Name:
3.2 Tree traversal

Class:

Date:

Time:
239 minutes

Marks:
163 marks

Comments:

## Q1.

Figure 1 shows the data Norbert, Phil, Judith, Mary, Caspar and Tahir entered into a binary search tree.

Figure 2 contains pseudo-code for a recursive binary tree search algorithm.
Figure 1

 (left or right). It returns a Boolean value indicating if the node given as a parameter has a child node in the direction specified by the second parameter. For instance, Exists (Mary, left) will return a value of False as there is no node to the left of Mary in the binary tree.
node.right evaluates to the child node to the right of node, eg Judith.right is Mary.
node. left evaluates to the child node to the left of node, eg Judith. left is Caspar.
(a) What is meant by a recursive subroutine?
$\qquad$
$\qquad$
(b) There are two base cases for the subroutine TreeSearch. State one of the base cases.
(c) Complete the unshaded cells of the table below to show the result of tracing the TreeSearch algorithm shown in Figure 2 with the function call TreeSearch (Olivia, Norbert). You may not need to use all of the rows.

| Function call | Output |
| :--- | :--- |
| TreeSearch (Olivia, <br> Norbert) |  |
|  |  |
|  |  |
|  |  |
|  |  |

Q2.
A binary search tree can be used to represent a list of data so that it can be efficiently searched. Figure 1 shows an example of a binary search tree:

Figure 1

(a) The tree in Figure $\mathbf{1}$ is to be searched for data item "Lisa". The tree does not contain "Lisa".

List the data items that will be examined, in the order that they will be visited, when "Lisa" is searched for.
$\qquad$
(b) Tick one box in the table to indicate the time complexity of the algorithm used to search for data in a binary search tree.

| Time <br> Complexity | Tick one box |
| :---: | :---: |
| $\mathrm{O}(\mathrm{n})$ |  |
| $\mathrm{O}(\log \mathrm{n})$ |  |


| $\mathrm{O}\left(\mathrm{n}^{2}\right)$ |  |
| :--- | :--- |

(c) In Figure 2 below, show how the tree in Figure 1 could be represented by a Start Index, together with a one-dimensional array of records, each of which contains the fields Left Pointer, Data and Right Pointer:

Figure 2

|  | Index <br> [1] | Left Pointer | Data | Right Pointer |
| :---: | :---: | :---: | :---: | :---: |
|  |  |  |  |  |
| Start Index | [2] |  |  |  |
|  | [3] |  |  |  |
|  | [4] |  |  |  |
|  | [5] |  |  |  |
|  | [6] |  |  |  |
|  | [7] |  |  |  |

(d) The array shown in Figure $\mathbf{2}$ is an example of a static data structure.

Explain the differences between a static data structure and a dynamic data structure, and what the heap is used for with a dynamic data structure.
$\qquad$

$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
(e) An in-order traversal is carried out on the binary tree in Figure 1 to output the values stored in the nodes of the tree.

Figure 1

(i) Write out the data items from the tree, in the order that they will be output during the traversal.
$\qquad$
$\qquad$
(ii) What is the significance of the order that the data items have been output in?
$\qquad$
$\qquad$

(f) Graph traversal is a more complex problem than tree traversal. State one feature that a graph might have, which a tree cannot have, that makes graph traversal more complex.


## Q3.

A computer program stores a list of integers in an array named List. The numbers in the array are to be sorted into ascending order so that a particular efficient search algorithm can be used to search for a number.
(a) One of the search algorithms in Table 1 can only be used successfully on a sorted list.

Place one tick next to the name of the algorithm that requires a list to be sorted.
Table 1

| Algorithm Name | Requires Sorted <br> List? <br> (Tick one box) |
| :--- | :--- |
| Binary search |  |

(b) The pseudo-code for a standard algorithm that can be used to sort the data in the array List into order is shown in Figure 1. The variable ListLength stores a count of the number of items in the array List.

Array indexing starts at 1.
Figure 1

```
For OuterPointer \leftarrow 2 To ListLength
    CurrentValue \leftarrow List[OuterPointer]
    InnerPointer \leftarrow OuterPointer - 1
    While InnerPointer > O And
                List[InnerPointer] > CurrentValue Do
        List[InnerPointer + 1] \leftarrow List[InnerPointer]
        InnerPointer \leftarrow InnerPointer - 1
    EndWhile
    List[InnerPointer + 1] \leftarrow CurrentValue
EndFor
```

Complete the empty (unshaded) cells in the trace table (Table 2) for an execution of the algorithm in Figure $\mathbf{1}$ when the array List contains the values $9,8,5$ and 6 in that order.

