

NAME:

ALGORITHMICS UNIT 3 & 4**Trial Exam 2: 2020**

Reading Time: 15 minutes
Writing time: 120 minutes (2 hours)

QUESTION AND ANSWER BOOK

<i>Section</i>	<i>Number of questions</i>	<i>Number of questions to be answered</i>	<i>Number of marks</i>
A	20	20	20
B	8		80

- Students are permitted to bring into the examination room: pens, pencils, highlighters, erasers, sharpeners, rulers and one scientific calculator.
- Students are NOT permitted to bring into the examination room: blank sheets of paper and/or correction fluid/tape

Materials supplied

- Question and answer book of 20 pages
- Answer sheet for multiple-choice questions

Instructions

- Write your student number in the space provided above on this page.
- Check that your name and student number as printed on your answer sheet for multiple-choice questions are correct, and sign your name in the space provided to verify this.
- All written responses must be in English, point form is preferred.

Students are NOT permitted to bring mobile phones and/or any other unauthorised electronic devices into the test room.

The VCAA Exam will include the Master Theorem in this form.

Use the Master Theorem to solve recurrence relations of the form shown below.

$$T(n) = \begin{cases} aT\left(\frac{n}{b}\right) + kn^c & \text{if } n > 1 \\ d & \text{if } n = 1 \end{cases} \quad \text{where } a > 0, b > 1, c \geq 0, d \geq 0, k > 0$$

$$\text{and its solution } T(n) = \begin{cases} O(n^c) & \text{if } \log_b a < c \\ O(n^c \log n) & \text{if } \log_b a = c \\ O(n^{\log_b a}) & \text{if } \log_b a > c \end{cases}$$

The VCAA form of Master Theorem is equivalent to the form of Master Theorem taught in our class by consideration of log laws.

$$\log_b a = c \Leftrightarrow a = b^c \Leftrightarrow \frac{a}{b^c} = 1$$

$$\log_b a < c \Leftrightarrow a < b^c \Leftrightarrow \frac{a}{b^c} < 1$$

$$\log_b a > c \Leftrightarrow a > b^c \Leftrightarrow \frac{a}{b^c} > 1$$

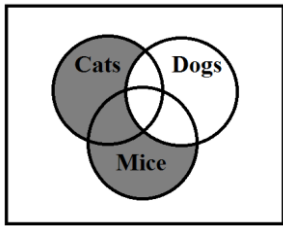
$$T(n) = aT\left(\frac{n}{b}\right) + f(n^k)$$

- $\frac{a}{b^k} < 1$ then $O(n^k)$
- $\frac{a}{b^k} = 1$ then $O(n^k \log_b n)$
- $\frac{a}{b^k} > 1$ then $O(n^{\log_b a})$

SECTION A – Multiple Choice – select one option only

Question 1

Which expression corresponds to elements in the shaded area shown in the Venn Diagram?



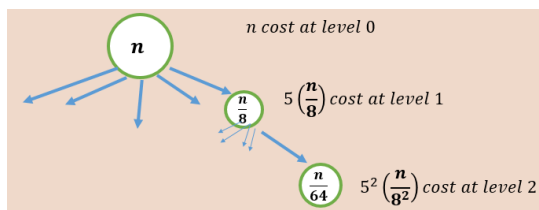
- A. If (NOT DOGS AND (CATS AND MICE)) then
- B. If (NOT DOGS OR (CATS OR MICE)) then
- C. If (NOT DOGS AND (CATS OR MICE)) then
- D. If (NOT DOGS AND NOT (CATS OR MICE)) then

Question 2

Which of the following statements is true about graph properties?

- A. To find the diameter of a graph, first find the shortest path between each pair of vertices. The greatest length of any of these paths is the diameter of the graph.
- B. To find the diameter of a graph, find the longest path between each pair of vertices.
- C. To find the diameter of a graph, first find the shortest path between each pair of vertices. The shortest length of any of these paths is the diameter of the graph.
- D. To find the diameter of a graph, first find the longest path between each pair of vertices. The smallest length of any of these paths is the diameter of the graph.

