COLLEGE OF BUSINESS AND ECONOMICS

DEPARTMENT OF APPLIED INFORMATION SYSTEMS

JUNE 2019 SUPPLEMENTARY EXAMINATION

MODULE:	Advanced Programming 1A
CODE:	APM11A1 (Supplementary)
DATE:	June 2019
DURATION:	3 Hours
TIME:	TBA
TOTAL MARKS:	172 (Weight: 50% of Final Mark)

EXAMINER(S):	Dr Abejide Ade-Ibijola (Jide / x1213)
MODERATOR(S):	Dr Gideon Nimako (WITS)
NUMBER OF PAGES:	10

INSTRUCTIONS:

- This is a CLOSED book examination.
- There are 172 marks available. 172 marks = 50%.
- You may write your programs in any of the 4 prescribed programming languages. You may not mix languages within the same program.
- Keep your writing short. Do not write code where algorithm is required; and vice-versa.
- There are five sections in this paper. You are advised to spend an average of 30 minutes on each section. This will leave you with an extra 30 minutes to check your work.
- 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 (47 marks)

1. It is possible for an algorithm to belong to one or more classification or group of algorithms.

   (a) True
   (b) False
   (c) Depends on the programming language
   (d) Depends on the paradigm
   (e) ONLY in object-oriented programming languages

   2 MARK(S)

2. List any four programming paradigms.

   2 MARK(S)

3. In comparing programming languages, list two topics that are worth studying.

   2 MARK(S)

4. What supports are built in programming languages for abstraction?

   2 MARK(S)

5. List four domains where efficient implementation of a programming language may find applications.

   2 MARK(S)

6. A formal language for describing computation?

   2 MARK(S)

7. Programming languages differ mainly by common and uncommon constructs. List two common and two uncommon constructs.

   2 MARK(S)

8. State one attribute of a second generation programming language.

   2 MARK(S)

9. A programming language with features that are built upon a small, mutually independent set of primitive operations is referred to as?

   2 MARK(S)

10. List three trends in research and development for programming languages.

3 MARK(S)

11. In class diagram, list three types of method visibility, and three types of relationships.

3 MARK(S)

12. List any two structural, and two behavioural types of UML diagram.

4 MARK(S)

13. Using terms such as: algorithm, data, functions, facts, rules, objects, and messages; *arithmetically describe* the following styles of programming:
    (a) Imperative
    (b) Functional
    (c) Logic
    (d) Object-orientation

4 MARK(S)

14. Give two examples of a visual programming language.

2 MARK(S)

15. Which of the following variable name(s) is/are invalid?
    (a) `abs`
    (b) `sum of n`
    (c) `aveRAge`
    (d) `while`
    (e) `log`
    (f) `password1`

3 MARK(S)

16. List four reasons for using recursion.

2 MARK(S)

17. Computing the Fibonacci sequence using a recursive function is not efficient. Why is this so?

2 MARK(S)

18. If method A calls method B, method B calls method C, and method C calls method A. How do we refer to methods A, B, C?

2 MARK(S)

19. As a property of a good algorithm design, inputs and outputs must be?

2 MARK(S)

20. List the four programming languages allowed in this module.

2 MARK(S)

Section B: Algorithms and Advanced Data Structures (45 marks)

1. Write an algorithm to sum two numbers together.

3 MARK(S)

2. Write an algorithm that computes the sum of `n` integer numbers.

7 MARK(S)

3. Write an algorithm to read in three fractions of the form:

$$\frac{a}{b}$$

and calculate the sum of these fractions.

5 MARK(S)

4. Write an algorithm that determines if a number entered by a user is prime or not.

10 MARK(S)

5. With the aid of an algorithm, describe the dequeue operation of a Queue object.

5 MARK(S)

Examine the graph $G = \{V, E\}$ in Figure 1, and answer the questions that follo

