SOLUTIONS

ALGORITHMICS UNIT 3 & 4

CHES Trial Exam 2: 2023

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

QUESTION AND ANSWER BOOK

Section	*Number of questions*	*Number of questions to be answered*	*Number of marks*
A	20	20	20
B	10	10	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 ?? 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 you 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.

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

- the Master Theorem for solving recurrence relations of the form:

$$T(n) = \begin{cases} a \cdot T\left(\frac{n}{b}\right) + kn^c & \text{if } n > 1 \\ d & \text{if } n = 1 \end{cases}$$

where $a > 0, b > 1, c \geq 0, d \geq 0, k > 0$

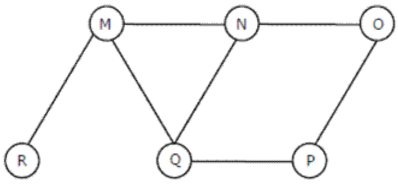
and its solution: $T(n) = \begin{cases} O(n^c) & \text{if } a < b^c \\ O(n^c \log(n)) & \text{if } a = b^c \\ O\left(n^{\log_b(a)}\right) & \text{if } a > b^c \end{cases}$

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

The Breadth First Search algorithm is run on the following graph. One possible order of visiting the nodes of the following graph is:



A. MNOPQR

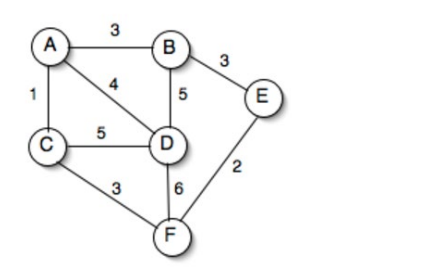
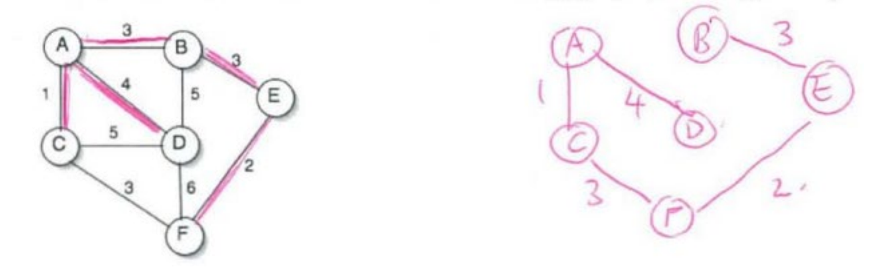
B. NQMPOR

C. QMNPRO

D. QMNPOR

Question 2

When Prim's is executed on the following graph, starting at any vertex:

The Edges (in no particular order) included and the weight of the MST are:

A. E={AC,AB,BE,EF,AD} weight=13

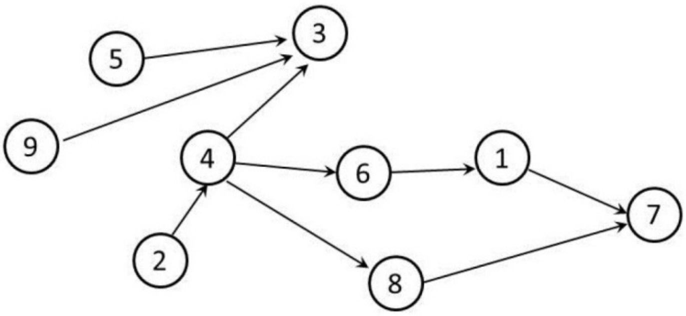
B. E={AC,AD,CF, EF,BE} weight=13

C. E={AC,AB,BE,EF,BD} weight=14

D. Both E={AC,AB,BE,EF,AD} weight=13 and E={AC,AD,CF, EF,BE} weight=13

Question 3

Which of the following sequences are topological sorts for the directed acyclic graph below?



A. 9, 2, 5, 4, 3, 8, 1, 6, 7

B. 9, 2, 5, 4, 8, 5, 1, 7, 3

C. 9, 2, 5, 3, 4, 6, 8, 1, 7

D. 2, 9, 5, 4, 6, 3, 8, 1, 7

Question 4

Which of the following sorting algorithms does not have a worst case running time of O(n^2)

- **A.** Insertion sort
- **B.** Merge sort
- **C.** Quick sort
- **D.** Bubble sort

Question 5

```
function quiz(i)
// Input integer 'i'

       if (i > 1)then

          quiz(i / 2)

          quiz(i / 2)

       end if

       print("*")

end function
```

How many asterisks are printed by the function call quiz(5)?

- **A.** 4
- **B.** 7
- **C.** 8
- **D.** Some other number

Question 6

Consider the following operations that are performed on an empty queue "Q”:

- Enqueue(Q,”ron”)
- Dequeue(Q)
- Enqueue(Q,”harry”)
- Enqueue(Q,”hermoine”)
- Enqueue(Q,”snape”)
- Dequeue(Q)

The contents of the queue Q after these operation is:

A. hermoine, harry

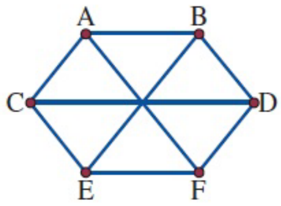
B. hermoine, snape

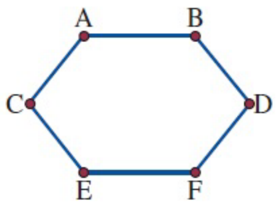
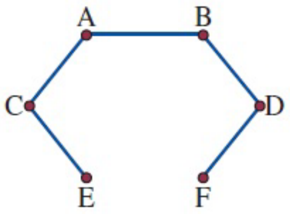
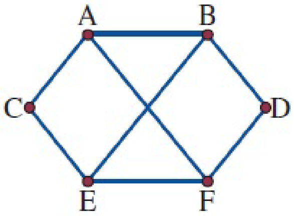
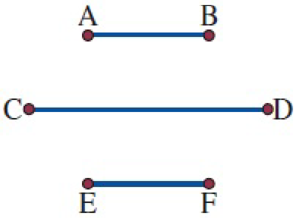
C. harry, hermoine, snape

D. ron, harry

Question 7

When converted to a spanning tree, the graph shown will resemble which of the following?