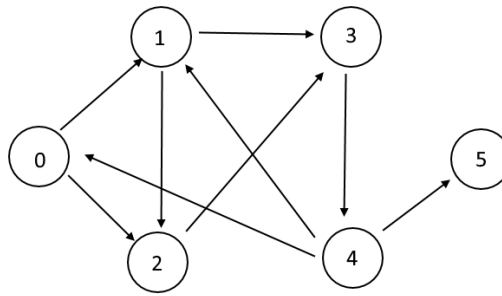
Question 3



The recurrence relation from the call tree shown above where at each level an algorithm makes 5 recursive calls to itself and each recursive call splits the input into eighths could be:

- A. $T(n) = 8T\left(\frac{n}{5}\right) + 8n, T(1) = 1$
- B. $T(n) = T\left(\frac{n}{8}\right) + 5n, T(1) = 1$
- C. $T(n) = 8T\left(\frac{n}{5}\right) + n, T(1) = 1$
- D. $T(n) = 5T\left(\frac{n}{8}\right) + n, T(1) = 1$

Question 4



The Depth first search node processing order on the directed graph shown above starting at node 0 **cannot** be:

- A. 0, 1, 3, 4, 5, 2
- B. 0, 2, 3, 4, 1, 5
- C. 0, 1, 2, 3, 4, 5
- D. 0, 1, 3, 2, 4, 5

Question 5

A big- O estimate for the number of operations, where an operation is an addition or a multiplication, used in this segment of an algorithm is:

```
t := 0
for i := 1 to 3
  for j := 1 to 4
    t := t + ij
```

- A. $O(n)$
- B. $O(1)$
- C. $O(\log n)$
- D. $O(n^2)$

Question 6

Which of the following statements is **not** true

- A. **greedy algorithm:** an algorithm that makes the best choice at each step according to some specified condition
- B. **tractable problem:** a problem for which there is a worst-case polynomial-time algorithm that solves it
- C. **intractable problem:** a problem for which no worst-case polynomial-time algorithm exists for solving it
- D. **undecidable problem:** a problem that can be solved by an algorithm

Question 7

The missing logic from the Floyd Warshall algorithm shown at the right here for determining transitive closure are in consecutive order of the empty boxes:

- A. $1, 1, (T[i,k] \text{ AND } T[k,j])$
B. $1, 0, (T[i,k] \text{ AND } T[k,j])$
C. $1, 0, (T[i,j] \text{ AND } T[j,k])$
D. $0, 1, (T[i,k] \text{ AND } T[k,j])$

```
Algorithm FloydWarshallTransitiveClosure(Input: G)  
// let T be a  $|V| \times |V|$  boolean array of Transitive Closure)  
// initialized to run on directed graph  $G=(V,E)$   
for i from 1 to  $|V|$  do  
    for j from 1 to  $|V|$  do  
        if ((i is equal to j) OR (edge i-j exists)) then  
            T[i,j] :=   
        else  
            T[i,j] :=   
        End if  
    End do  
End do  
// build up the array of reachability  
for k from 1 to  $|V|$  do  
    for i from 1 to  $|V|$  do  
        for j from 1 to  $|V|$  do  
            T[i,j] := T[i,j] OR   
        End do  
    End do  
End do  
End Algorithm
```

Question 8

Which of the following techniques uses randomness for avoiding getting trapped in local maxima?

- A. Best first search
B. Local beam search
C. Simulated annealing
D. Gradient descent

Question 9

The following operations may be used on a Dictionary Abstract Data type.

- new(dictionary)
- insert(key,value,dictionary)
- delete(key,dictionary)

Another common operation on a Dictionary Abstract Data type is:

- A. find(key,dictionary)
B. enqueue(key,dictionary)
C. pop(key,dictionary)
D. front(key,dictionary)

Question 10

Consider the following function defined in pseudocode:

```
function fibonacci(integer n) {  
    if (n < 2) {  
        return n;  
    }  
    if (value=find(key=n,dictionary) exists) {  
        return value;  
    }  
    fiboNumber = fibonacci(n - 1) + fibonacci(n - 2);  
    insert(key=n,fiboNumber,dictionary);  
    return fiboNumber;  
}
```

The space complexity of the fibonacci function above is:

- A. $O(n)$
- B. $O(n^3)$
- C. $O(n!)$
- D. $O(k^n)$

Question 11

What is the time complexity of the brute force algorithm used to solve the 01- Optimal Knapsack problem?