Table 2

(c) In the trace table (Table 2), when the variable OuterPointer contains the value 2
and then 3 , the value of the variable InnerPointer decreases to 0 . When OuterPointer contains 4, InnerPointer stops decreasing when it reaches the number 1.

Explain why InnerPointer does not decrease to 0 when OuterPointer contains 4.
$\qquad$
$\qquad$
$\qquad$
$\qquad$
(d) Tick one box in Table 3 to indicate the correct Order of Time Complexity of the standard algorithm in Figure 1.

Table 3

(e) State the name of the standard algorithm that is represented by the pseudo-code in Figure 1.
(f) Instead of storing a list of numbers in an array as in (b), the numbers could be stored in a binary search tree. This would also enable efficient searching.

The numbers $9,6,1,8,20$ and 10 are put into a binary search tree in that order. Figure 2 shows this binary search tree.

Figure 2

(i) A search of the binary tree is performed for the number 8.

List the numbers, in the order that they would be checked, for the search to determine that the number 8 is present in the tree.
$\qquad$
(ii) A search of the binary tree is performed for the number 11.

List the numbers, in the order that they would be checked, for the search to determine that the number 11 is not present in the tree.
$\qquad$
(g) The numbers 4, 5 and 3 are to be added into the binary search tree, in that order.

Figure 3 below is an identical copy of Figure 2.
Complete Figure $\mathbf{3}$ below to show the binary search tree from Figure $\mathbf{2}$ after the extra numbers have been added into it.

Figure 3


## 

## Q4.

A tree can be used to represent a mathematical expression. This is known as an expression tree. Figure 1 is an expression tree for the infix expression $4+9$ * 6 .

Figure 1

(a) An expression tree is an example of a rooted tree.

State the contents of the root node: $\qquad$
List the contents of all of the leaf nodes: $\qquad$
(b) The expression tree in Figure 1 could be represented using three one-dimensional arrays named A, B and C. Figure 2 shows a representation of Figure $\mathbf{1}$ together with the array indices.

| Figure 2 |
| :--- |
| Arrays |


Describe the role of each of the arrays $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$.
A:

B:

C:
(c) What does an entry of 0 in array B indicate?
$\qquad$
$\qquad$
(d) The procedure in Figure 3 describes a type of tree traversal that can be carried out on the representation of the tree shown in Figure 2.

Figure 3

```
Procedure Traverse(Pos:Integer)
    If B[Pos] > O Then Traverse(B[Pos])
    If C[Pos] > 0 Then Traverse(C[Pos])
    Output A[Pos]
End Procedure
```

Using the table below, trace the execution of the procedure when it is called using Traverse (1). You may not need to use all of the lines provided in the table.

(e) Which type of tree traversal does the procedure Traverse carry out?
$\qquad$
(f) What does the output of the procedure represent?
$\qquad$

## Q5.

A graph can be drawn to represent a maze. In such a graph, each graph vertex represents one of the following:

- the entrance to or exit from the maze
- a place where more than one path can be taken
- a dead end.

Edges connect the vertices according to the paths in the maze.
Diagram 1 shows a maze and Diagram 2 shows one possible representation of this maze.
Position 1 in Diagram 1 corresponds to vertex 1 in Diagram 2 and is the entrance to the maze. Position 7 in Diagram 1 is the exit to the maze and corresponds to vertex 7.
Dead ends have been represented by the symbol $-\quad$ in Diagram 2.
Diagram 3 shows a simplified undirected graph of this maze with dead ends omitted.

## Diagram 1



ENAIVI トAトCRS RACTICE

## Diagram 2



Representation of maze including dead ends

## Diagram 3



Graph representing maze with dead ends omitted
(a) The graph in Diagram 3 is a tree.

State one property of the graph in Diagram 3 that makes it a tree.
$\qquad$
$\qquad$
(b) The graphs of some mazes are not trees.