A		C	
B		D	

Question 8

The definition of the recursive function T(n) where n is in the set of Natural numbers is shown below:
T(n)=T(n-1)-1,where T(0)=20
The value of T(6) is:

A. 14

B. 13

C. 10

D. 15

Question 9

Which of the following problems is classified as being tractable?

A. Travelling Salesman Problem (TSP)

B. Finding a Spanning Tree in a simple graph

C. Knapsack 0/1 Problem

D. Hamiltonian Path Problem

Question 10

List refers to a list. If there are k items in the list, roughly how many iterations are there for the following pseudocode?

```
sum:=0
for each value in List do
    sum:=sum+value
enddo
```

A. k

B. k/2

C. 1

D. 2

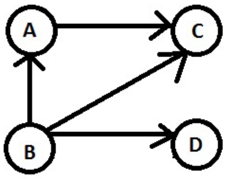
Question 11

The Transitive Closure matrix below matches which graph?

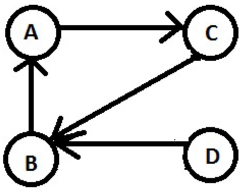
	A	B	C	D
A	1	0	0	0
B	1	1	0	0
C	1	1	1	0
D	1	1	0	1

Select one:

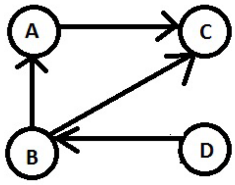
A



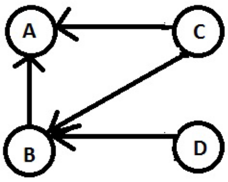
C



B



D



Question 12

If T(n) is a function that defines the time complexity of an algorithm based on the size of the input n:

T(0)=3, T(n)=2T(n-1), when n>0

Then the equivalent function expression for T(n) is:

A. $T(n)=3^n$

B. $T(n)=3(2^n)$

C. $T(n)=2^n+3$

D. $T(n)=3(2^{(n-3)})$

Question 13

Which of the following best describes the position of Strong AI?

A. Having a mind is a matter of having the right outputs

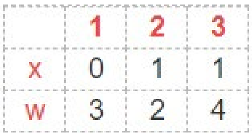
B. The principal value of computers is that they are powerful tools for studying the mind

C. Computers cannot be minds.

D. An appropriately programmed computer is a mind, in the sense that it can understand

Question 14

What is the output of the perceptron activation function, given the inputs of x1, x2, x3 with corresponding weights w1, w2, w3 respectively in the table shown and a bias value of 5?

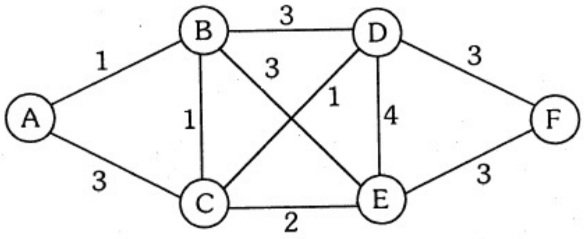


	1	2	3
x	0	1	1
w	3	2	4

- **A.** 1
- **B.** 0
- **C.** 6
- **D.** 5

Question 15

Dijkstra's algorithm on the following graph from node A, explores the nodes in order:



- **A.** A,B,C,D,E,F
- **B.** A,C,B,E,D,F
- **C.** A,B,C,E,D,F
- **D.** A,C,B,E,D,F

Question 16

The approach to algorithm design for optimization problems that makes direct use of the fact that the most apparent next component of a solution is part of the optimal solution is ..

- **A.** dynamic programming;
- **B.** divide and conquer;
- **C.** brute force;
- **D.** greedy;

Question 17

Master's theorem can be applied on which of the following recurrence relation?

A. T (n) = 2T (n/2) + 2^n

B. T (n) = 0.5T (n/3) + n

C. T (n) = T(n/2) + $2n^2$ + 1

D. None of these

Question 18

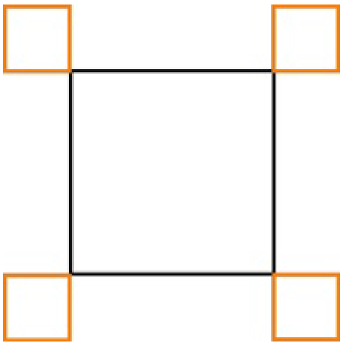
A robot has been programmed to draw rectangles. It can execute the following instructions:

- **Orange** draw an orange line of length 1
- **Black** draw a black line of length 1
- **Turn** turn 90° clockwise

Besides those simple instructions the robot can also execute complex instructions by combining instructions. If A and B are instructions (either simple or complex) the robot can do:

- **A, B** first execute A and then execute B
- **n×(B)** execute B n times

The robot must draw the following:



Which instructions will not create the drawing?

A. 4x(3xBlack, 3x(Orange, Turn), Orange)

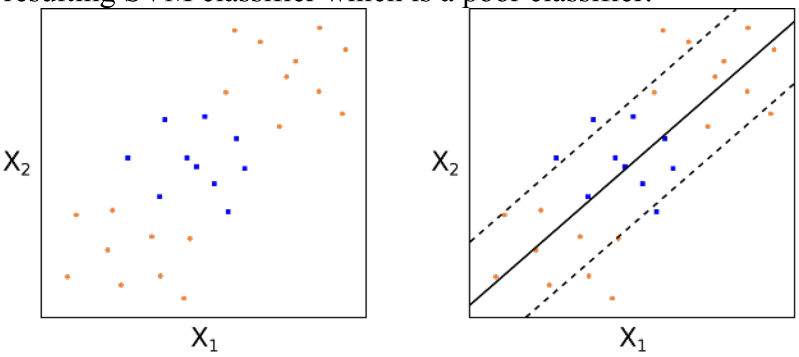
B. 4x(2x(Orange, Turn), Orange, 3x(Black), Orange, Turn)

C. 4x(2x(Orange, Turn), 3x(Black), 2x(Orange, Turn))

D. 4x(Black, 3x(Orange, Turn), Orange, 2x(Black))

Question 19

Consider the following scatterplot showing two classes of data against features X1 and X2 and the resulting SVM classifier which is a poor classifier.



