C1

1. In the context of a design process, define the term *iteration* [1]
2. In the context of a networked world, state the role of a *router*. [1]
3. Identify one difference between *alpha test* and *beta test* [2]
4. School library has a book borrow register system including two computers: one at delivery desk and one center computer.

When a student checks a book at the delivery desk, the staff at the desk use a

Keyboard to input the information of the book.

The information is sent to the central computer where the condition of each book and the relative return data are held in a database on a disk.

When the information is found, it is sent to the delivery desk’s computer where all necessary information is stored on a local disk and a receipt is printed out.

1. Construct a system ﬂow chart for the system described above. [5]

At the point of sale there are peripheral devices other than the scanning device and printer.

1. Outline the purpose of one other possible peripheral device in this scenario. [2]
2. Outline the purpose of protocols in transferring this data. [2]

C4

1. Consider the following iterative method, where N is a positive integer

magic(N)

if N mod 2 = 0 then

K = 0

loop while K <= N

output K

K = K + 2

end loop

else

output N

end if

end magic

(a) Determine the output produced by the method call magic(3). [1]

(b) Determine the output produced by the method call magic(4). [3]

(c) Construct an recursion algorithm for the method magic(), which uses a recursion

instead of iteration. [4]

1. Outline the reason why recursive solutions can be memory intensive. [2]
2. Define the term recursion. [1]
3. Greek alphabets were recorded in a stack data structure.

|  |
| --- |
| … |
| theta |
| lambda |
| zeta |
| epsilon |
| delta |
| gamma |
| mu |
| alpha |

The first item stored in the stack was “alpha”.  
Note that “theta” is currently in position 0 in the stack.  
(a) Construct the pseudocode that will search the stack for a specific name, and output its position in the stack. You may assume that all names in the stack are unique. [5]  
(b) Explain the benefits of using a binary search tree, compared to a stack, when  
searching for a specific item. [3]

If the tree is populated with the data from the stack, the first item popped oﬀ will become the root. For each subsequent item popped from the stack, a recursive procedure is followed until the item is correctly placed in the tree.  
(c) Without writing code, describe this recursive procedure. [4]  
(d) By considering only the data visible in the stack shown above, sketch the binary search tree that has been created from the items removed from the stack. [3]

1. The letter letters F0，F1, F2, …, FN, …, where N>=0, are used to identify the Nth term of the sequence of Fibonacci numbers that starts as follow.

0, 1, 1, 2, 3, 5, 8, 13

1. State the value of the 10 term in the sequence.

The following method, fibo(N), generates the Nth term in the sequence. The return statement returns the value that the method generates.

fibo(N)

if (N = 0 or N = 1) then

return N

else

return (fibo(N-1)) + fibo(N-2))

end if

end fibo

1. Trace fibo(4), showing the different levels of recursion
2. Construct a non-recursive algorithm to generate Fibonacci numbers
3. Six students’ numbers are stored in a one-dimension array in the order that they arrived the CS classroom.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 14 | 7 | 5 | 19 | 2 | 28 | 9 | 17 | 1 | 23 |

（1） state the value of array[5] [1]

（2） Construct an algorithm InsertSort to sort the array (smallest in the left end) though using insert sort. [5]

Consider the following situation: after sorted, all student’s numbers are stored in a queue Q instead of array.

（3）Construct an algorithm Inverse to output the sorted array in inverse order using stack.

C5

1. Describe the characteristics of a linked list. [2]
2. Consider the following binary tree:

A

E

B

D

C

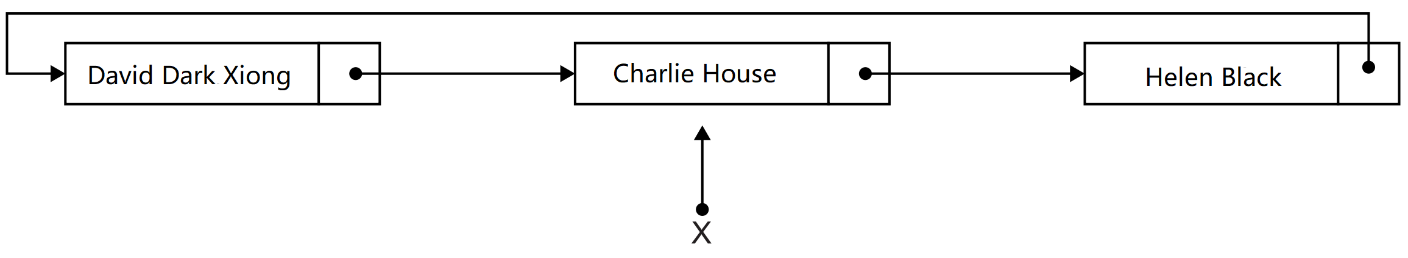
(a) Identify all leaf nodes in this binary root. [1]

(b) For this binary tree, state the result of:

(i) preorder tree traversal, [1]