Describe a feature of a maze that would result in its graph not being a tree.
$\qquad$
$\qquad$
(c) Complete the table below to show how the graph in Diagram 3 would be stored using an adjacency matrix.

(d) (i) What is a recursive routine?
$\qquad$
$\qquad$
(ii) To enable the use of recursion a programming language must provide a stack. Explain what this stack will be used for and why a stack is appropriate.
$\qquad$
$\qquad$
$\qquad$
$\qquad$

Diagram 3 is repeated here so that you can answer Question (e) without having to turn pages.

(e) A recursive routine can be used to perform a depth-first search of the graph that represents the maze to test if there is a route from the entrance (vertex 1 ) to the exit (vertex 7).

The recursive routine in the diagram below is to be used to explore the graph in Diagram 3. It has two parameters, v (the current vertex) and Endv (the exit vertex).

Procedure DFS (V, EndV)
EXAMPAPERS PRACTICE
For each vertex $U$ which is connected to $V$ Do
If Discovered [U] = False Then DFS (U, EndV)
EndFor
CompletelyExplored[V] $\leftarrow ~ T r u e ~$
EndProcedure
Complete the trace table below to show how the Discovered and CompletelyExplored flag arrays and the variable Found are updated by the algorithm when it is called using $\operatorname{DFS}(1,7)$.

The details of each call and the values of the variables $V, \quad u$ and Endv have already been entered into the table for you. The letter $F$ has been used as an abbreviation for False. You should use T as an abbreviation for True.

|  |  |  |  | Discovered |  |  |  |  |  |  | CompletelyExplored |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| Call | V | U | EndV | [1] | [2] | [3] | [4] | [5] | [6] | [7] | [1] | [2] | [3] | [4] | [5] | [6] | [7] | Found |
|  | - | - |  | F | F | F | F | F | F | F | F | F | F | F | F | F | F | F |
| DFS (1,7) | 1 | 2 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(2,7)$ | 2 | 1 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | 3 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS (3,7) | 3 | 2 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(2,7)$ | 2 | 4 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(4,7)$ | 4 | 2 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | 5 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(5,7)$ | 5 | 4 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | 6 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(6,7)$ | 6 | 5 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(5,7)$ | 5 | 7 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(7,7)$ | 7 | 5 | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(5,7)$ | 5 | - | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(4,7)$ | 4 | - | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(2,7)$ | 2 | - | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFS $(1,7)$ | 1 | - | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | (To | 12 | $(5)$ marks) |

The table below shows an adjacency matrix representation of a directed graph (digraph).

|  | 1 | 2 | 3 | 4 | 5 |
| :---: | :---: | :---: | :---: | :---: | :---: |
| 1 | 0 | 1 | 0 | 1 | 0 |
| $F \quad 2$ | 0 | 0 | 1 | 1 | 0 |
| $r 3$ | 0 | 0 | 0 | 0 | 0 |
| O 4 | 0 | 0 | 0 | 0 | 1 |
| T 5 | 0 | 1 | 0 | 0 | 0 |

(a) Complete this unfinished diagram of the directed graph.

(b) Directed graphs can also be represented by an adjacency list.

Explain under what circumstances an adjacency matrix is the most appropriate method to use to represent a directed graph, and under what circumstances an adjacency list is more appropriate.
$\qquad$


(d) Data may be stored as a binary tree.

Show how the following data may be stored as a binary tree for subsequent processing in alphabetic order by drawing the tree. Assume that the first item is the root of the tree and the rest of the data items are inserted into the tree in the order given.

Data items: Jack, Bramble, Snowy, Butter, Squeak, Bear, Pip
(e) A binary tree such as the one created in part (d) could be represented using one array of records or, alternatively, using three one-dimensional arrays.

Describe how the data stored in the array(s) could be structured for one of these two possible methods of representation.
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$


A binary tree has the following functions defined
RootValue ( T ) Returns the contents of the root node of the tree $T$
Leftchild ( T ) Returns the left child of the root node of the tree $T$


A recursively-defined procedure P with a tree as a parameter is defined below.
Procedure $\mathrm{P}(\mathbb{T})$
If RightChild(T) Exists
Then P(RightChild(T))
Output RootValue(T)
If LeftChild(T) Exists
Then P(LeftChild(T))
EndProc
(a) What is meant by a recursively-defined procedure?
$\qquad$
$\qquad$
(b) (i) Complete the table below by dry running the procedure call $\mathrm{P}(\mathrm{T})$ for the tree T given below

(ii) What does the procedure P describe?
$\qquad$

Q8.
A binary search tree is used by software to store and then search for user names on a college network.

