NAME:

ALGORITHMICS UNIT 3 & 4

Trial Exam 1: 2020

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 |
| В | 7 | 7 | 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 22 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.

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

$$T(n) = \begin{cases} aT\left(\frac{n}{b}\right) + kn^c & \text{if } n > 1\\ d & \text{if } n = 1 \end{cases}$$
 where $a > 0, b > 1, c \ge 0, d \ge 0, k > 0$
and its solution
$$T(n) = \begin{cases} O(n^c) & \text{if } \log_b \ a < c\\ O(n^c \log n) & \text{if } \log_b \ a = c\\ O(n^{\log_b a}) & \text{if } \log_b \ a > c \end{cases}$$

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.

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

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

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

$$T(n) = aT\left(\frac{n}{b}\right) + f(n^k)$$
• $\frac{a}{b^k} < 1$ then $O(n^k)$
• $\frac{a}{b^k} = 1$ then $O(n^k \log_b n)$
• $\frac{a}{b^k} > 1$ then $O(n^{\log_b a})$

•
$$\frac{a}{n^k} < 1$$
 then $O(n^k)$

•
$$\frac{a}{n^k} = 1$$
 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

If the actions are completed in sequence for the abstract data type **MQ** as shown below:

- 1. Create minimum priority queue MQ
- 2. Enqueue into MQ item="Blue", rank=5
- 3. Enqueue into MQ item="Red", rank=3
- 4. Enqueue into MQ item="Orange", rank=2
- 5. Enqueue into MQ Item="Green", rank=4

The contents of **MQ** after these actions could be:

- A. Blue, Red, Orange, Green
- B. Orange, Red, Green, Blue
- C. Blue, Green, Orange, Red
- D. Blue, Orange, Red, Green

Question 2

Two graphs, labelled Graph 1 and Graph 2, are shown below.

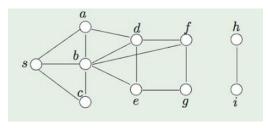


Graph 1 Graph 2

The sum of the degrees of the vertices of Graph 1 is:

- **A.** More than the sum of the degrees of the vertices of Graph 2
- **B.** Two less than the sum of the degrees of the vertices of Graph 2
- C. One less than the sum of the degrees of the vertices of Graph 2
- **D.** Equal to the sum of the degrees of the vertices of Graph 2

Question 3



A Breadth First Search node traversal order starting at **node** "s" for the graph shown above could be:

- **A.** s, c, a, b, e, d, g, f
- **B.** s, a, d, f, g, e, b, c
- **C.** s, a, b, c, d, e, f, g, h, i
- **D.** s, b, e, c, a, d, f, g

Consider the **Algorithm Check** shown in pseudocode below to answer **Question 4** and **Question 5**:

```
Algorithm Check (Items, Key)

// Input: Items, a list of items

// Input: Key a variable

If (there are no items in Items) then
    Return False

Else
    If (the 1st item of Items is equal to Key) then
        Return True
    Else
        Check(all but the first item of Items, Key)
    End if
End Algorithm
```

Question 4

Which is the equivalent iterative algorithm for Algorithm Check in pseudocode?

```
Algorithm A (Items, Key)
                                                       Algorithm B (Items, Key)
    KeyFound:=FALSE
                                                       For i=1 to length of Items do
    For i=1 to length of Items do
                                                            If (ith item of Items equals Key)
        If (ith item of Items equals Key) then
                                                       then
            KeyFound:=TRUE
                                                                Return i
        End if
                                                            End if
    End do
                                                       End do
                                                       End Algorithm
    Return KeyFound
    End Algorithm
    Algorithm C (Items, Key)
                                                       Algorithm D (Items, Key)
C.
                                                   D.
                                                       Repeat until (first item of Items equals
    KeyFound:= FALSE
                                                       Key) do
    While (there are items in the List) AND
                                                            If (first item of list equals Key)
    NOT(KeyFound) do
                                                                Return first item of list
        If (first item of List equals Key) then
             KeyFound:= TRUE
                                                               Remove first item of Items
        Else
                                                            End if
                                                       End Repeat
            Remove first item of Items
                                                       End Algorithm
        End if
    End do
    Return KeyFound
    End Algorithm
```

Question 5

The design pattern of **Algorithm Check** is:

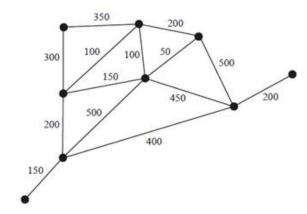
- A. Greedy Recursion
- B. Transform and Conquer
- C. Divide and Conquer
- **D.** Decrease and Conquer

Which of the following statements is true?

