MODULE:	Advanced Programming
CODE:	APM11A1
DATE:	25th May, 2019
DURATION:	3 Hours
TIME:	08h30
TOTAL MARKS:	160 (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 160 marks available. 160 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 (40 marks)

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

   1 MARK(S)

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

   1 MARK(S)

3. 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)

4. Traditionally, an algorithm is called Divide-and-Conquer, if?

   2 MARK(S)

5. What class of algorithms are based on Depth-first search?

   2 MARK(S)

6. In one sentence, state why algorithms are grouped together.

   2 MARK(S)

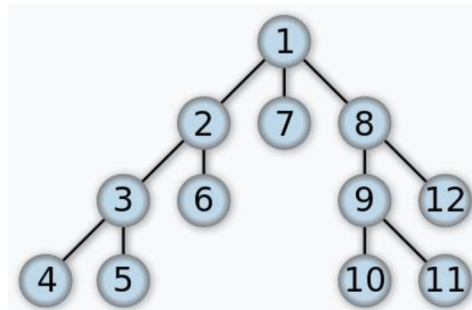7. Give two examples of Divide-and-Conquer algorithm.

   2 MARK(S)

8. Computing the Fibonacci sequence with recursive calls is computational expensive. State the time complexity of this algorithm.

   2 MARK(S)

9. What data structure can deck of cards be compared to?

   2 MARK(S)

10. What type of search algorithm is implied by the tree below?

2 MARK(S)

11. What class of algorithms remember past results and use them to find new ones?

2 MARK(S)

12. What class of algorithms require Optimal Sub-structures, and Overlapping Sub-problems?

2 MARK(S)

13. What class of algorithms work by *choosing local optimum at each step, with the hope of ending up with a global optimum*?

2 MARK(S)

14. List any two class of algorithms that are used to solve optimisation problems?

2 MARK(S)

15. Describe the Travelling Sales Person (TSP) problem.

2 MARK(S)

16. Thabang is trying to hack into his girlfriend's high technology purse. This purse is protected with a four-digit code. He decides to try all the possibilities/permutation of digits. What class of algorithm is this?

2 MARK(S)

17. A class of algorithms is known for requiring exponential time. This can be improved with heuristics, and/or optimisation. What class is this?

2 MARK(S)

18. 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)

19. In divide and conquer algorithms, sub-problems always overlap?

    (a) True

    (b) False

    (c) Depends on the programming language chosen for implementation

    (d) Depends on the paradigm chosen for implementation

    (e) ONLY in complex problems

    2 MARK(S)

20. 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)

Section B: Algorithms and Advanced Data Structures (40 marks)

1. Write an algorithm that computes the sum of **n** integer numbers.

<div align="right">

5 MARK(S)

</div>

2. Write an algorithm to read in two fractions of the form:

$$\frac{x}{y}$$

and calculate the sum of these fractions.

<div align="right">

5 MARK(S)

</div>

3. With the aid of an algorithm, describe the enqueue operation of a Queue object.

<div align="right">

5 MARK(S)

</div>

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

<div align="right">

10 MARK(S)

</div>

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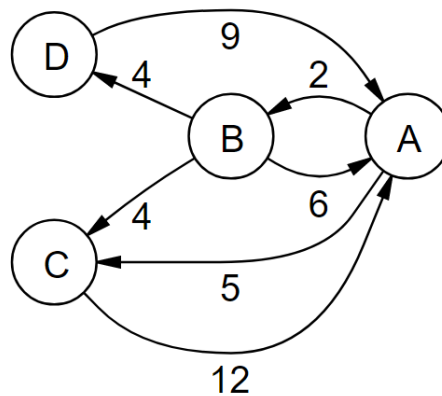

Figure 1: A directed graph, $G = \{V, E\}$

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

<div align="right">

10 MARK(S)

</div>

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

<div align="right">

3 MARK(S)

</div>

3. Is this graph strongly connected? If yes, why? If no, why not?

<div align="right">

2 MARK(S)

</div>

Section C: Probabilistic Algorithms and Simulations (40 marks)

1. Write a program to generate 300 random PINs of 12 digits each.

   10 MARK(S)

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

   10 MARK(S)

3. Zizile is observing cars on the highway in Johannesburg. The probability that the colour of a passing car is white is given as 0.71. Write a program that determines the colour of 200 cars on any given day.