- A. $O(n)$
- B. $O(n!)$
- C. $O(2^n)$
- D. $O(n^3)$

Question 12

In a modified Mergesort, the input array is split into 3 thirds of the length(n) of the array, resulting in a split of $\frac{n}{3}$, $\frac{n}{3}$ and $\frac{n}{3}$ elements respectively. Which of the following is the tightest upper bound on time complexity of this modified Mergesort?

- A. $O(\log_3 n)$
- B. $O(n \log_2 n)$
- C. $O(n \log_3 n)$
- D. $O(n^3)$

Question 13

A greedy algorithm uses a heuristic function to

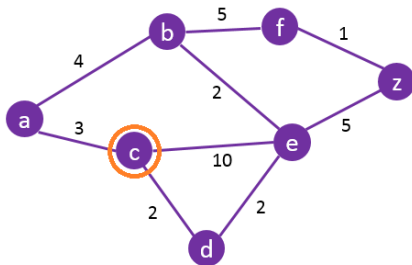
- A. expand the node that appears to be closest to the goal
- B. expand the node that is closest to the goal
- C. expand the node that is the most expensive
- D. expand the leftmost node

Question 14

Which of the following problems **cannot be** solved by backtracking method?

- A. n-queen problem
- B. a sudoku puzzle
- C. hamiltonian circuit problem
- D. travelling salesman problem

Question 15



Running Dijkstra's algorithm on the weighted graph above starting at node C will result in the following order of node expansion and processing:

- A. c, d, a, b, e, z, f
- B. c, d, e, a, b, z, f
- C. c, d, e, a, b, z, f
- D. c, d, a, e, b, z, f

Question 16

The missing terms for the following statement

"Prim's algorithm is a _____ algorithm, that initialises with _____."

- A. Divide and conquer, vertex
- B. Greedy algorithm, vertex
- C. Dynamic Programming, edge
- D. Backtracking, vertex

Question 17

Problems that can be solved in polynomial time are known as?

- A. intractable
- B. tractable
- C. decision
- D. complete

Question 18

Who did **not** invent the Turing machine?

- A. Alan Turing
- B. Turing
- C. Mr. Turing
- D. None of the above

Question 19

Which of the following problems **is not** NP complete?

- A. Hamiltonian circuit
- B. Decision version of the Travelling Salesman problem
- C. Searching a network graph
- D. Halting problem

Question 20

The missing words that are most appropriate as a description of Hilbert's Program in the following sentence:

"Hilbert's Program calls for a of all of mathematics in form, together with a proof that this axiomatization of mathematics is , using methods."

are:-

- A. differentiation, theoretical, consistent, finitary
- B. formalization, axiomatic, consistent, infinitary
- C. formalization, axiomatic, consistent, finitary
- D. classification, basic, consistent, infinitary

SECTION B – Extended Response Questions Answer all questions in the space provided.

Question 1 (10 marks)

A Sudoku puzzle is created in a matrix of **k elements × k elements** with some values already set. The solution to the puzzle has the integers **1 to k** appear once in each row, column and cage.

Example: k=9 rows, columns, cages, where each cage is a sub-array of 3x3 cells.

5			1			4		
2	7	4				6		
	8		9		4			
8	1		4	6		3		2
		2		3		1		
7		6		9	1		5	8
			5		3		1	
		5				9	2	7
1				2				3

With numbered cages

1	2	3						
4	5	6						
7	8	9						

The top right hand corner row=*a*, column=*b* of each cage numbered sequentially from left to right, top to bottom is given by the provided modules **geta(cageNumber)** and **getb(cageNumber)**

(*a*,*b*)

cageNumber

Function **geta(cageNumber, k)**

// returns the top right hand corner cell
 // row number for a cage number
 // of a $k \times k$ Sudoku
 // the **ceil** of a real number will round
 // it up to the next highest integer
 return $(\sqrt{k} \times \text{ceil}(\frac{\text{cageNumber}}{\sqrt{k}}) - \sqrt{k} + 1)$