- **A.** A greedy algorithm can be used to solve the 0-1 knapsack optimization problem.
- **B.** Dynamic programming can be used to solve optimization problems where the size of the space of possible solutions is exponentially large.
- **C.** Dynamic programming can be used to find an approximate solution to an optimization problem, but cannot be used to find a solution that is guaranteed to be optimal.
- **D.** Decision trees are always binary.

Question 7

A theme park has nine rides and the position of the nine rides are shown in the graph below.



The numbers on the edges represent the distances in metres between the rides. Electrical cables are required to power the rides. These cables will form a connected graph. The shortest total length of cable that can be used to connect all the rides is:

- **A.** 1500 metres
- **B.** 1550 metres
- **C.** 1250 metres
- **D.** 1350 metres

Question 8

The correct ascending order of the time complexities shown in each option below is:

| Α. | $O(n^2)$ | В. | O(1) |
|----|------------|----|----------|
| | $O(2^{n})$ | | O(nlogn) |
| | O(n!) | | O(logn) |
| | O(logn) | | O(n) |
| | O(n) | | $O(n^2)$ |
| C. | 0(1) | D. | (logn) |
| | O(nlogn) | | O(nlogn) |
| | O(n) | | O(n) |
| | $O(n^2)$ | | $O(n^2)$ |
| | $O(2^{n})$ | | O(n!) |

Consider the Algorithm Mystery defined in pseudocode.

```
Algorithm Mystery (A, L, H, K)

If (H ≥ L) then

M:=L + round((H-L)/2)

If A[M]=K then

Return M

Else

If (K < A[M]) then

Mystery(A,L,M-1,K)

Else

Mystery(A,M+1,H,K)

End if

End if

Else

Return -1

End if

End Algorithm
```

Use the Algorithm Mystery shown above to answer Question 9 Question 10 and Question 11.

Question 9

The Algorithm Mystery can most closely be described as having the algorithmic design pattern of:

- A. Brute Force
- **B.** Greedy
- C. Divide and Conquer
- **D.** Backtracking

Question 10

The time complexity of the Algorithm Mystery is closest to:

- A. O(nlogn)
- **B.** $O(n^2)$
- **C.** $O(2^n)$
- \mathbf{D} . $O(\log n)$

Question 11

If the Algorithm Mystery is called as follows:

```
Create Array Hue
Hue[1]=128
Hue[2]=178
Hue[3]=209
Hue[4]=221
Mystery (Hue, 1, 4, 221)
```

The **next** recursive call to Algorithm Mystery will be:

```
A. Mystery (Hue, 2, 4, 221)
```

B. Mystery (Hue, 4, 4, 221)

C. Mystery (Hue, 3, 4, 221)

D. Mystery (Hue, 4, 4, 209)

```
name Digraph;
import node, edge, set;
ops newDigraph : →Digraph;
insertNode : Digraph × node → Digraph;
removeNode : Digraph × edge →Digraph;
insertEdge : Digraph × edge →Digraph;
removeEdge : Digraph × Digraph;
startNode : edge→node;
endNode : edge→node;
incident : Digraph×node→set;
```

The missing terms to complete the Abstract Data Type signature shown above are:

- A. Node, node
- B. Digraph, node
- C. Node, edge
- D. Digraph, edge

Question 12

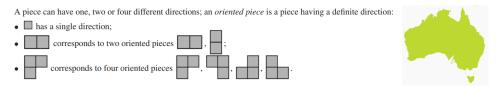
A big-O estimate for the number of operations, where an operation is an addition or a multiplication, used in this segment of an algorithm (ignoring comparisons used to test the conditions in the **while** loop).

```
i := 1
t := 0
while i \le n
t := t + i
i := 2i
```

- A. O(logn)
- \mathbf{B} . O(n)
- \mathbf{C} . $O(n^2)$
- **D.** O(nlogn)

Question 13

A Tetris-like game has regular shaped pieces that can be fitted closely (tessellated) together in different orientations to form a flat surface.