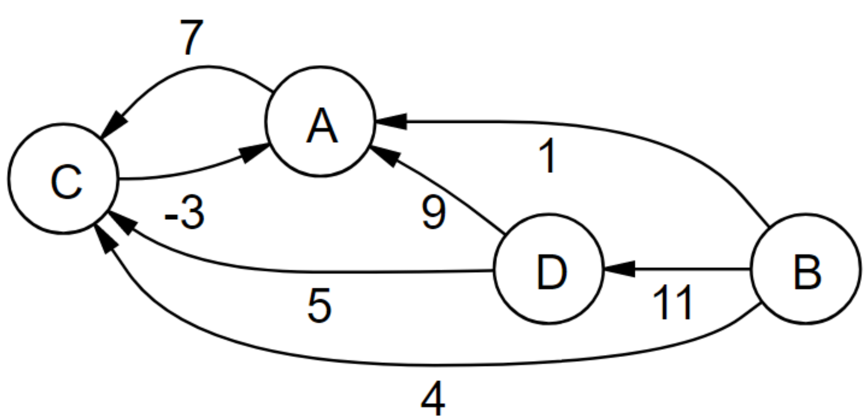

Figure 1: A directed graph, $G = \{V, E\}$

6. Determine $N_G(B)$, $N_G(C)$, $N^-{}_G(A)$, $N^+{}_G(B)$, and $N^-{}_G(C)$.

10 MARK(S)

7. What are the elements of the sets V and E?

3 MARK(S)

8. Is this graph strongly connected?

2 MARK(S)

Section C: Probabilistic Algorithms and Simulations (40 marks)

1. Write a program to generate 600 random PINs of 16 digits each.

10 MARK(S)

2. Write a program to display the configurations in Appendix I, using nested loops.

10 MARK(S)

3. Bongi is observing cars on the highway in Johannesburg. The probability that the colour of a passing car is red is given as 0.88. Write a program that determines the colour of 400 cars on any given day.

10 MARK(S)

4. Write a **recursive function** `GetStringAB(int x)` that returns a string containing $x - 1$ hyphenated A's and a single 'B' at the tail end. For example, if x is 4, the string to be displayed is "$A - A - A - B$". If $x = 7$, "$A - A - A - A - A - A - B$" should be returned.

10 MARK(S)

Section D: Formal Languages, Automata, and Compilers (25 marks)

1. For each of the following regular expressions, defined over the indicated alphabet, determine (from options A to E) which string is accepted by the expression.

10 MARK(S)

(D1.1). Rex $= e^*j^+$; $\Sigma = \{x, h, e, j\}$.

A. $jjjj$
B. $eeeeejjjjjj$
C. $xxhhhhhhhhhhjjjj$
D. $xxxxxxxxxxee$
E. $xxjjjjjjj$

(D1.2). Rex $= q^*f^+$; $\Sigma = \{l, u, q, f\}$.

A. $luuuuuuuuuqqqffffffff$
B. $uuuuuuuuuu$
C. $lllllllll$
D. $qqqqqffffffff$
E. $lllllllffff$

(D1.3). Rex $= p^*a?$; $\Sigma = \{r, w, p, a\}$.

A. $rrrrrwwwwwpppppppaaaaaaaa$
B. $rrrrrrrrrwwwwwwwwwwwaaaaaaa$
C. $pppppppp$
D. $rrrrrrr$
E. $wwwwwwwwwwwpppppppppp$

(D1.4). Rex $= r^*m^*$; $\Sigma = \{b, d, r, m\}$.

A. ddd
B. $bbbbbbbbbrrrrrrr$
C. $bdddddddddrrrmmmmm$
D. $dddrrrrrrrmmmmmm$
E. rr

(D1.5). Rex $= r^+qx^+$; $\Sigma = \{m, a, d, r, q, x\}$.

A. $maaaaaqqqxxxxxxxxx$
B. $mmmmmaaaaaaadddddqqqqqxxxxxxxx$
C. $rrrqxxxxxx$
D. $mmmaddddddddd$
E. $mmmmmmmmmmaaaaaaaaaqqqqqxxxxx$

(D1.6). Rex $= k^+e^+$; $\Sigma = \{y, q, k, e\}$.

A. $kkkkkkkk$
B. $eeeeeeeee$
C. $kkkkkkkkkkeeeeeeeeee$
D. $kkkkeeeeeeeee$
E. $yyyyyyqqqqqq$

(D1.7). Rex $= bd^+$; $\Sigma = \{h, e, b, d\}$.

A. $hhhhhhhhbbbbbbddddddddd$
B. $hhhhhhhhhheeeeebbbbbbbbbbdddd$
C. $eeeeeeeeeebbbbbbbddddd$
D. $eeeeeeeeebdddd$
E. bdd

(D1.8). Rex $= q?t^+$; $\Sigma = \{r, i, q, t\}$.

A. q
B. $iiiiiiiiqqqqttttttt$
C. $rrrrrrriiiii$
D. $rrrrrrriiiiqqqqqqt$
E. $ttttttt$

(D1.9). Rex $= k^+tz^*$; $\Sigma = \{d, x, f, k, t, z\}$.