What can be done to improve the automatic classification of the two classes using machine learning?

A. Use a perceptron neural network to learn the weights of W1 and W2.

B. Apply a linear transformation to both the original data features X1, X2.

C. Increase the width of the maximising SVM hyperplane margin for X1 and X2.

D. Apply a quadratic transformation to both the original data features X1, X2.

Question 20

Which of the following statements about the Turing Machine is false?

A. Lambda calculus can compute anything that any Turing Machine could possibly compute.

B. If P equals NP, then the Traveling Salesperson Problem can be solved in polynomial time by a deterministic Turing Machine.

C. The Turing machine is a universal model of computation: with a Turing machine we can solve any decision problem that can be solved with a Supercomputer or with a Pentium 4 running Linux.

D. Because the Halting Problem is unsolvable, it is impossible to tell if any algorithm has an infinite loop.

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

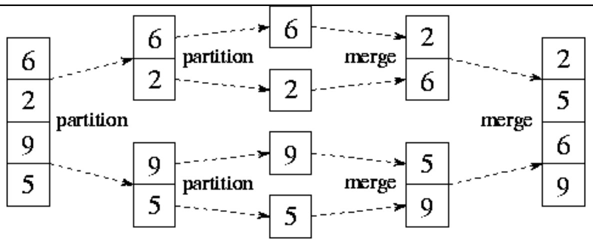
Question 1 (6 marks)

a. Using Pseudocode complete the MergeSort algorithm below for sorting data. ***(2 marks)***

Solution – should show the list being halved and the left half and the right half being passed recursively to Mergesort

```
Procedure Mergesort (Input A)
       Size=Length(A)
       If (Size <=1) then
              Return A
       else
              LeftA = Mergesort([A[1..Size/2])
              RightA = Mergesort([A[Size/2+1..Size])
              Return Merged(LeftA,RightA)
       End If
End Procedure
```

b. Show all steps either in words or by a labelled diagram that are required to sort the following list of integers using the MergeSort Algorithm. ***(2 marks)***

	Solution should show the splitting of data to the single element and then the merge phase combining the elements into the final solution.
6 2 9 5	 6 2 9 5 partition 6 2 partition 6 2 merge 2 6 9 5 partition 9 5 merge 5 9 merge 2 5 6 9

c. What is the design pattern that is used by the MergeSort Algorithm? Describe the general principles of this design pattern and discuss what kinds of problems lend themselves to being solved in this way. ***(2 marks)***

- Mergesort follows the Divide and Conquer design pattern.
- Problems that can be divided into independent sub-problems that can be solved and then merged to form the final solution

Question 2 (12 marks)

Consider the following algorithm **MinDistance** for finding the distance between the two closest elements in an array of unsorted unique integers.

```
Function MinDistance(A[1..n])
// Input: An array A[1..n] of unique integers
// Output: The minimum distance d between two of its elements
dmin := max
for i := 1 to n-1 do
  for j:= i+1 to n do
      temp := |A[i] – A[j]| // absolute difference of values
      if temp < dmin then
          dmin = temp
          minI = i
          minJ = j
return |minI-minJ|
end function
```

a) What is the result of MinDistance([1,4,2])? Justify your response with a full trace of variables i,j, temp, A[i],A[j], dmin, minI, minJ in step by step execution in the table below. (2 marks)

i	j	A[i]	A[j]	temp	dmin	minI	minJ	Value that is returned is 2
					Max	?	?	
1	2	1	4	3	3	1	2	
1	3	1	2	1	1	1	3	
2	3	4	2	2	1	1	3	

b) Analyse the time complexity of this naive algorithm. Giving a justification for your reasoning.(2 marks)

- The inner most loop of the algorithm above is executed (n-1)+(n-2)+(n-3)+….+1 times
- which can also be expressed as $\sum_{i=1}^{n-1} i = \frac{(n-1)n}{2}$ which leads to a time complexity of $O(n^2)$
- due to the fixed for loops without conditions the best and worst case are the same for this algorithm

Vivian and Daniel think that a more efficient algorithm could use a dictionary using the array element as the key and the index as the value.
Let Dmap be the label of the dictionary representation selected by Daniel and Vivian.

c) Write signature specifications for the following three operations of Dmap. (3 marks)

- Add a record to Dmap of a new array value and its index.
- Lookup the value in Dmap using the key A[k]
- Get all the keys of Dmap.

InsertDict: Dictionary × key × value → Dictionary
LookupDict: dictionary × key → value
KeysDict: Dictionary → List or KeysDict: Dictionary → Set

d) Using Vivian and Daniel's idea show the contents of the dictionary "Dmap" and the instance of # Test Case 1: A = [1, 9, 6, 8] (1 mark)

Key	Value
1	1
6	3
8	4
9	2

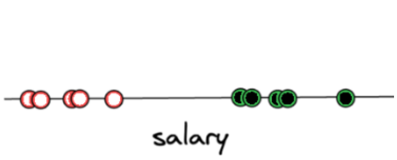
e) Using Vivian and Daniel’s idea write a new algorithm in structured pseudocode for solving the MinDistance problem using the dictionary Dmap. (3 marks)

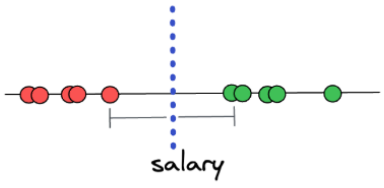
```
function minimumDistance(A):
//Input A[1..n] array of unique integers
   Create dictionary Dmap
   minDifference = biggestInteger
   minDistance = biggestInteger
   prevKey = -1
// Build the dictionary Dmap
   for i =1 to length(A) do
       if (A[i] not in the dictionary Dmap) then
            InsertDict(Dmap, A[i], i)   // Key=A[i], Value=i
       End if
   End do
// Determine the minimum distance in array positions of closest points
   foreach k in sorted(KeysDict(Dmap))) do
         if (prevKey > 0) then
            // Compare with the previous key in Dmap.
            If ((k – prevKey) < minDifference) then
                minDifference = (k -prevKey)
                currIndex = LookupDict(Dmap, A[k])
                prevIndex = LookupDict(Dmap, A[prevKey])
                minDistance = (currIndex – prevIndex)
         end if
         prevKey = k
   end do
   return minDistance

// Answers will vary
```

Question 3 (8 marks)

The data on ten people's yearly salaries was collected and labelled into two categories: either a person is underpaid or overpaid. In the example below, if a person is considered underpaid, then it will be labeled with red/white dots. Otherwise, if a person is considered overpaid, then it will be labeled with green/black dots.