End function

Function **getb(cageNumber, k)**

// returns the top right hand corner cell
 // column number for a cage number
 // of a $k \times k$ Sudoku
 // the **mod** returns the integer remainder
 // of the division of two integers
 If $(\text{mod}(\frac{\text{cageNumber}}{\sqrt{k}}) \text{ is } 0)$ then
 return $(k - \sqrt{k} + 1)$
 else
 return $(k - (\sqrt{k} - \text{mod}(\frac{\text{cageNumber}}{\sqrt{k}}) + 1)\sqrt{k} + 1)$

End if

End function

- a) Complete the module **checkRow** below in structured pseudocode that will check a Sudoku matrix row for correct values. **(2 marks)**

Module **checkRow(S,row,k)**

//Input: Sudoku matrix *S*, row of Sudoku, size *k* of Sudoku

End module

Question 1 (Continued)

- b) Complete the module **checkColumn** below that will check a Sudoku matrix column for correct values **(2 marks)**

Module checkColumn(S,column,k)

//Input: Sudoku matrix S, column of Sudoku, size k of Sudoku

End module

- c) Complete the module **checkCage** using the provided modules/functions **geta(cageNumber)** and **getb(cageNumber)** (from previous page) to check a Sudoku cage for correct values. **(3 marks)**

Module checkCage(S,cage,k)

//Input: Sudoku matrix S, cage number of Sudoku, size k of Sudoku

a:=geta(cage) //Find top right hand corner row=a of given cage number (see previous page)

b:=getb(cage) //Find top right had corner column=b of given cage number (see previous page)

End module

- d) Combine the modules **checkRow**, **checkColumn**, **checkCage** to make a complete Sudoku solution checking algorithm **checkSudoku** in structured pseudocode. **(3 marks)**

Algorithm checkSudoku(sudoku,k)

End algorithm

Question 2 (8 marks)

- a) For any connected graph, by definition there exists a path from any one node to another. Give a structured proof that Depth First Search (DFS) is always correct, being that it will always find a possible path, or return that there isn't one for any given graph input. **(4 marks)**

Consider the pseudocode for the Floyd-Warshall's shortest path algorithm.

(NOTE: you probably won't be given the F-W pseudocode in VCAA exam)

```
Floyd-Warshall's Algorithm
// initialise distance |V|×|V| matrix
set all distance[ , ] cells to infinity
for all weighted edges i-j in G update distance[i,j] with weight
for all nodes in G i update distance[i,i]=0
// relaxation
for k = 1 to n do
  for i = 1 to n do
    for j = 1 to n do
      if (i != k AND j != k) then
        if (distance[i,j] > distance[i,k] + distance[k,j]) then
          distance[i,j] = distance[i,k] + distance[k,j];
        end if
      end if
    enddo
  enddo
enddo
```

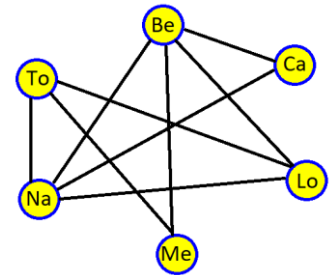
- b) Give a proof by contradiction of the correctness of the Floyd-Warshall algorithm for finding shortest paths between pairs of nodes in weighted, connected and undirected graphs. **(4 marks)**

Question 3 (14 marks)

The “**Cities Puzzle**” tries to place citizens of cities into a seating plan so that citizens **are not seated** next to other citizens in adjacent horizontal or vertical grouping of seats if the name of their city **shares any letter(s)**. For example citizens of Tokyo **can be** seated adjacent to citizens of Berlin since their cities do not share any of the same letters, but citizens of Berlin and Paris cannot be seated next to each other since they have the letters “i” and “r” in common.

Consider the following 6 cities, abbreviated in brackets: Berlin (Be), Caracas (Ca), London (Lo), Metz (Me), Nairobi (Na) and Tokyo (To).

These 6 cities are represented by a graph model (at the right) that indicates cities that **share** at least one same letter in their full name, which are connected by an edge.