The following are the first seven user names to join the tree:
PollardJ, AtkinsP, RogersG, AbbottJ, SearleF, CollinsK, RuddleA
(a) Sketch the tree structure.
(b) The tree is to be searched for various user names.
(i) The task is to search for the user name CollinsK. List in order the nodes

(ii) A second search is done to find the user name RuddleA. How many comparisons does this require?

Q9.
A binary search tree has the following functions defined:
RootValue( $T$ ) Returns the value stored in the root node of the tree $T$ LeftChild( $T$ ) Returns the left child (subtree) of the root node of the tree $T$ RightChild( T ) Returns the right child (subtree) of the root node of the tree T

A recursively-defined procedure P with a tree as a parameter is defined below.

```
Procedure P(T)
    If RightChild(T) exists
        Then P(RightChild(T))
    Output RootValue(T)
    If LeftChild(T) exists
        Then P(LeftChild(T))
EndProc
```

(a) What is meant by a recursively-defined procedure?
$\qquad$
$\qquad$
(b) (i) Complete the table below by dry running the procedure call $\mathrm{P}(\mathrm{T})$ for the tree T given below.

(6)
(ii) What does the procedure P describe?
$\qquad$

Q10.
A tree has the following functions defined:
RootValue( $T$ ) Returns the contents of the root node of the tree $T$
LeftChild(T) Returns the left child of the root node of the tree $T$
RightChild( $T$ ) Returns the right child of the root node of the tree $T$
A recursively-defined procedure P with a tree as a parameter is defined below.

```
Procedure P (T)
    If LeftChild(T) exists
                then P(LeftChild(T))
        Output RootValue(T)
        If RightChild(T) exists
            then P(RightChild(T))
EndProc
```

(a) What is meant by recursively-defined?
$\qquad$
$\qquad$
(b) (i) Complete the table below by dry running the procedure call $\mathrm{P}(\mathrm{T})$ for the tree T given below.


(ii) What does procedure P describe?
$\qquad$

Q11.
A recursively-defined procedure Process, which takes an integer as its single parameter,
is defined below.
(a) What is meant by recursively-defined?
$\qquad$
$\qquad$
(b) Describe how a stack is used in the execution of procedure Process?
$\qquad$
$\qquad$
$\qquad$
$\qquad$
(c) Dry run the procedure call Process(1), using the data in the table below, showing clearly the order the values are printed.

```
Procedure Process (P)
    Print (P)
    If Table[P].Left <> 0
                Then Process (Table[P].Left)
    EndIf
    Print (Table[P].Data)
    If Table[P].Right <> 0
            Then Process (Table[P].Right)
        EndIf
EndProcedure
```



Printed Output:=
$\qquad$
(d) What does procedure Process describe?

## Q12.

A binary search tree is a data structure where items of data are stored such that they can be searched for quickly and easily.

The following data items are to be entered into a binary search tree in the order given:
Louise, Peter, Robert, Christine, Alan, Leslie, Maria
(a) Draw a diagram to show how these values will be stored in the tree.

(c) If Maria is being searched for in this binary tree, list the data items which have to be accessed.
$\qquad$

## Q13.

An algebraic expression is represented in a binary tree as follows.

(a) On the above diagram, circle and label the root of this tree, a branch and a leaf node.
(b) In the spaces below, draw the left sub-tree and the right sub-tree of this tree.
left sub-tree right sub-tree
(c) What is the result if this tree is printed using in-order traversal?

Q14.