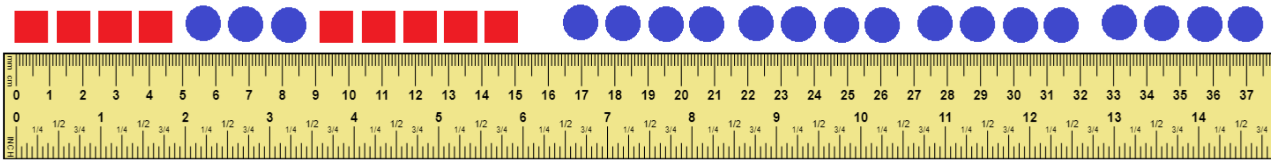
Answer

a) Indicate on the diagram above where an SVM would place the hyperplane classifier for this data. Justify your answer (2 marks)

The SVM classifier hyperplane that has an equal distance to each of the data points that are located on the edge, as you can see in the illustration. The distance between the hyperplane and each of the edge points is equal, find the widest margin possible, and this concept is called the Maximal Margin Classifier.

https://www.stratascratch.com/blog/machine-learning-algorithms-explained-support-vector-machine/

Suppose that we want to classify one dimensional data with one feature X1 into two classes squares and circles.
We have a set of training data that is plotted for feature X1 as follows in the diagram below with squares in position [0,5), circles in position [5,9), squares in position [9,15) and then only circles which extend toward the positive direction from [17,∞):



b) Can a single perceptron separate the data feature X1 for the circles and the squares? Explain why or why not and justify your response. (2 marks)

A single perceptron cannot linearly separate squares and circles for the input X1 as the output equation for the perceptron O=aX1+X0 can only be a straight line.

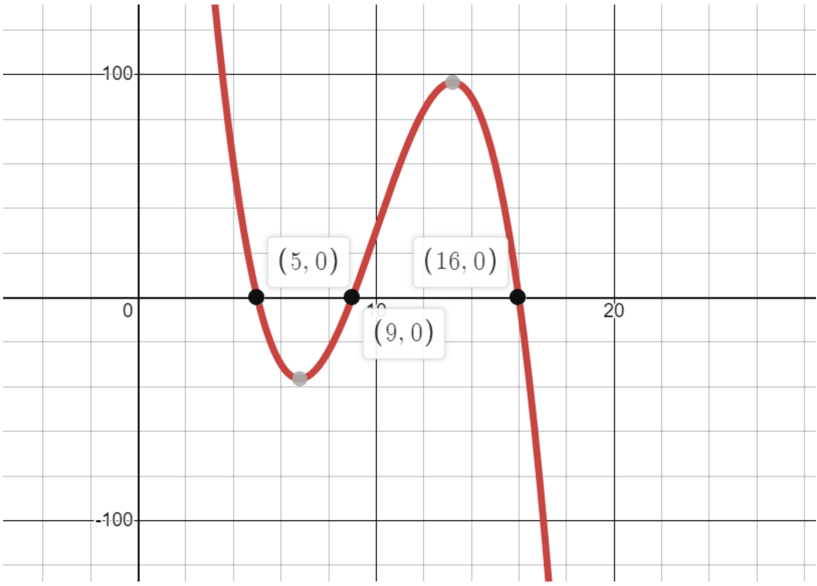
Question 3 (8 marks)

c) Can a Multilayer Perceptron Neural Network classify this data? Justify your response. (2 marks)

A multilayer Neural Network can classify non-linearly separable data, since it contains more than one perceptron in the hidden layer and have generate multiple classifiers.

d) Can a Support Vector Machine (SVM) machine learning algorithm classify this data? Justify your response (diagrams accepted). (2 marks)

An SVM can apply a transformation known as a kernel trick to add another dimension to the data to allow it to be separated by a hyperplane in 2D.
By applying the cubic transformation y=-(x-5)(x-9)(x-16) and adding another feature y, a cubic transformation the data can be used so that the data can be classified into circles and squares.
Where y<0 there are circles, where y>0 there are squares.



(NOTE: Answers will vary)

Question 4 (8 marks)

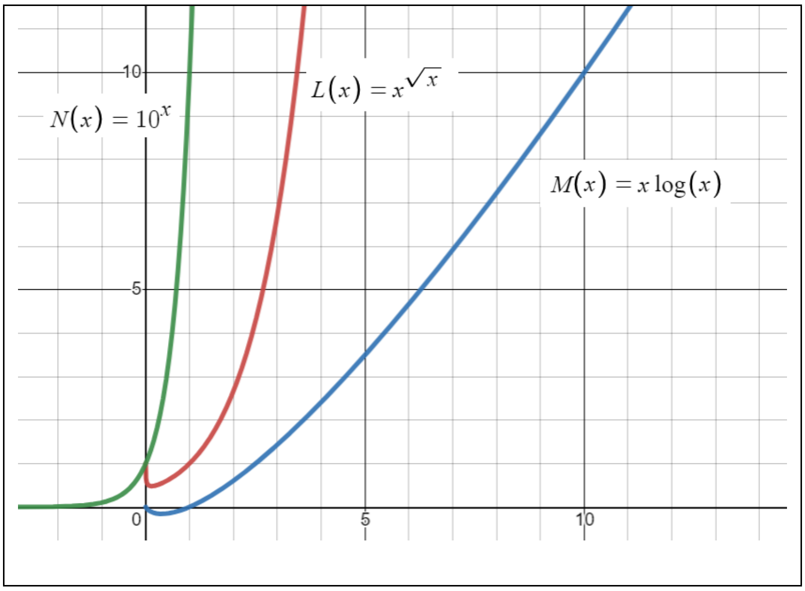
Consider the runtime complexity statistics of the following three algorithms: L, M and N.