The problem of finding a way of tessellating a large irregular flat area such as the map of Australia with regular shaped tetris pieces so that there are **minimal** gaps is classified as:

- A. NP
- **B.** P
- C. NP-Complete
- D. NP-Hard

The PageRank Algorithm is run on a directed graph with four nodes A, B, C and D as shown below. After the initialize step runs, each page has an initial rank of 0.25:



If d=0.85 which is the probability of links being followed, then on the next iteration the rank of page B will be found by the following mathematical expression:

A.
$$d\left(\frac{0.25}{3}\right) + \left(\frac{1-d}{4}\right)$$

B.
$$d\left(\frac{0.25}{1}\right) + d\left(\frac{0.25}{1}\right) + d\left(\frac{0.25}{4}\right) + \left(\frac{1-d}{4}\right)$$

C.
$$d\left(\frac{0.25}{3}\right) + d\left(\frac{0.25}{4}\right) + \left(\frac{1-d}{4}\right)$$

D.
$$d\left(\frac{0.25}{4}\right) + \left(\frac{1-d}{4}\right)$$

Question 15

Which of the following heuristics uses randomised approaches to solve hard problems?

- A. Hill Climbing
- B. Simulated Annealing
- C. Nearest Neighbour
- **D.** Generate and Test

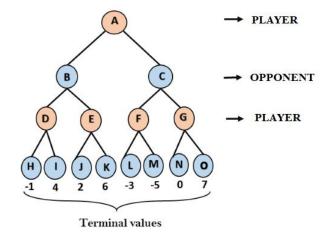
Question 16

If the fastest known algorithm to solve a problem X can run in $O(2^n)$ time, then the problem X is classified as:

- **A.** P class problem
- **B.** Not enough information given.
- C. NP Complete problem
- **D.** NP problem

A game tree of several levels is shown where the PLAYER and OPPONENT levels are indicated.

The values that will be calculated by the Minimax Algorithm for nodes A, B and C will be:



- A. Node A=4, Node B=4, Node C=-3
- B. Node A=0, Node B=2, Node C=0
- C. Node A=7, Node B=6, Node C=7
- D. Node A=4, Node B=4, Node C=0

Question 18

Which of the following is **not** an Intractable problem?

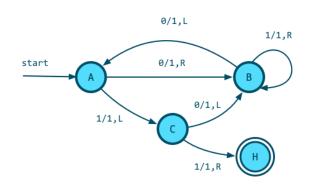
- **A.** The decision version of the Travelling Salesman problem in a weighted connected graph.
- **B.** Finding the shortest path between two locations in a weighted connected graph.
- **C.** Finding the cheapest Hamiltonian path in a weighted connected graph.
- **D.** Finding the maximum path length between two locations in a weighted connected graph.

Question 19

The recurrence relation $T(n) = 3T(\frac{n}{3}) + \sqrt{n}$, where T(1) = O(1) for a recursive algorithm, can be expressed by the following equivalent function:

- $\mathbf{A.} \ \mathrm{O}(\mathrm{n}^3)$
- **B.** $O(\sqrt{n})$
- **C.** O(n)
- **D.** $O(3^n)$

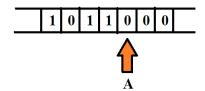
A Turing Machine (TM) of four states A, B, C and H is defined as follows:



TM Symbols: {0,1} TM Movement:

- L=Left one cell
- R=Right one cell

TM Edge transitions <Input Symbol>/<Output Symbol>, <Movement>



The **next four** actions for this TM for the memory tape shown above with the current position and state indicated will be:

- A. 0/1,L => State A, 1/1,L => State C, 1/1,R => State H, no action
- **B.** 0/1,R => State B, 1/1,L => State B, 1/1,L => State B, 0/1,R => State A
- C. 0/1,R => State C, 0/1,L => State B, 1/1,L => State B, 1/1,R => State B
- **D.** 0/1,R => State B, 0/1,L => State A, 1/1,L => State C, 1/1,R => State H