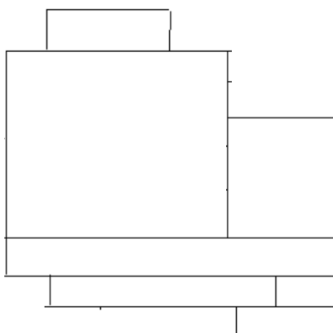
This model does not immediately solve the “**Cities Puzzle**”.

- a) Give the degree of each node in the graph model above. (1 mark)

- b) What property of graphs can be used to solve the “**Cities Puzzle**” for this example with the provided information? (1 mark)

- c) Place the names of these 6 cities into a graph Abstract Data Type model that indicates cities that **do not share** any same letters in their name. (2 marks)

- d) If the 6 cities are represented as 6 seating areas, label the adjacent seating areas with the cities that **do not share any** of the same letters in their name and solve the “**Cities Puzzle**” for this example. (2 marks)



Question 3 (continued)

- e) What well known and studied problem has the “**Cities Puzzle**” been transformed into? **(1 mark)**

- f) Is it possible to solve the “**Cities Puzzle**” for larger number of cities, say 50 cities? With consideration of your responses in parts a), b), c) and d) of this question. Explain what kind of information, abstract data types and associated algorithms could be used **to prepare the information** prior to finding a solution. Explain if there are any limitations on those algorithms. **(3 marks)**

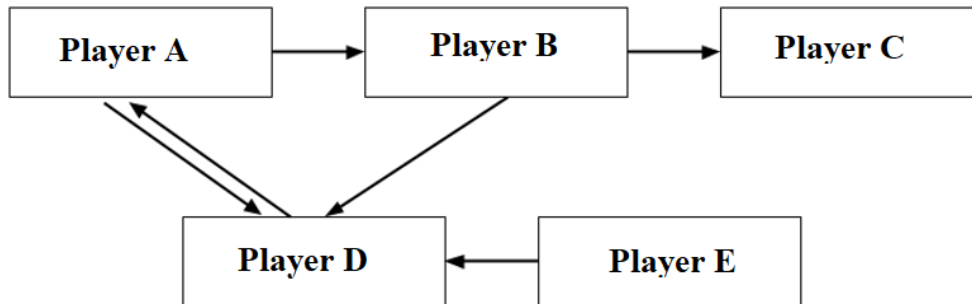
- g) Discuss the time complexity for solving the “**Cities Puzzle**” problem for a large number of cities, with an explanation of the feasibility of finding the solution. **(2 marks)**

- h) What class of problem is the “**Cities Puzzle**”? Explain and justify your conclusions. **(2 marks)**

Question 4 (9 marks)

Player A	Player B	Player C	Player D	Player E
				

In one particular Soccer season the **player to player** passing rate of 85% for 5 players A, B, C, D and E is shown by the directed graph below.



- a) Using the Page Rank algorithm show all the initialisation steps and values for each player's node that are actioned by the algorithm. **(2 marks)**

- b) Show recurrence relations used by the Page Rank Algorithm for each player $\Pr(A_{i+1})$, $\Pr(B_{i+1})$, $\Pr(C_{i+1})$, $\Pr(D_{i+1})$, $\Pr(E_{i+1})$ for finding the next iteration $(i+1)^{\text{th}}$ page rank of each player. **(3 marks)**

Question 4 (continued)



- c) Using the Page Rank algorithm find the ranking for each player after the **first iteration**. (2 marks)

The Soccer coach has statistics from a previous season, where the **player to player** passing rate for the same players in the same directions was **70%**.

- d) Explain how the Page Rank algorithm applies in detail for **Player A** in the previous season. (2 marks)

Question 5 (9 marks)

The optimal Travelling Salesman problem seeks to travel to each city in a connected network once only by the cheapest tour before returning home.



- a) Why is the Travelling Salesman Problem (TSP) studied so extensively in Computer Science? (1 mark)

- b) What other real world application can be formulated as the Travelling Salesman Problem? (1 mark)

Question 5 (continued)



- c) What exact algorithms exist to solve the optimal Travelling Salesman Problem? Briefly explain the time complexity of the exact algorithm and the amount of work it will do for 20 cities. **(2 marks)**

- d) What is the classification of the optimal Travelling Salesman Problem? Justify your answer. **(1 marks)**

- e) What other approaches exist for finding good solutions for the optimal TSP? Describe **three** of these approaches. **(4 marks)**

Question 6 (6 marks)

An iterative algorithm described in three steps for finding a Square Root approximation is shown below. This algorithm accepts a positive number S and finds the square root using an approximation algorithm that dates from ancient Babylonian times.