   10 MARK(S)

4. Write a **<u>recursive function</u>** `GetStringYX(int k)` that returns a string containing $k-1$ hyphenated Y's and a single 'X' at the tail end. For example, if k is 4, the string to be displayed is "$Y-Y-Y-X$". If $k = 7$, "$Y-Y-Y-Y-Y-Y-X$" 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 $= i^+n^+$; $\Sigma = \{r, p, i, n\}$.

A. *rriiiiiiiii*
B. *rrrrrrrpppppnnnnnnnn*
C. *rrrrrrrrrr*
D. *rppii*
E. *iiinnnn*

(D1.2) Rex $= k^*u$?; $\Sigma = \{c, l, k, u\}$.

A. *k*
B. *cccccccuuuuuuuuuu*
C. *ccc*
D. *llllllllkkkuuuuuuuuu*
E. *ccclllkkkkk*

(D1.3) Rex $= g^*gj$?; $\Sigma = \{t, u, a, g, j\}$.

A. *ttttttttt*
B. *ggj*
C. *tttagggggggggggggjjjjjjjjjjjj*
D. *tttttttttuuuuuuuuuuuaaaaa*
E. *tttuuuuuuuuuuuaaaaaaajjjjjjjjjjj*

(D1.4) Rex $= n^*y$?; $\Sigma = \{c, p, n, y\}$.

A. *pppyyyyyyy*
B. *ppppppppppnnnnnnyyyyyyyyyyy*
C. *nnnnnnnnnn*
D. *ccccccccccpppppppp*
E. *ccccnnnnnnnnyyyyyyyyyy*

(D1.5) Rex $= c^*e^+$; $\Sigma = \{c, f, e\}$.

A. λ
B. *ccccccffff*
C. *fffffeeeeee*
D. *eeeeeeeee*
E. *ffeee*

(D1.6) Rex $= w|s^+$; $\Sigma = \{a, s, w\}$.

A. *aaaaaaaswwwwwwwww*
B. *ssss*
C. *aaaaaassssssssss*
D. *aaaaawwwwwww*
E. *aaaaaas*

(D1.7) Rex $= i^*r$?; $\Sigma = \{n, s, i, r\}$.

A. *r*
B. *ssi*
C. *nnn*
D. *nnnrr*
E. *sirrrrr*

(D1.8) Rex $= nz^*$; $\Sigma = \{x, c, n, z\}$.

A. *nnnnnnnnzzzzzzzzzz*
B. *xcccccccc*
C. *xxxxx*
D. *cnnnnnnnzzzzzzz*
E. *nzzzzzzzzzz*

(D1.9) Rex $= y^+h^*$; $\Sigma = \{c, t, y, h\}$.

A. *tttttyyyyyyyyyyhhhhhh*
B. *ccccccccctttttttyyy*
C. *cccctttttttyyyyyyyyyyy*
D. *yyyyyyyhhh*
E. *tttttttttyyyyyyyy*

(D1.10) Rex $= l?p^+$; $\Sigma = \{n, k, l, p\}$.