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

Question 1 (15 marks)

a) Complete the missing actions in pseudocode of the following Quicksort Algorithm and the partition module shown below when it is called as quicksort(A, 0, length(A)-1). (3 marks)

| algorithm quicksort(A, low, high) | module partition(A, low, high) |
|--|---|
| // input A: an array zero referenced | // input A: an array zero referenced |
| // input low: an integer index into the Array | // input low: an integer index into the Array |
| // input high: an integer index into the Array | // input high: an integer index into the Array |
| if low < high then | pivot := A[high] |
| p := partition(A, low, high) | i := low |
| quicksort() | for j := low to high do |
| | if A[j] < pivot then |
| quicksort() | swap |
| end if | i := i + 1 |
| end algorithm | end if |
| | end do |
| | swap A[i] with A[high] |
| | return i |
| | end module |
| b) What are the main features and design pattern of the | ne Quicksort algorithm? (2 marks) |
| b) What are the main features and design pattern of the | ne Quicksort algorithm? (2 marks) |
| Use the array of unsorted numeric values [9, 7, 5, 1 c) If we run Quicksort using the rightmost element as | 1, 12, 2, 14, 3, 10, 6] to answer part c) |
| Use the array of unsorted numeric values [9, 7, 5, 1 c) If we run Quicksort using the rightmost element as | 1, 12, 2, 14, 3, 10, 6] to answer part c) s the pivot value, show the partitioning steps that |
| Use the array of unsorted numeric values [9, 7, 5, 1 c) If we run Quicksort using the rightmost element as | 1, 12, 2, 14, 3, 10, 6] to answer part c) s the pivot value, show the partitioning steps that |
| Use the array of unsorted numeric values [9, 7, 5, 1 c) If we run Quicksort using the rightmost element as | 1, 12, 2, 14, 3, 10, 6] to answer part c) s the pivot value, show the partitioning steps that |

Question 1 (continued)

d) If we run Quicksort using the rightmost element as the pivot value, what could be the best case and worst case time complexity for execution? Justify your analysis using recurrences or other mathematical constructions and provide examples to justify your conclusions. (4 marks)

| | Quicksort Worst case | Quicksort Best case | | | |
|----|---|--|--|--|--|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| e) | What strategies could be used for Quicksort to avoi | d the worst case time complexity? (1 mark) | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| f) | | ared to Mergesort? Compare the time complexity and | | | |
| | space complexity of the two sorting algorithms. | (2 marks) | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Question 2 (12 marks)

| a) | Describe the 01 Knapsack problem, and the two main versions of it. | (2 marks) |
|----|--|-----------|
| | | |
| | | |
| b) | What complexity class do the two main versions of 01 Knapsack belong to and why? | (3 marks) |
| | | |
| | | |
| | | |
| c) | What are real life applications of each of these two versions of the Knapsack problem? | (2 mark) |
| | | |
| d) | Outline a brute force approach for finding a solution, outlining the benefits and limitations. | (2 marks) |
| | | |
| | | |
| e) | What other kind of algorithm(s) or algorithmic approaches can be used to find solutions for to problems? Describe in detail two algorithmic strategies for finding a solution, outlining the blimitations. | |
| | | |
| | | |
| | | |
| | | |

Question 3 (13 marks)

Consider the following board puzzle.

| Complact the for | | | | | | | |
|------------------|---|---|---|---|--|--|--|
| | 1 | 2 | 3 | 4 | | | |
| \boldsymbol{a} | S | | 0 | | | | |
| b | | | | 1 | | | |
| \boldsymbol{c} | 施 | 0 | | | | | |
| d | | | 0 | E | | | |

S=a1 E=d4

Circle={a3, c2, d3}

Black= $\{b4,c1\}$

In this board puzzle there are five types of cells; white, black, circle, S and E. The task is to find a way how to move from the point S (Start) to the point E (End) using the smallest number of steps as possible keeping to the following rules:

- You can only move horizontally right or vertically down on the board
- If you are on a white cell you can move only 1 cell
- If you are on an S you can move 1 cells
- If you are on a circle, you can move 1, 2 or 3 cells
- You cannot enter nor go through black cells.

| this puzzle be modelled as using a graph Abstract Data Type. | (4 marks |
|--|----------|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
|) What are the main attributes of the model you have created in part a)? | (1 marks |
| | |