1. Begin with an arbitrary positive starting value x_0, that is between 1 and S, the closer to the actual square root of N, the better. 2. Let $x_{(n+1)}$ be the average of $\left(x_n + \frac{S}{x_n}\right)$, which is the arithmetic mean $x_{(n+1)} = \frac{1}{2}\left(x_n + \frac{S}{x_n}\right)$, and use it to approximate the geometric mean 3. If $\left(x_{(n+1)}\right)^2 \approx S$ to the required accuracy then you are done, otherwise repeat from step 2	<i>Here are the mathematical representations for the algorithm</i> It can also be represented as: $x_0 \approx \sqrt{S},$ $x_{n+1} = \frac{1}{2} \left(x_n + \frac{S}{x_n} \right),$ $\sqrt{S} = \lim_{n \rightarrow \infty} x_n.$
--	---

- a) Create a **recursive function** in structured pseudocode for square root approximation using the Babylonian algorithmic method that will only recur for a maximum of **maximum** times. **(5 marks)**

- b) Give an example of how your function from part a) will be called. **(1 marks)**

Question 7 (16 marks)

Queue Theory – is a real study, here is a limited model to study the order in which customers will be served by cashiers at a supermarket.



The conditions in this supermarket are:

- Each cashier spends the same amount of time with each customer
- The number of active queues will be between 1 and 3 inclusive.
- The number of customers, will be between 1 and 20 inclusive identified by a unique letter (A,B,C,...)

On Monday at 9am at this supermarket, there were 3 active queues:

- **Queue1**, Cashier number 1 spends 3 minutes on each customer, customers in this queue: **A, B, C**
- **Queue2**, Cashier number 2 spends 2 minutes on each customer, customers in this queue: **D, E, F, G**
- **Queue3**, Cashier number 3 spends 4 minutes on each customer, customers in this queue: **H, I, J**

- a) What was the order of service of customers A, B, C, D, E, F, G, H, I, J across the three queues with this Monday situation? **(1 marks)**

- b) Build a model using appropriate abstract data type(s) and operations to model the information of the Monday queues. **(2 marks)**

- c) Create a simple algorithm to process the information in the model you created in part b) and output the order of service of customers A, B, C, D, E, F, G, H, I, J. **(4 marks)**

Question 7 (continued)

A Turing machine is set up to service one queue.

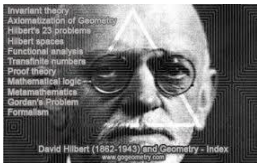
The states are:		The symbols are:	
IQ – Inactive Queue with no cashier AQ – Active Queue with a cashier SP – Serving Person Halt – Supermarket closed		C – Cashier B – Blank P – Person M – Manager	
Input received	If in state	Do this action	
C	IQ	Change state to AQ, move to next input	
M	IQ	Change state to Halt	
B	IQ	Remain in state IQ, move to next input	
C	AQ	Remain in state AQ, move to next input	
B	AQ	Remain in state AQ, move to next input	
M	AQ	Change state to IQ	
P	AQ	Change state to SP	
P	SP	Overwrite P with B, change to state AQ, move to next input	

d) Convert this Turing Machine above to directed graph form. (4 marks)

e) Why is the Turing machine considered so important in Computer Science with respect to the classification of problems and their algorithms? (3 marks)

f) What is the Church Turing thesis, how is it connected to the Turing machine and to the classification of problems? (2 marks)

Question 8 (8 marks)



- a) Describe the main goals of Hilbert's program, in terms of **Completeness**, **Consistency** and **Decidability**. (3 marks)

- b) How was Hilbert's program shown to be unachievable? Explain. (3 marks)

- c) How did Hilbert's program contribute to the foundations of Computer Science? (2 marks)

END OF TRIAL EXAM 2