	Time Complexity	**Space Complexity**
Algorithm L	$O(n^{\sqrt{n}})$	$O(n^4)$
Algorithm M	$O(nlogn)$	$O(nlogn)$
Algorithm N	$O(10^n)$	$O(n)$

a) If the log is base 10, evaluate the approximate time and space complexity for an input of 100 elements for each Algorithm. (2 marks)

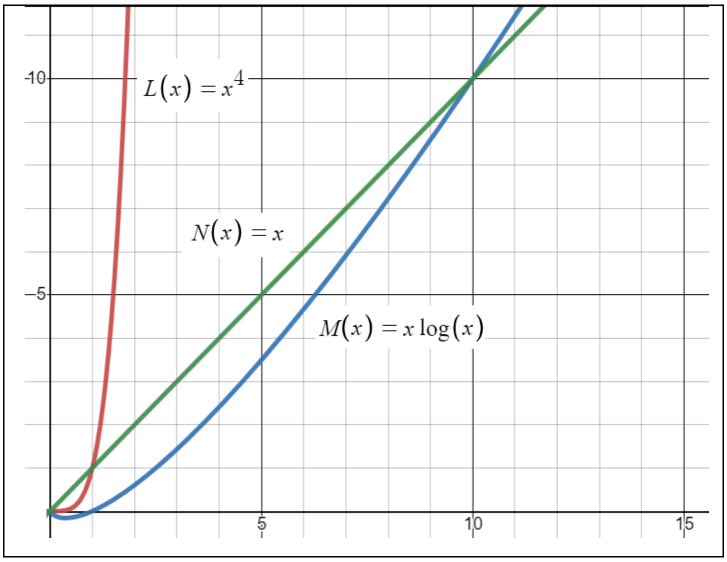
	Time Complexity for 100 elements	**Space Complexity for 100 elements**
Algorithm L	$O(100^{\sqrt{100}}) \approx (10^2)^{10} = 10^{20}$	$O(100^4) \approx (10^2)^4 = 10^8$
Algorithm M	$O(100log100) \approx 2(10^2)$	$O(100log100) \approx 2(10^2)$
Algorithm N	$O(10^{100}) \approx 10^{100}$	$O(100) \approx 10^2$

b) Identify the Algorithms with their letter L, M, N on the curves shown below when the **time complexity** is graphed against the input size n. (1 mark)



Question 4 (continued)

c) Identify the Algorithms with their letter L, M, N on the curves when the **space complexity** is graphed against the input size n. (1 mark)



d) All of these algorithms L, M, N can be used to solve a particular problem X. What is the time complexity classification of this particular problem X? Justify your response. ***(2 marks)***

- The fastest algorithm to solve problem X is algorithm M which is O(nlogn) – a log/polynomial function
- This indicates that problem X is in class P, since it can be determined in polynomial time

Use the Master Theorem on the recurrence relations below that represent time complexity of various algorithms to find the equivalent function time complexity function $T(f(n))$.

e) $T(n) = 4T\left(\frac{n}{2}\right) + n^2,$ T(1)=O(1) (1 mark)

$a = 4, b = 2, k = 2 \quad \frac{a}{b^k} = \frac{4}{4} = 1 ==> O(n^2 \log_2 n)$

f) $T(n) = T\left(\frac{n}{3}\right) + \sqrt{n},$ T(1)=O(1) (1 mark)

$a = 1, b = 3, k = 0.5 \quad \frac{a}{b^k} = \frac{1}{\sqrt{3}} < 1 ==> O(n^{0.5}) = O(\sqrt{n})$

Question 5 (7 marks)

A Transport company will use a plane to deliver the most valuable set of items possible to a remote location in one trip without exceeding the plane's loading capacity.

There are n items that can be selected

item	1	2	3	………..	i		n
weight	w1	w2	w3		wi		wn
value	v1	v2	v3		vi		vn

a) What is the classification of this type of problem using conventions of Computer Science? What is the implication for solvability of this problem? Justify your responses. (2 marks)

- This problem is a NP-Hard problem, not in P, so according to Cobham's thesis is intractable.
- This is because of the combinatorial explosion of choices and the exponential time complexity of deciding how to fill the plane, this problem is essentially the same as the optimal 01 knapsack problem.

The problem of plane transportation is changed so that the company needs to make sure that the value of the items transported in one plane trip while staying within the plane's load capacity is at least $100 000. A cargo value of at least $100 000 covers all the costs and gives a very good profit.

b) How does this change the re-classification of the problem? Explain your reasoning. (2 marks)

- This problem is an NP-Complete problem, because the solution should it be found can now be verified in polynomial time.
- This is because of the combinatorial explosion of choices and the exponential time complexity of deciding how to fill the plane and this problem is essentially the same as the decision 01 knapsack problem.

c) Why are heuristics used in computer science to solve problems? Give an example of a problem that uses heuristic algorithms. What features of that problem indicate a heuristic solution needs to be used?

(3 marks)

- Heuristics are used when it is not possible to find an exact solution in a reasonable or feasible time for a problem.
- uses heuristic algorithms is the 01 knapsack problem as there is a combinatorial explosion in the decisions that can be made for the best combination of items that fit in the limited capacity knapsack to optimise the value.
- The indicators for using heuristics are that the 01 knapsack problem is in class NP-complete as it has a quickest correct algorithm of exponential time complexity.