A. $ddddddddddxxxxxzzz$
B. $ddddddddfftttttttzzzz$
C. $kkktzzzzzz$
D. $dddxxxxfffffffkkkkkkkkkktttz$
E. $xxxxxfffffffffkk$

(D1.10). Rex $= r^+w^+$; $\Sigma = \{k, x, r, w\}$.

A. xx
B. $kkkkkkkkx$
C. $kkxxrrrrrrrrrrwwwwwwwwww$
D. $rwwwwww$
E. $kkkkkkxxxxxxxxxxrrwwwwwwwwww$

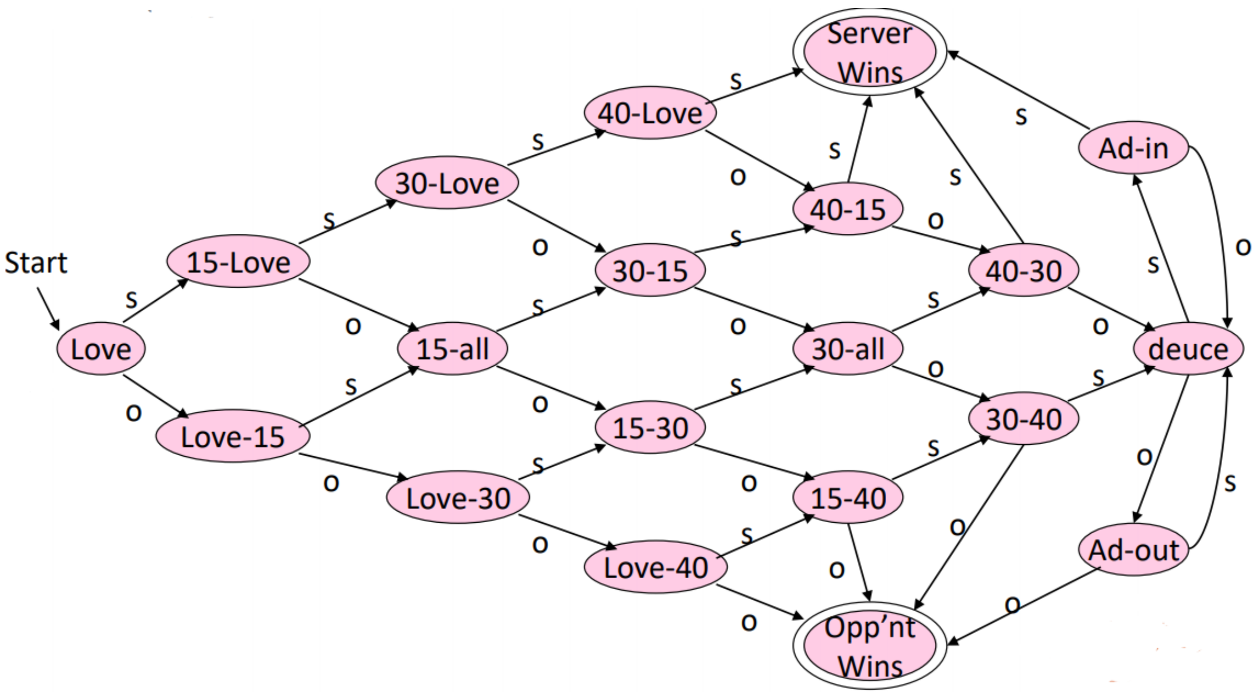

Figure 2: The Tennis Automaton

2. In the Tennis Automaton shown in Figure 2, is the string $(so)^4o^2s^2$ accepted? If yes, list the states that lead to the accepting state. If no, state why?

4 MARK(S)

3. The language of the set of strings of 0's and 1'a with no consecutive one's, produces a set whose length of strings are similar to a particular mathematical sequence? What is this sequence?

2 MARK(S)

4. In program compilation or translation, a parse tree or IR (Intermediate Representation) can be passed to at least four different components for further processing. What are these components.

4 MARK(S)

5. What is Lazy Evaluation?

2 MARK(S)

6. What stage of compilation deals with the checking of the source code against the grammar of the programming language?

3 MARK(S)

Section E: Natural Language Processing (15 marks)

Recall the Email Extractor Application developed during this module. The screenshot of this application is shown in Figure 3.