Question 3 (Continued)

| | 1 | 2 | 3 | 4 |
|------------------|-----|---|---|---|
| a | S | | 0 | |
| b | | | | 1 |
| \boldsymbol{c} | 100 | 0 | | |
| d | | | 0 | E |

c) For each of the following graph algorithms complete the following table with the main Abstract Data Type (ADT) used by the algorithm in controlling the information of the graph $G=\{V,E\}$, and its time complexity for a graph $G=\{V,E\}$ in terms of |V| and |E|, and the main actions done. (5 marks)

| Algorithm for | Main ADT used | Time | Main action(s) of algorithm |
|----------------|---------------|---------------|---|
| $G=\{V,E\}$ | | complexity in | (,, , , , , , , , , , , , , , , , , , , |
| , , | | terms of V | |
| | | and E | |
| Dijkstra's | | | |
| Shortest Path | | | |
| | | | |
| | | | |
| Bellman-Ford | | | |
| Shortest Path | | | |
| | | | |
| | | | |
| Floyd Warshall | | | |
| Shortest Path | | | |
| | | | |
| | | | |
| Breadth First | | | |
| Search | | | |
| | | | |
| | | | |
| Depth First | | | |
| Search | | | |
| | | | |
| | | | |

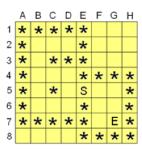
| d) Considering the algorithms shown in the table in part c), which of these algorithms is best to use to find the shortest path from node "S" to node "E" using the graph model from part a)? Justify your choice of algorithm based on the graph attributes that you have identified from part b), and time complexity. (3 marks) |
|---|
| |
| |
| |

Question 4 (10 marks)

Consider the recursive function eMaze defined in pseudocode below for an 8x8 cell matrix representation of a maze.

Blocked cells are filled with an " * " or an " + " symbol.

An example of the input matrix for the function eMaze is shown to the right.



function eMaze(maze[][],x,y)

```
//Input maze: a 2 dimensional matrix, maze[x][y] is a cell in the matrix
//Input x: a column of the maze
//Input y: a row of the maze
if (y>8 OR y<1
  OR x < 'A' OR x > 'H') then
    //position is outside the matrix
    return false
end if
if (maze[x][y] is '*' or '+') then
    print (blocked) //blocked cell
    return false
end if
if (maze[x][y] equals 'E' then
    print (found E) //reached end
    return true
end if
//mark cell x,y as part of path
print (x,y)
If (maze[x][y] is empty) then
    maze[x][y]='+'
end if
if eMaze(maze,x,y+1) then
    return true
end if
if eMaze(maze,x,y-1) then
    return true
end if
if eMaze(maze,x+1,y) then
    return true
end if
if eMaze(maze,x-1,y) then
    return true
end if
//unmark cell x,y as part of path
If (maze[x][y]='+') then
    maze[x][y]=' '
end if
return false
End function
```

Question 4 (continued)

Use the following 8x8 matrix as input for the **function eMaze** to answer the questions.

The starting cell is labelled with an "S" and has the row x = E and the column y = 5.

| | Α | В | C | U | | г | G | Н |
|---|---|---|---|---|---|---|---|---|
| 1 | * | * | * | * | * | | | |
| 2 | * | | | | * | | | |
| 3 | * | | * | * | * | | | |
| 4 | * | | | | * | * | * | * |
| 5 | * | | * | | S | | | * |
| 6 | * | | | | * | | | * |
| 7 | * | * | * | * | * | | Е | * |
| 8 | | | | | * | * | * | * |

| a) | Identify all the base cases for the function eMaze. | (3 mar | ks) |
|----|---|--------|-----|
|----|---|--------|-----|

b) Complete the table below and show the output that will result from the recursive function eMaze in the table below when the **function is called** eMaze(maze, E, 5) (4 marks)

