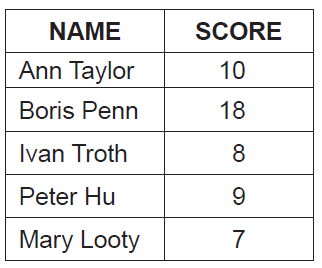
**HL Unit 5** **– Abstract Data Structures**  
Test 1

1. The table below holds student names and scores, from a class test.
2. Draw a diagram to show how the data given in the table could be stored in a binary tree

in the order of scores. Data should be inserted into the binary tree in the order given in

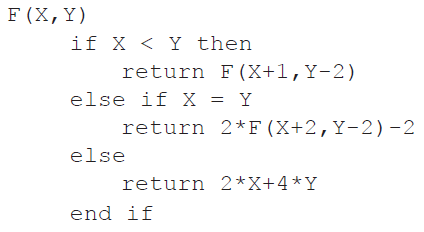
the table (ie data about Ann Taylor is to be inserted first). [3]

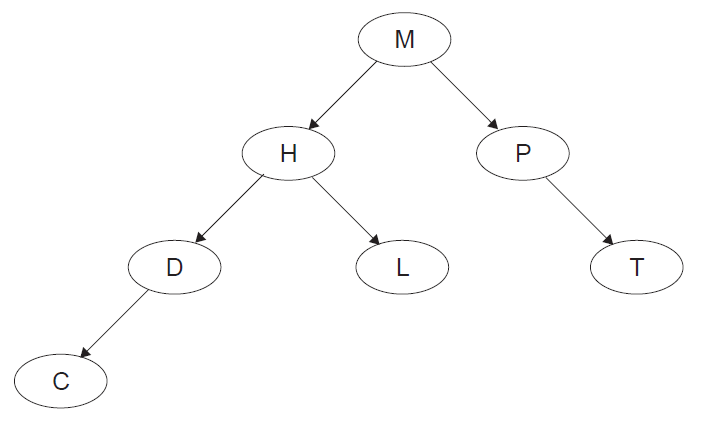
1. The same data could be inserted into a singly linked list in descending order of scores.

Draw a diagram of this singly linked list. [3]

1. Compare the data structures in part (a) and part (b) in terms of:  
     
   (i) searching [2]

(ii) storage requirements. [2]

1. Consider the following **recursive** algorithm, in which X and Y are parameters in the  
   method F. The return statement gives the value that the method generates.  
     
     
     
     
     
     
     
     
     
   Determine the value of F(5,11). [5]

1. Consider the following binary tree.
2. State the order that the nodes will be listed using the postorder tree traversal. [1]
3. The node H is deleted so that the postorder traversal of the **remaining** nodes is

preserved from part (a). Sketch the updated binary tree following this deletion. [2]

1. A laptop computer supplements its primary memory by making use of virtual memory.  
   The laptop has 1GB of random access memory (RAM) and a single processor.  
   The laptop is using one of the latest operating systems to run multimedia gaming programs.

One of the laptop’s game applications stores the data relating to the different actions of one  
of its characters in a stack.

1. Suggest **one** reason why the character’s actions might be stored in a stack. [2]

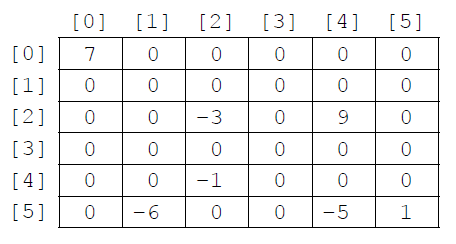
At specific moments during the game this data is read from the stack (S) into a queue (Q).

1. Using appropriate access methods for stacks and queues, construct an algorithm that

reads the data from the stack and enters it into the queue. You should assume that the

queue structure exists and that both structures are of a fixed size. [6]

1. Outline **one** advantage of making the queue dynamic. [2]
2. Consider the following two-dimensional array, MAT, with dimensions 6 × 6.



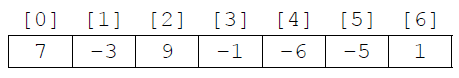
The value −1 is stored in MAT at position [4][2]. The position [4][2] means row 4  
 and column 2.

1. State the total number of elements stored in MAT. [1]
2. State the number of non-zero elements in MAT. [1]

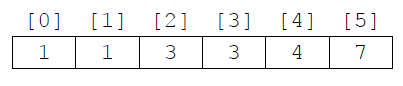
A two-dimensional array in which most of the elements are zero is called a **sparse matrix**. A sparse matrix can be compressed by storing only non-zero elements using three one‑dimensional arrays.

The **first array**, VALUES, stores all non-zero elements taken from the sparse matrix in row‑major order (left-to-right then top-to-bottom order).

The length of the array VALUES is equal to the number of non-zero elements in the sparse matrix. For the sparse matrix above, MAT, the array VALUES is:

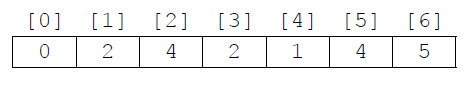


The **second array** is ROWC. ROWC[i] stores the number of non-zero elements, from row 0 to row i of the sparse matrix, **inclusive**.

The length of ROWC is equal to the number of rows in the sparse matrix. For MAT the array ROWC is:

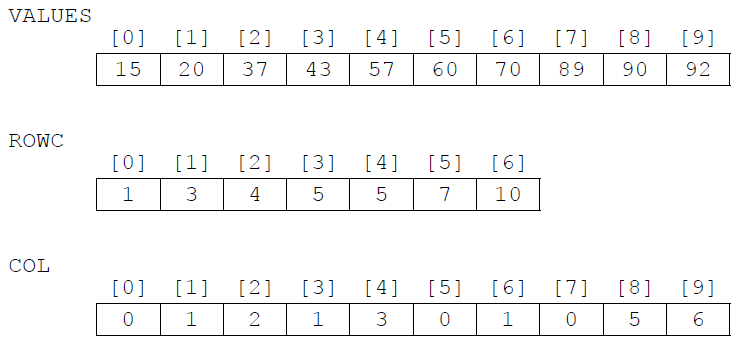
For example, ROWC[2] stores 3 because in MAT there are three non-zero elements from row 0 to row 2, inclusive.

The **third array**, COL, stores the column index for each non-zero element in the sparse matrix. COL[i] stores   
the sparse matrix column index for the non-zero element stored in VALUES[i]. For MAT the array COL is:



1. Construct an algorithm that compresses a 6 × 6 two-dimensional array, such as MAT,   
   into the three one-dimensional arrays described on page 8. You may assume that the   
   6 × 6 array is inputted and all three one-dimensional arrays are initialized. [6]

Consider the following three arrays. They hold the compressed contents of a 7 × 7 sparse   
 matrix, BIGMAT.



1. For a given column, C, in BIGMAT, outline how it could be determined that this column contains no non-zero elements. [2]

1. State how many rows in BIGMAT contain only zeros. [1]
2. (i) State the index in VALUES of the first non-zero element in row 5 of BIGMAT. [1]

(ii) For a given row, R, in BIGMAT, determine the range of indexes in VALUES where  
 non-zero elements in row R of BIGMAT are placed. You may assume that there  
 is at least one non-zero element in row R. [3]