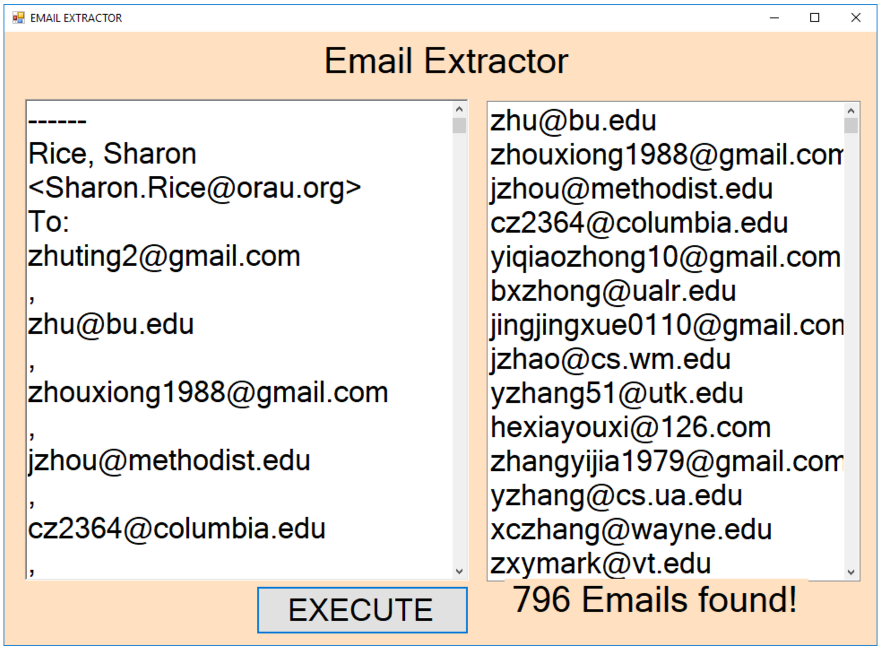

Figure 3: Email Extractor Application

The following are code snippets from this Application. Study them, and answer the following questions about the source code of this application.

Listing 1: Execute Button

```
Private Sub btnExecute_Click(sender As Object, e As EventArgs) Handles btnExecute.Click
    Dim aText As String = txtRawText.Text.ToString.Replace(",", " ")
    Dim arr As New ArrayList
    arr.AddRange(aText.Split(" "))
    For Each itm As String In arr
        If isEmail(itm.ToString.Trim) Then
                ListResult.Items.Add(itm)
        End If
    Next
    Label1.Text = ListResult.Items.Count.ToString() & " Emails found!"
End Sub
```

Listing 2: IsEmail Method

```
Private Function isEmail(aStr As String) As Boolean
    Dim result As Boolean = False
    Dim text_or_digit = "[A-Za-z0-9_]+"
    Dim email_re As String = "^" & text_or_digit & "(\@)" & text_or_digit & "(\.)" & text_or_digit & "$"
    Dim email_re2 As String = "^" & text_or_digit & "(\@)" & text_or_digit & "(\.)" & text_or_digit & "
       (\.)" & text_or_digit & "$"
    If StringPatternRecognition.doesRegexRecogniseValue(email_re, aStr) Then
        result = True
    ElseIf StringPatternRecognition.doesRegexRecogniseValue(email_re2, aStr) Then
        result = True
    End If
    Return result
End Function
```

1. The code fragment in Listing 1 runs when the EXECUTE button is clicked. What is the function of Line 2 in Listing 1?

   (a) To offer some form of normalisation of text before processing

   (b) To do syntax analysis on the raw text

   (c) To normalise the regular expression

   (d) To add emails to the ListBox

   (e) It compares a regular expression to a string

4 MARK(S)

2. What is `arr` in the code in Listing 1?

   (a) A variable that stores all tokenised words from the raw text

   (b) A variable that stores all recognised emails in the raw text

   (c) An arraylist of integers

   (d) An arraylist of splited emails

   (e) A class for recognising strings

4 MARK(S)

3. Modify the code in Listing 2 to allow ONLY emails addresses that end with "`.up.org.za`".

7 MARK(S)

Appendix A

2-1-1
2-1-2
2-3-1
2-3-2
2-5-1
2-5-2
2-7-1
2-7-2
4-1-1
4-1-2
4-3-1
4-3-2
4-5-1
4-5-2
4-7-1
4-7-2
6-1-1
6-1-2
6-3-1
6-3-2
6-5-1
6-5-2
6-7-1
6-7-2