if the Exam Score is **greater than or equal to 60**.
       - **YES Path (True):**
         - **Output:** Display **"Passed."**
       - **NO Path (False):**
         - **Output:** Display **"Failed (Poor Exam Score)."**
     - **End** the program.

Part B: Python Program (15 marks, 3 marks each)

Below is a **Python program** that implements the logic designed in the above flowchart. The program should use **nested if statements** to handle the two-level condition. Ensure user input is converted to a number. The program listed below has 5 items missing in the syntax, please fill in the blanks. Please indicate the item you are answering, that is I, ii, iii, iv and v. You do not need to copy the whole program or the whole line in your answer.

```
1  Items
2          # Prompt the user for input
3          attendance_perc = float(input("Enter Attendance Percentage (0-100): "))
4          exam_score = float(input("Enter Exam Score (0-100): "))
5
6          # Check attendance first
7   I,ii   if attendance_perc    ___ 80_
8              # Nested check for exam score
9   iii        if _________ >= 60:
10                 print("Passed.")
11  iv         ________
12                 print("Failed (Poor Exam Score).")
13  v      _______
14             print("Failed (Poor Attendance).")
15
```


版權所有©
All Rights Reserved.

Question 2: Number Accumulator (Loop) (30 marks)

Part A: Flowchart Design (15 marks)

Design a flowchart that continuously asks the user for numbers and adds them to a running total until the user enters the number **0**.

1. **Start** the program.
2. **Process:** Initialize Total to 0.
3. **Input:** Get the first New Number from the user.
4. **Loop Condition (Decision):** Check if New Number is **not equal to 0**.
   - **YES Path (True):**
     - **Process:** Update Total = Total + New Number.
     - **Input:** Get the **next** New Number from the user.
     - (Flow returns to the Loop Condition/Decision step).
   - **NO Path (False):**
     - **Output:** Display the final calculated Total.
     - **End** the program.

Part B: Python Program (15 marks, 3 marks each)

The following is a **Python program** that implements the logic designed in the flowchart above. Used a **loop** with a **priming read** (getting the first input before the loop starts) and a **loop read** (getting the next input inside the loop). There are 5 items missing in the program as indicated as I, ii, iii, iv and v. Please fill in the blanks and write your answers in the answer book. You do not need to copy the whole program or the whole line in your answer.

	Items	
1		
2		`# Initialise the running total`
3	i	`total = ____`
4		
5		`# Get the first number from the user`
6	ii	`new_number = _____(input("Enter a number (0 to quit): "))`
7		
8		`# Continue adding numbers while the input is not zero`
9	iii, iv	`_____ new_number __ 0:`
10	v	`    total = _____ + new_number`
11		`    new_number = float(input("Enter the next number (0 to quit): "))`
12		
13		`# Output the final total when the loop ends`
14		`print("Total:", total)`
15		

End of Section B and the Paper

版權所有©
All Rights Reserved.