A. *kkkkkk*
B. *kkkkkkkllll*
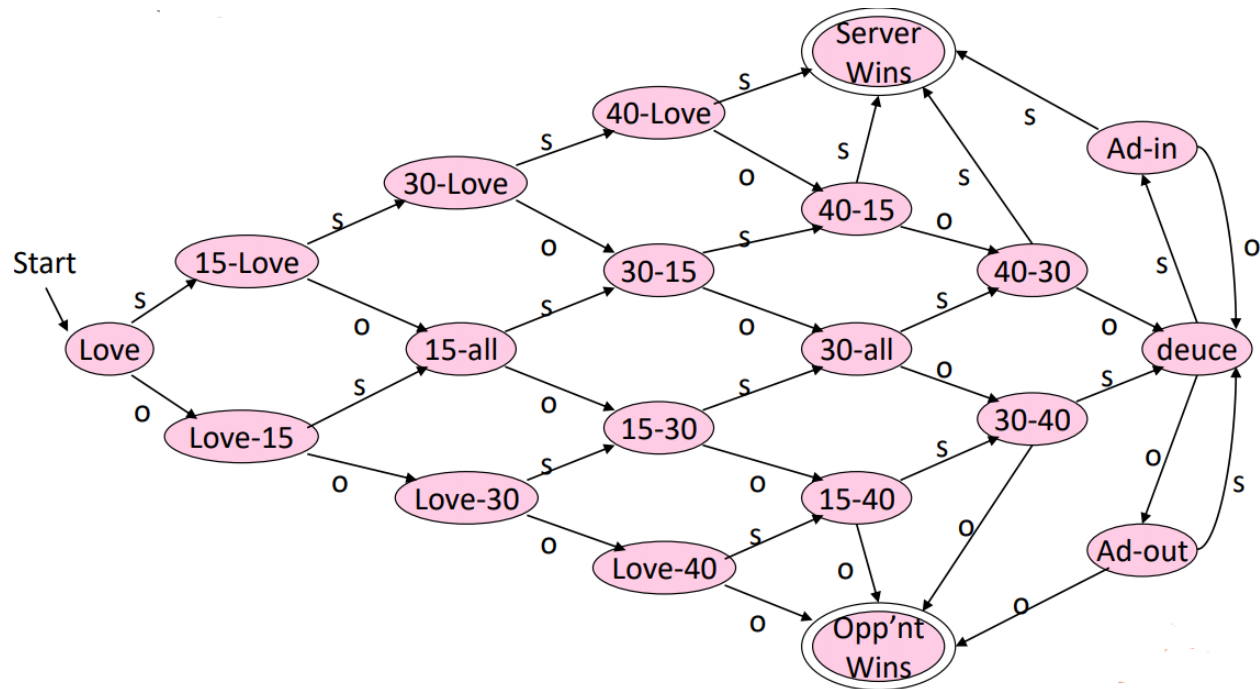C. *pppppp*
D. *llll*
E. *nnnnnnnn*

Figure 2: The Tennis Automaton

2. In the Tennis Automaton shown in Figure 2, is the string $(so)^4s^2$ 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?

3 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?

2 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.
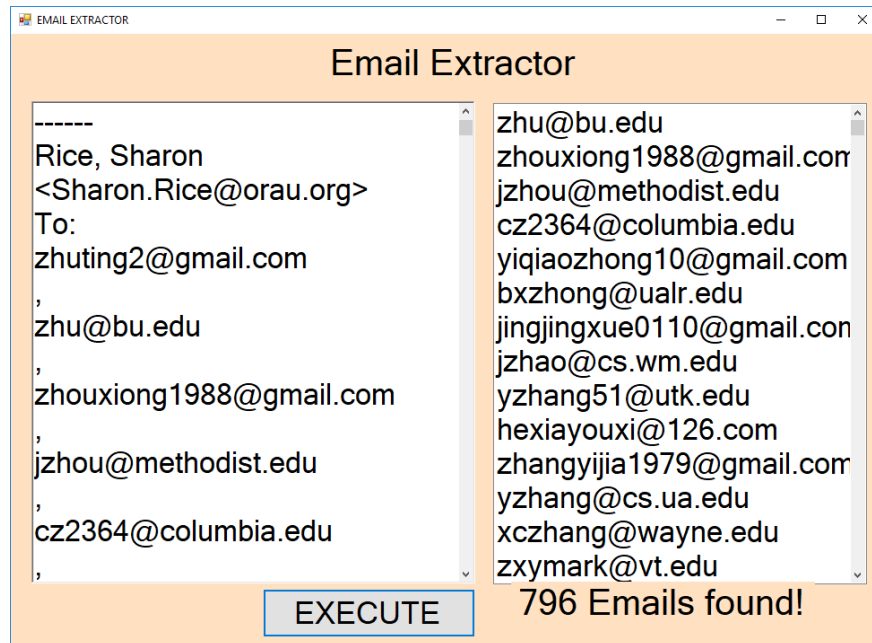

Figure 3: Email Extractor Application

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

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

Listing 2: IsEmail Method

```
1  Private Function isEmail(aStr As String) As Boolean
2      Dim result As Boolean = False
3      Dim text_or_digit = "[A-Za-z0-9_]+"
4      Dim email_re As String = "^" & text_or_digit & "(\@)" & text_or_digit & "(\.)" & text_or_digit & "$"
5      Dim email_re2 As String = "^" & text_or_digit & "(\@)" & text_or_digit & "(\.)" & text_or_digit & "
            (\.)" & text_or_digit & "$"
6      If StringPatternRecognition.doesRegexRecogniseValue(email_re, aStr) Then
7          result = True
8      ElseIf StringPatternRecognition.doesRegexRecogniseValue(email_re2, aStr) Then
9          result = True
10     End If
11     Return result
12 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 ".`uj.ac.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