For the expression $3+x$ the binary tree stores + at the root, 3 at the left hand node and $x$ at the right hand node. If the nodes of this tree are printed as the tree is traversed, what will be printed when the traversal is
(a) pre-order; $\qquad$

(Total 3 marks)

## Q15.

(a) The series of characters J, F, H, U, S, X, T are to be entered into a binary search tree in the order given. Draw a diagram to show how these values will be stored.
(b) The following data are held in arrays Data, L and R:

L

| $\mathbf{2}$ $\mathbf{0}$ $\mathbf{0}$ $\mathbf{5}$ $\mathbf{0}$ $\mathbf{0}$ $\mathbf{0}$ <br> $\left[\begin{array}{c}{[2]}\end{array}[3]\right.$ $[4]$ $[5]$ $[6]$ $[7]$   |
| :---: |

R

| $\mathbf{4}$ | $\mathbf{3}$ | $\mathbf{0}$ | 6 | 7 | 0 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |

Using the arrays above, dry-run the following pseudo-code by completing the trace table opposite:


Trace Table:

(Total 10 marks)

## Q16.

A binary tree can be used to represent the alphabet in a code. Part of the tree is shown below. Starting at the root of the tree, branch left is a dot and branch right is a dash.

So N has the code: dash dot.
SOS has the code dot dot dot dash dash dash dot dot dot.
(a) Place the missing letters S and O into the correct positions in the diagram.

(2)
(b) What does the following 2 letter code spell: dot dot dash ?

(a) Identify each of the following:
(i) the root node,
(ii) the parent nodes,
$\qquad$
(iii) the leaf (terminal) nodes.
(b) Each node contains a data item. What else must a node contain?
$\qquad$
$\qquad$
$\qquad$
$\qquad$

Q18.
An algebraic expression is represented in a binary tree as follows:

(b) Mark and label the left sub-tree and the right sub-tree of this tree.


A recursively-defined procedure $T$, which takes a tree structure, $\operatorname{tree}(x, y, z)$ as its single parameter, where x is the root, y is the left sub-tree and z is the right sub-tree, is defined below (<> means not equal to).

```
Procedure T (tree(x, y, z))
    If y <> empty
        Then
            PRINT ')'
            T(y)
    EndIf
        PRINT x
        If z <> empty
        Then
            T(z)
            PRINT ')'
        EndIf
    EndProc
```

(c) What is meant by recursively-defined?
$\qquad$
$\qquad$
(d) Explain why a stack is necessary in order to execute procedure T recursively.
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
(e) Dry run the following procedure call

T ( tree( '*', tree (' + ', tree ('A', empty, empty), tree ('B', empty, empty) ), tree ( ${ }^{\prime}-$ ', tree ('C', empty, empty), tree ('D', empty, empty) ) )
)

## -

showing clearly the PRINTed output and the values of the parameter omitted from the table (rows 4,5,6,7) for the seven calls of $T$.

| Call Number | Parameter |
| :---: | :---: |
|  | $\text { tree('*', tree(' }+ \text { ', tree('A',empty,empty), tree('B',empty, empty) ), }$ <br> tree('-', tree('C',empty,empty), tree('D',empty,empty)) |
| 2 | tree(' +', tree('A', empty, empty), tree('B', empty, empty) ) |
| 3 | tree('A',empty, empty) |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |

(f) What tree traversal algorithm does procedure T describe?
$\qquad$
$\qquad$

## E旬 <br> EXAM PAPERS PRACTICE

## Mark schemes

## Q1.

(a) Mark is for AO2 (apply)
-2;
(b) Mark is for AO2 (apply)
[8, 3];
I. missing brackets
I. wrong type of brackets
(c) Marks are for AO2 (apply)

| Calculation | Result |
| :--- | :---: |
| U | $[1,1]$ |
| $\mathrm{V}=[$ [position of hero] - [position of enemy] | $[6,-4] ;$ |
| u.v | $2 ;$ |
| EnemyCanSee | True; |

A. different answers that have been correctly calculated based on an incorrect answer for 5.2
(d) 1 mark for AO1 (knowledge)
heuristic approach employs a method of finding a solution that might not be $=$ the best;

## 1 mark for AO1 (understanding)

algorithm might need to consider visiting less/fewer cells/co-ordinates // algorithm might use knowledge of the domain to cut-down the search space // algorithm might consider visiting certain cells/coordinates first;

## Examiner reports

## Q1.

This was the first A Level exam question on the topic of vectors and most students were well-prepared for this topic though there were clear differences between centres with students from some centres showing no familiarity with this area of the syllabus. A significant number of students found calculations that involved negative numbers more difficult.

The explanations of what a heuristic technique is often showed some understanding but were too vague to be creditworthy.


## EXAM PAPERS PRACTICE

