COLLEGE OF BUSINESS AND ECONOMICS

DEPARTMENT OF APPLIED INFORMATION SYSTEMS

JUNE 2019 SUPPLEMENTARY EXAMINATION

MODULE:	Development Software 1A
CODE:	DEV1A01
DATE:	June 2019
DURATION:	3 Hours
TIME:	TBA
TOTAL MARKS:	140 (Weight: 50% of Final Mark)

EXAMINER(S):	Dr Abejide Ade-Ibijola (Jide / x1213)
MODERATOR(S):	Dr Gideon Nimako (WITS)
NUMBER OF PAGES:	7 (Excluding the cover page)

INSTRUCTIONS:

- This is a CLOSED book examination.
- There are 140 marks available. 140 marks = 50%.
- Electronic devices, and digital watches are NOT allowed.
- You are given one question paper and one answer booklet. You must write your student number on the answer booklet and hand it in.
- Write neatly and legibly.
- For all multiple-choice questions, indicate your answer clearly. Your answer must be a unique option. If two or more answers are indicated where only one is required, you will be marked wrong for that question. Note that some multiple-choice questions allow two or more options to be selected.
- The general University of Johannesburg policies, procedures and rules pertaining to written assessments apply to this examination.

Section A: Starter (50 marks)

1. Sibongile is writing an algorithm. She has a fragment of code that he wants to repeat if a condition is true. However, she also wants the fragment to run at least once, before the condition is tested. What control structure do you recommend for Sibongile?

1 MARK(S)

2. Mathematical style algorithm is sometimes referred to as:

1 MARK(S)

3. A well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output, is called?

2 MARK(S)

4. The stage of problem solving that deals with the question: *how will the program interact with the user?* is:

2 MARK(S)

5. Knowing how to represent the information (or data) describing a computational problem is the study of:

2 MARK(S)

6. Determining the steps to transform the information from one representation into another is the study of:

2 MARK(S)

7. In flowcharting, what is the symbol for internal storage?

2 MARK(S)

8. The sequence: 1, 1, 2, 6, 24, 120, 720, 5040, . . . is called?

2 MARK(S)

9. Examine the following expression: $\dfrac{xy}{z} + \dfrac{ab}{c} = \dfrac{ab}{c} + \dfrac{xy}{z}$. The mathematical law(s) that was applied here is:

    (a) Commutative Law
    (b) Associative Law
    (c) Distributive Law
    (d) Precedence of operator Law
    (e) Addition of Fractions Law
    (f) Disjunction Law
    (g) Conjunction Law

2 MARK(S)

10. In mathematically styled algorithms, Σ refers to:

    (a) Sum of a sequence.
    (b) Product of a sequence.
    (c) Factorial sequence.
    (d) Fibonacci sequence.
    (e) Sum of fractions.

2 MARK(S)

11. A `for` loop iterator needs to be incremented in a loop by the programmer to enable it go to the next value.

    (a) This statement is true.
    (b) This statement is false.
    (c) Depends on what program you are writing.
    (d) Only true in Python.
    (e) True for object-oriented languages.

2 MARK(S)

12. In problem solving, what is the relationship between Data Structures and Algorithm?

3 MARK(S)

13. State the syntax of an IF-statement in algorithms.

3 MARK(S)

14. Give two examples, one for each of: Conjunction and Disjunction.

4 MARK(S)

15. Give two differences between a tuple and a list.

4 MARK(S)

16. Complete the following truth table:

P	Q	$P \wedge ((Q \vee P) \wedge P)$

4 MARK(S)

17. With the aid of a truth table, proof that the De Morgan's laws are true.

12 MARK(S)

Section B: Algorithms and Flowcharts (20 marks)

Determine the output of the following algorithms where possible. If buggy, specify the bug (without re-writing the algorithms):

1. Algorithm 1

```
x = TRUE
FOR y = 3 TO -8 STEP -4
    DISPLAY y*y
    WHILE ((x == TRUE) AND (ABS(y) MOD 2 == 0))
        DISPLAY y-2
    END WHILE
    x = NOT(x)
END FOR
```

5 MARK(S)

2. Algorithm 2

```
FOR i = 1 TO 5 STEP 3
        FOR j = i TO (i-5) STEP -3
                DISPLAY (i + j)
        END FOR
END FOR
```

5 MARK(S)

3. **Largest of Three**: Write an algorithm to determine the largest of three numbers, entered by a user.

5 MARK(S)

4. **Grading**: Write an algorithmic fragment that decides the grade of a student based on their mark. Assume the following:

   75 to 100: A
   60 to 74: B
   50 to 59: C
   0 to 49: F

5 MARK(S)

Section C: Program Fragments and Debugging (20 marks)

Determine the output of the following python program fragments where possible. If buggy, specify the bug (without re-writing the programs). In **ALL** cases, assume the math library has been imported.

1. Fragment 1

```
k = -5
f5 = 20
w1 = ((f5 - math.sqrt(9))) - ((44%2) + k)
if not (k <= -6) and not (f5 > 22):
        print(w1)
else:
        print("Do nothing!")
```

5 MARK(S)

2. Fragment 2

```
i = 8
d6 = 15
e7 = (i - d6)
if (i == 20) or not (d6 > 11):
        print(e7)
else:
        print("Do nothing!")
```

5 MARK(S)

3. Fragment 3

```
c = 12
t2 = 9
w4 = 13
c0 = 8
r = ((c0 - w4)) - ((t2 - c))
if not (c == 12) and not (t2 >= -6):
        print(r)
else:
        print(((c - w4)) + ((49%4) + c0))
```

5 MARK(S)

4. Fragment 4

```
x = 16
j1 = -17
b1 = 14
u = (math.ceil(-155.13) + j1) - ((b1 + x))
if not (j1 <= -31):
        print(((u - b1)) + (x + math.ceil(46.28)))
else:
        print((u + x) - ((math.pow(-1,1) + u)))
```

5 MARK(S)

Section D: Python Programs (50 marks)

1. **Sum of Halves**: Write a python program that reads in `n` numbers entered by the user, until the user enters a negative number. Your program should sum up the halves of all the entered numbers.

   Sample Input:
   10
   6
   -1

   Sample Output:
   8

   10 MARK(S)

2. **Vector Addition**: Write a program that reads in any two vectors of any length and adds them, if possible. The resulting vector should be displayed.

   10 MARK(S)

3. **Fractional Sequence**: Write a python program that displays the sequence: $1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \ldots$, until this division converges to zero; test the convergence using 0.00001.

   10 MARK(S)

4. **Combinatorial Analysis**: Write two python function that implements the permutation (${}^{n}P_{r}$) and combination (${}^{n}C_{r}$) functions, using the formulae:

$$ {}^{n}P_{r} = \frac{n!}{(n-r)!} \quad {}^{n}C_{r} = \frac{n!}{(n-r)!r!} $$

   Write a main program that calls the two functions above. Create a menu in your main program that prompts the user to select one of these operations. This menu should run infinitely until the user says they want to quit/exit.

   20 MARK(S)

***** End of paper *****