Question 6 (5 Marks)
Catalan Numbers are a special kind of number that is generated by the following expression:

$$C_0 = 1 \quad \text{and} \quad C_{n+1} = \sum_{i=0}^{n} C_i\, C_{n-i} \quad \text{for } n \ge 0;$$

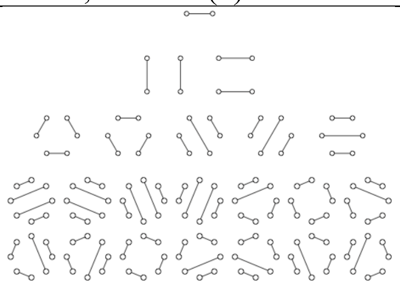
Alternatively expressed as

$$C_0=1, C_{n+1}=C_0C_n+C_1C_{n-1}+\cdots+C_kC_{n-k}+\cdots+C_nC_0$$

The first Catalan numbers for $n = 0, 1, 2, 3, 4, 5$ are 1, 1, 2, 5, 14, 42

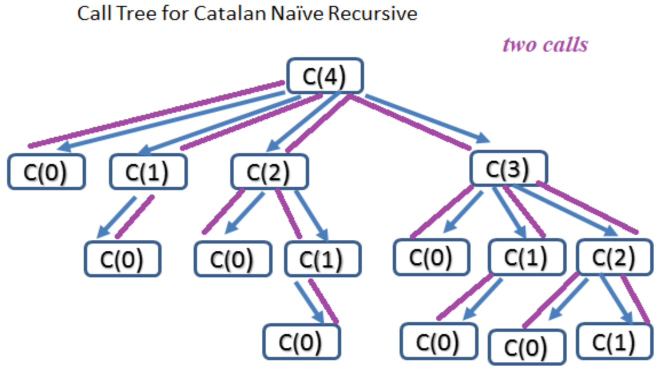
A Naïve algorithm **"C"** for calculating Catalan numbers is shown below, where C(0) = 1:

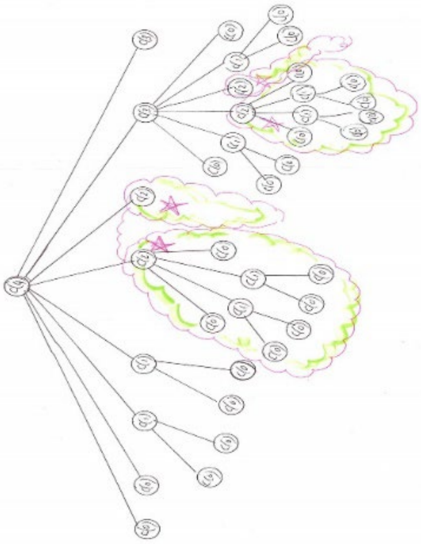
```
Function C(input integer: n)
    If (n==0) then
        return 1
    End if
    sum:=0
    For i=1 to n do
        sum:=sum+C(i-1)*C(n-i)
    End do
    return sum
End Function
```



Catalan numbers count the total number of non-crossing perfect matchings in the complete graph K_{2n}
http://rosalind.info/glossary/catalan-numbers/

a) Show a call graph/tree for the instance when the algorithm is called **C(4)** (2 marks)





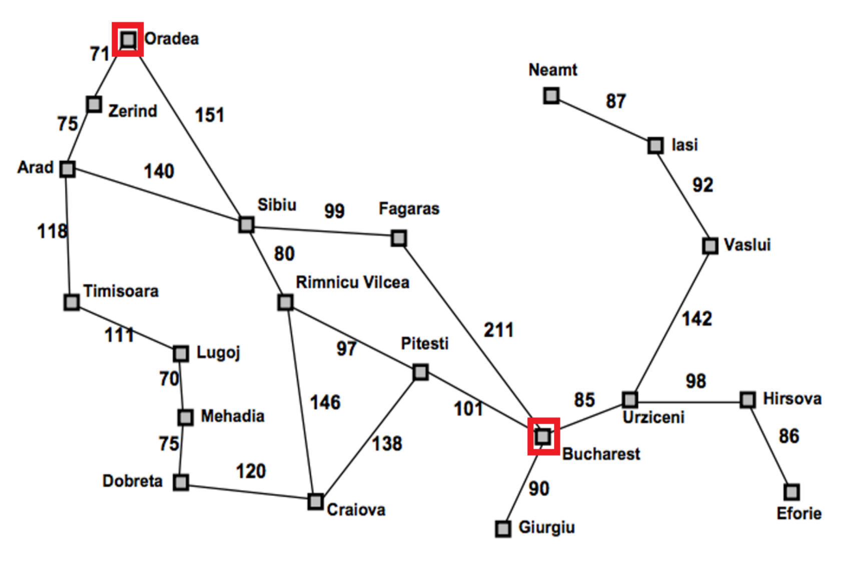
b) Hence using the call tree, what kind of time complexity do you estimate for Function C? (1 mark)

By sketching a call tree of the recursive calls you can see the exponential growth of the recursive calls.

c) Can the Master Theorem be used to calculate the time complexity of this algorithm? Explain your answer (2 marks).

- The Master theorem cannot be applied since there is a non-constant amount of recursive calls which is n, and
- the input to the recursive calls is decreased by 1 for each of the n recursive calls, the input data is not divided for the recursion

Question 7 (9 marks)



Consider the following road map of a small region of Romania showing the distance in kilometres between towns/cities.

Dorina who is a Computer Scientist lives in Oradea and wishes to visit her friend Anton who lives in Bucharest.

a) Describe a model and a "Best First Search" approach that could be used by Dorina to find the best route from the town of "Oradea" to the city of "Bucharest". (4 marks)

1. Create a model of nodes representing all the cities with (x,y) attributes to describe the location of the city in the real world and use weighted edges (roads) showing kilometres between the nodes by road travel.
2. Calculate the straight line direction between Oradea (x,y) to Bucharest(x,y) as the heuristic.
3. Use the best matching edge (road) in the closes orientation and direction from Oradea toward Bucharest and move along that edge to the next node.
4. Rank all roads leaving current position (node) by the heuristic calculation of incident roads (edges) that are closest in orientation and distance to the direct line and go to the next node (city) that best matches the direct line to the destination of the node Bucharest from the current position in the path, repeat step 4 until Bucharest is reached.