| starting at cell x=E,y=5 | Show the order that | Show the Main Actions trace starting |
|--|---------------------|--------------------------------------|
| | cells are visited | from Cell x=E,y=5 |
| <pre>function eMaze(maze[][],x,y)</pre> | | 7,0 |
| if (y>8 OR y<1 | | |
| OR x <'A' OR x>'H') then | | |
| // outside the matrix | | |
| return false | | |
| end if | | |
| if (maze[x][y] is '*' or '+') then | | |
| print (blocked) //blocked cell | | |
| return false | | |
| end if | | |
| <pre>if (maze[x][y] equals 'E' then</pre> | | |
| print (found E) //reached end | | |
| return true | | |
| end if | | |
| //mark cell x,y as part of path | | |
| print (x,y) | | |
| <pre>If (maze[x][y] is empty) then</pre> | | |
| maze[x][y]='+' | | |
| end if | | |
| <pre>if eMaze(maze,x,y+1) then return true</pre> | | |
| end if | | |
| if eMaze(maze,x,y-1) then | | |
| return true | | |
| end if | | |
| if eMaze(maze,x+1,y) then | | |
| return true | | |
| end if | | |
| if eMaze(maze,x-1,y) then | | |
| return true | | |
| end if | | |
| //unmark cell x,y as part of path | | |
| <pre>If (maze[x][y]='+') then</pre> | | |
| maze[x][y]=' ' | | |
| end if | | |
| return false | | |
| End function | | |

Question 4 (continued)

c) Using the trace from part b) Show the call tree for eMaze(maze, E, 5).

(3 marks)

Extra space provided on this page to answer question 4.

Question 5 (10 marks)



At the **Super Big Rooster** restaurant you can only buy chicken nuggets in packages containing 6, 9 or 20 pieces.

| a) | Write a brute force naive algorithm in structured pseudocode that accepts an integer, N, as an argument and finds all the possible ways that it is or isn't possible to buy N nuggets at the Super Big Rooster restaurant. (5 marks | |
|----|---|---|
| | 10staurant. (5 marks | ソ |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Question 5 (continued)



At the **Super Big Rooster** restaurant you can still only buy chicken nuggets in packages containing 6, 9 or 20 pieces.

For 1 to 10 nuggets the minimum package purchase possibilities are:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|----|
| F | F | F | F | F | 1 | F | F | 1 | F |

For 11 to 20 nuggets the minimum package purchase possibilities are:

| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|----|-------|----|----|-------|----|----|-------|----|------|
| F | 2 | F | F | 2 | F | F | 2 | F | 1 |
| | (6+6) | | | (9+6) | | | (9+9) | | (20) |

For 21 to 30 nuggets the minimum package purchase possibilities are:

| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
|--------|----|----|--------|----|--------|----|----|--------|--------|
| 3 | F | F | 3 | F | 2 | F | F | 2 | 4 |
| (15+6) | | | (15+9) | | (20+6) | | | (20+9) | (24+6) |

| b) | Using an Advanced Algorithmic Pattern such as Divide and Conquer, Backtracking, create an efficient algorithm in structured pseudocode t argument and finds the minimum packages that it is or isn't possible | hat accepts an integer, N, as an |
|----|---|----------------------------------|
| | the Super Big Rooster restaurant. | (5 marks) |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

| Question o (o mains) | O | uestion | 6 | (8 | marks) |) |
|----------------------|---|---------|---|----|--------|---|
|----------------------|---|---------|---|----|--------|---|

| a) Show that for all natural numbers n it is the case that n is divisible by 4 if and only by the last two digits of n is divisible by 4. (Hint: One may write any natural numbers n) | • |
|--|-----------|
| with $k \in N \cup \{0\}$ and $0 \le l \le 99$, so l is the number given by the last two dig Eg. $115 = 100 + 15$, $3921 = 3900 + 21$) | |
| Eg. 113 –100+13, 3921–3900+21) | (4 marks) |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| Consider the following recursive algorithm to compute n! Algorithm Factorial | |
| INPUT: n, a positive integer. OUTPUT: n!. | |
| <pre>if n = 1 then return 1;</pre> | |
| else return n × Factorial(n - 1); fi; | |
| end; | |
| b) Prove the correctness of the given factorial algorithm. | (4 marks) |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

Question 7 (12 marks)

a) Complete the following table of Computer Science terminology with a short description. (6 marks)

| Computer Science terminology | Description of terminology | |
|---|---|----------------------|
| P complexity class | | |
| NP complexity class | | |
| NP-Complete problem | | |
| Intractable problem | | |
| Decidable problem, give an example | | |
| Computable problem, give an example | | |
| | rogram? What were the aims of the program? Explain how Hilbert's programputer Science terminology and give reasons for your answer. | ram can be (3 marks) |
| | | |
| | | |
| | | |
| c) What is the Halting reasons for your ans | problem? What is its classification using Computer Science terminology swer. | and give (3 marks) |
| | | |
| | | |
| | | |
| | | |