ILERAMARK - SAMPLE MARKING GUIDE

Computer Science Department | 2023/2024 Academic Session

Course: CSC 201 - Introduction to Programming

Total Score: 100 marks Time Allowed: 2 Hours

Instruction: Answer ALL questions. Each question carries the marks indicated.

SECTION A - Theory (50 marks)

Q1. Define an algorithm and list FOUR characteristics of a good algorithm.

[10 marks]

Definition (2 marks):

An algorithm is a finite, step-by-step set of instructions for solving a problem.

Four characteristics - 2 marks each (any four from):

- Input: Has zero or more inputs
- Output: Produces at least one output
- Definiteness: Each step is clear and unambiguous
- Finiteness: Terminates after a finite number of steps
- Effectiveness: Each step is executable in finite time

Q2. Differentiate between compiled and interpreted programming languages. Give ONE example of each.

[10 marks]

Compiled language (4 marks):

Source code is translated entirely to machine code before execution by a compiler.

Example: C, C++, Fortran (accept any valid compiled language)

Interpreted language (4 marks):

Source code is executed line-by-line at runtime by an interpreter.

Example: Python, JavaScript, BASIC (accept any valid interpreted language)

Correct examples: 1 mark each

Q3. Write a pseudocode to read 10 numbers and display their average.

[15 marks]

Award marks as follows:

- Declare variables (sum, count, number, average): 2 marks
- Initialize sum = 0, count = 0: 2 marks
- Loop structure (FOR or WHILE) iterating 10 times: 4 marks
- Read each number and add to sum inside loop: 3 marks
- Compute average = sum / 10 after loop: 2 marks
- Display average: 2 marks

Deduct 2 marks if loop boundary is incorrect (e.g., 9 or 11 iterations).

Accept equivalent correct pseudocode styles.

Q4. Explain THREE types of programming errors with examples.

[15 marks]

Syntax Error (5 marks): Violation of language grammar rules.

Example: missing semicolon in C, misspelled keyword.

Runtime Error (5 marks): Error that occurs during execution.
Example: division by zero, null pointer dereference.

Logic Error (5 marks): Program runs but produces incorrect output.
Example: using + instead of * in a multiplication formula.

Award full marks only if both the type name and a valid example are given.

SECTION B - Practical / Problem Solving (50 marks)

Q5. Write a Python program that takes a student's score (0-100) as input and prints the grade: A (70-100), B (60-69), C (50-59), D (40-49), F (below 40).

[20 marks]

Award marks as follows:
- Correct input statement (input()): 2 marks
- Convert input to integer/float: 2 marks
- Correct if-elif-else chain covering all five grade bands: 10 marks (2 each)
- Correct boundary conditions (e.g., >= 70 not > 70 for A): 4 marks
- Print correct grade label: 2 marks

Sample accepted solution:

```
score = int(input('Enter score: '))
if score >= 70:   print('A')
elif score >= 60: print('B')
elif score >= 50: print('C')
elif score >= 40: print('D')
else:             print('F')
```

Award partial marks for correct logic with minor syntax errors.

Q6. Trace the following code and state the final output:

```
total = 0
for i in range(1, 6):
    if i % 2 == 0:
        total += i
print(total)
```

[10 marks]

Correct trace:
- i=1: 1 % 2 = 1 (odd) - skip
- i=2: 2 % 2 = 0 - total = 2
- i=3: 3 % 2 = 1 (odd) - skip
- i=4: 4 % 2 = 0 - total = 6
- i=5: 5 % 2 = 1 (odd) - skip

Final output: 6

Marking: 2 marks per correct iteration trace (up to 8), 2 marks for correct final output.

Award 5 marks for correct final answer even without full trace.

Q7. Define a Python function factorial(n) that returns the factorial of n using recursion.

[20 marks]

Award marks as follows:

- Correct function definition with parameter n: 2 marks
- Correct base case (if n == 0 or n == 1: return 1): 6 marks
- Correct recursive case (return n * factorial(n - 1)): 8 marks
- Function returns correct value (not just prints): 2 marks
- At least one correct test call shown or implied: 2 marks

Sample accepted solution:

```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    return n * factorial(n - 1)
```

Deduct 4 marks for infinite recursion (missing base case).

Award up to 14/20 for a correct iterative solution (not recursive).