b) What are the principles of the best first heuristic algorithm? How does it compare to Greedy Algorithm strategies? How does it find a "good" solution for a problem. (2 marks)

- Best-first search is guided by a greedy heuristic function that determines the next best choice from the current state, it is similar to other Greedy algorithms.
- The heuristic function is based on some prior knowledge of the problem and attempts to predict how close the solution is from the current position in the journey towards the solution, paths which are judged to be closer to the solution are ranked better and are extended first.

c) How does the Best First search approach compare with the A* search method? Describe A* search and compare and identify any similarities and differences in the two approaches. (3 marks)

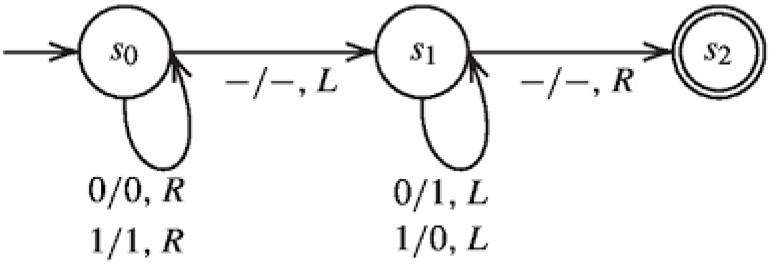
- Like Best First Search the A* search method is a greedy method that also uses a minimum priority queue and heuristic value h(n) to rank the node n for expansion, h(n) tries to determine the best path from the current node n to the target node which is similar to the Best First Search approach.
- The node ranking function used by A* for ranking and expanding node n is more complex, it uses f(n)=g(n)+h(n) and is different to Best First Search which just uses h(n)
- A* combines Best First search with Dijkstra's algorithm, where g(n) is the shortest distance from the source node to node n as determined by Dijkstra's shortest path from the source and h(n) is the heuristic value determined from node n to the target node.

Question 8 (10 marks)

a) Describe the main components of a Turing machine and the functions they perform. (2 marks)

- Unlimited tape with symbols on it, an input/output device to read/write to tape,
- a transition table that defines actions based on current state and input from tape.

Consider the following state diagram for a Turing Machine, where "b" indicates a blank:



TM LEGEND

States: s0, s1, s2

Edge Transitions:

Input Symbol/Output Symbol/Move

Symbols: 0,1,-

Move: L (left), R (right)

b) This 3 state Turing machine is run on the tape containing the sequence -101- and is starting at the third element 0 when in state **s0**. Show the Turing Machine Execution progression until it halts. (2 marks)

State	Tape Contents Sequence
s0	-101-
s0	-101-
s0	-101-
s1	-101-
s1	-100-
S1	-110-
S1	-010-
S1	-010-
S2	-010-

c) Translate this TM to a tabular system of rules. (2 marks)

Current State	Input Symbol	Output Symbol	Move	New State
S0	0	0	R	S0
S0	1	1	R	S0
S0	-	-	L	S1
S1	0	1	L	S1
S1	1	0	L	S1
S1	-	-	R	S2

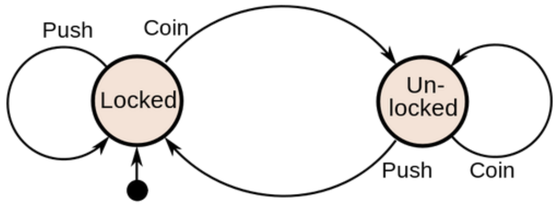
Question 8 (continued)

Finite state machines and Turing machines are models of computation, or hypothetical examples of machines, used as examples to state how much a certain machine/coding language can or needs to do.

A finite state machine (FSM) is a machine that has something called a state, and can receive inputs. The inputs change the state. For example: A turnstile for entry into a building can be modeled as a FSM.

It has two states: either it is locked or unlocked.
If it's locked and it gets a coin, it goes into the unlocked state.
If it's unlocked and it gets a coin, nothing happens, so it stays unlocked.
If it's locked and it gets pushed, nothing happens, so it stays locked.
If it's unlocked and it gets pushed, it goes into the locked state.

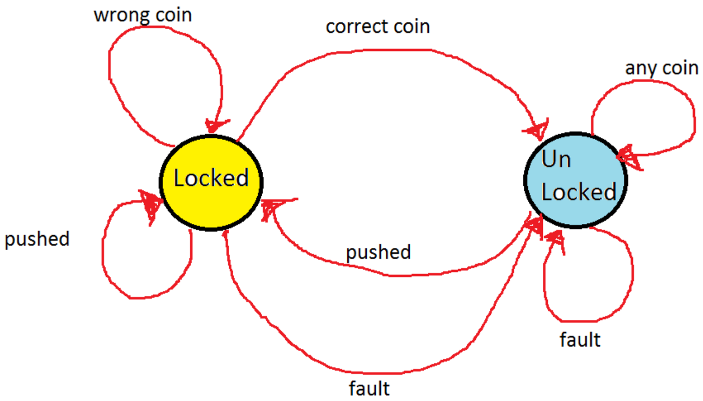
d) Show the information for the Turnstile FSM as a state/transition graph. (2 marks)



A few more cases may occur that need to be considered for the turnstile to operate correctly and need to be included in the operation.

- *The coin inserted must have a certain value for the turnstile to open.*
- *In the case of a fault there needs to be a manual override to open the turnstile.*

e) Create a new state/transition graph depicting the FSM to handle the existing and the extra cases. (2 marks)



Question 9 (7 marks)

a) What are the main features of comparison for Weak AI versus Strong AI? (3 marks)

Weak AI	**Strong AI**
• Limited capabilities: Weak AI is limited to performing specific tasks, such as identifying objects in images, translating languages, or playing chess. • Data-driven: Weak AI systems are trained on large datasets of examples to learn how to perform their tasks. • Human-in-the-loop: Weak AI systems typically require human intervention to define the tasks they are to perform, provide training data, and monitor their performance.	• General intelligence: Strong AI would have the ability to learn and perform any task that a human can. • Self-aware: Strong AI would be conscious of its own existence and able to think about itself. • Autonomous: Strong AI would be able to operate without human intervention.

The Turing test is conducted as follows:

1. A human evaluator engages in a natural language conversation with two other parties, one of which is a human and the other of which is a machine designed to generate human-like responses.
2. All participants are separated from one another. The evaluator is allowed to ask the other two parties any questions they wish, and the other two parties are allowed to respond to the evaluator's questions in any way they choose.
3. If the evaluator cannot reliably tell the machine from the human, the machine is said to have passed the Turing test.

b) How is the Turing test is related and compared to weak AI and strong AI? (2 marks)

- Weak AI: The Turing test can be used to evaluate the performance of weak AI systems. For example, a chatbot that can pass the Turing test is said to have achieved weak AI.
- Strong AI: The Turing test is not a definitive test for strong AI. This is because a machine could pass the Turing test without actually being intelligent. For example, a machine could be programmed to generate responses that are statistically indistinguishable from human responses, even if the machine does not understand the meaning of the responses it is generating.

c) How is Searles' Chinese Room Thought experiment related and compared to weak AI and strong AI? (2 marks)

- Weak AI: The Chinese room thought experiment suggests that it is possible to create weak AI systems that can generate human-like responses to questions without actually understanding the meaning of the responses.
- Strong AI: The thought experiment suggests that strong AI would require more than just the ability to manipulate symbols according to a set of rules. It would also require the ability to have a mind and to understand the meaning of the symbols it is manipulating.

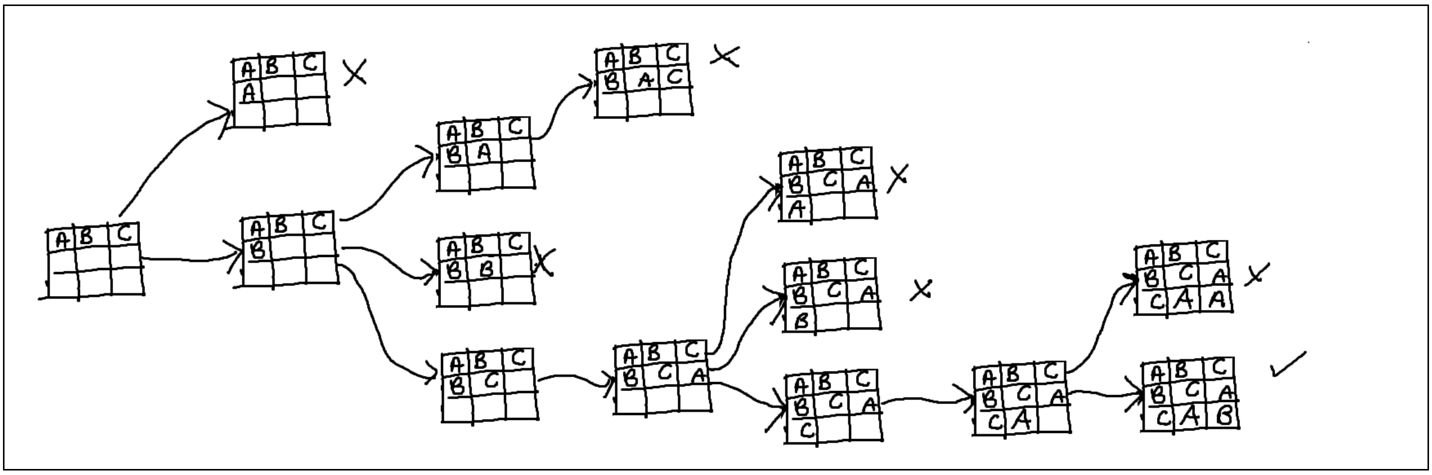
Question 10 (8 marks)

A Latin square is an *n* × *n* array filled with *n* different symbols, each occurring exactly once in each row and exactly once in each column. An example of a 3x3 Latin square using the symbols A, B, C is shown at the right.

A	B	C
B	C	A
C	A	B

<table>
<tr><th>Initial State</th><th>Goal State (s)</th></tr>
<tr>
<td><table><tr><td>A</td><td>B</td><td>C</td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table></td>
<td><table><tr><td>A</td><td>B</td><td>C</td></tr><tr><td>B</td><td>C</td><td>A</td></tr><tr><td>C</td><td>A</td><td>B</td></tr></table>
<table><tr><td>A</td><td>B</td><td>C</td></tr><tr><td>C</td><td>A</td><td>B</td></tr><tr><td>B</td><td>C</td><td>A</td></tr></table></td>
</tr>
</table>

a) Show the decision tree that would arise from the root node or initial state of the Latin square if the backtracking method of progressing toward the goal state(s) was being used. (2 marks)



b) What is the time complexity of the backtracking design pattern for solving this problem? What are the advantages and disadvantages as it compare with the Brute Force approach? Justify your responses. (2 marks)

- With 3 symbols possible for each cell the backtracking approach could be as bad as the brute force approach for n cells which is O(3^n).
- In the worst case time complexity the backtracking approach will be considered equivalent to the brute force approach.
- In the average and best cases of time complexity the backtracking approach should with intelligent conditional statements and logic quickly identify and prune dead end paths and cease exploring there.

c) Write a backtracking algorithm to solve the 3 x 3 Latin Square problem with symbols A, B, C.

(4 marks)

```
Algorithm TryValues(array):
    Letters := [“a”,”b”,”c”]
    fillSoFar = 0
    If CheckLatinOK(array, fillSoFar) AND fillSoFar == 9 then
        Return array // solved
    Else if (CheckLatinOK(array, fillSoFar) == True) then
        TryValues(array) // must have a recursive call
    End if
End Algorithm

Function CheckLatinOK(array, fillSoFar)
    For each row in array do
        for each item in row do
            if item is empty then
                For letter in letters do
                    Set item to letter
                    If (count item in row) = 1 and (count item in column) = 1 then
                        fillSoFar = fillSoFar + 1
                        return true // no conflicts
                    else
                        return false // conflict found
                    End if
                End do
            End do
    End do
end function
```

END OF TRIAL EXAM 1