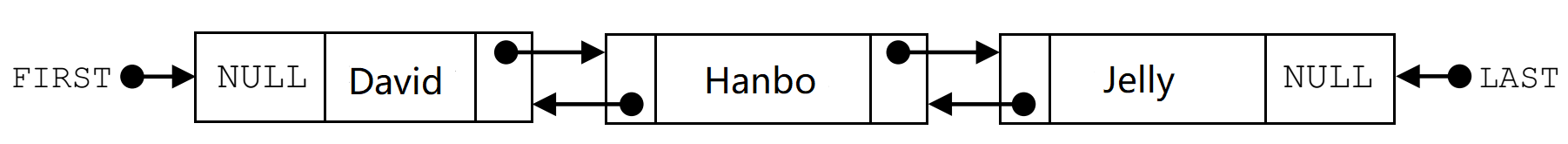
(ii) inorder tree traversal. [1]

1. The diagram shows a list of names held in a circular linked list. The end of the list is pointed to by an external pointer, X.



1. State the first name in this circular list. [1]
2. Describe how the number of names held in this list could be determined. [4]
3. Compare the use of static and dynamic data structures. [3]
4. Describe the features of a dynamic data structure. [2]

Consider the following doubly linked list which holds the names of newbies in alphabetical order.



1. Explain how “XYX” could be inserted into this doubly linked list. You should draw

a labelled diagram in your answer.

**MS:**

**C1**

1. **a procedure in which repetition of a sequence of operations yields results successively closer to a desired result. [1]**
2. **A router is a hardware device that routes data from a local area network (LAN) to another network connection. [1]**
3. **Award up to [2 max].**

**Alpha test is performed at developer's site, Beta test is performed at client location;**

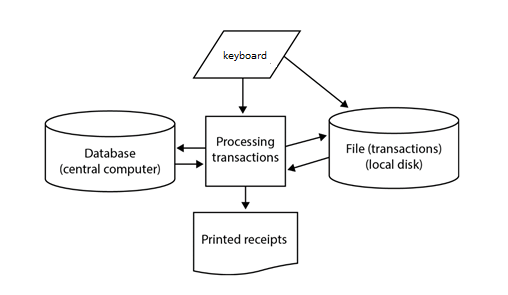
**Alpha test is performed by internal employees, Beta test is performed by external clients or end users;**

**Alpha test is done before the software is made available to general public, Beta test is done before final release.**

**4.**

**(1)**

*Award* ***[1]*** *for all correct labels or symbols.  
(Accept meaningful, consistent symbols.)  
Award* ***[1]*** *for correct input flow.  
Award* ***[1]*** *for correct output flow.  
Award* ***[1]*** *for correct internal processes.  
Award* ***[1]*** *for correct dataflow.  
Answers given as a process flowchart may still be awarded* ***[1]*** *or* ***[2]****.*

****

**(2)** *Award* ***[1]*** *for identifying a peripheral device,* ***[1]*** *for stating its purpose.*

Monitor;  
So the staffs can see the information/data on the screen;  
Visual display;  
So the students can read the information/data on the display;  
Speakers;  
For students to hear information;

**(3)** *Award up to* ***[2 max]****.*Protocols are sets of rules for transmitting data correctly;  
They ensure that data is sent from a customer's computer and received by the  
shop’s computer;  
To create a secure transmission of data from the client to the server through the use  
of the Hypertext Transfer Protocol (HTTPS) *ie* the customer can pay for the books  
securely (using TLS or SSL).

**C4**

**1.**

**(a) 3**

**(b) 0; 2; 4**

**(c)**

*Award marks as follows up to* ***[3 max]****.*

*Award* ***[1]*** *for correct if judgement;  
Award* ***[1]*** *for correct recursive function calling  
Award* ***[1]*** *for correct output*

magic(X)

if (X > 0) AND (X mod 2 = 0) then

magic(X-2)  
end if  
output X

end magic

**2.**

**Award up to [2 max].**

**A recursive call involves the use of stacks;**

**For storing/pushing on/popping out data/ return addresses/return values etc;**

**If many recursive calls are made, the memory usage can be very large; [2]**

**3.**

**Process/method/ procedure/ subroutine/ function/ algorithm that calls itself; [1]**

**4.**

**(1)** *Award* ***[5 max]*** *as follows:  
Example answer 1:*searchName(NAME, STACK)  
 DEPTH = 0  
 NAMEDEPTH = -1  
 loop while not STACK.isEmpty()  
 X = STACK.pop()  
 if X == NAME then  
 NAMEDEPTH=DEPTH

end if

DEPTH = DEPTH + 1

end while  
if NAMEDEPTH == -1 then  
 output(NAME + ’not found’)  
else

output(NAMEDEPTH)  
endif  
end searchName  
*Award* ***[1]*** *for initialization.  
Award* ***[1]*** *for a correct while/until loop.  
Award* ***[1]*** *for correct comparison and assignment.  
Award* ***[1]*** *for updating and outputting correct position of NAME on the  
stack(NAMEDEPTH)  
Award* ***[1]*** *for use of stack methods (pop() and isEmpty())*

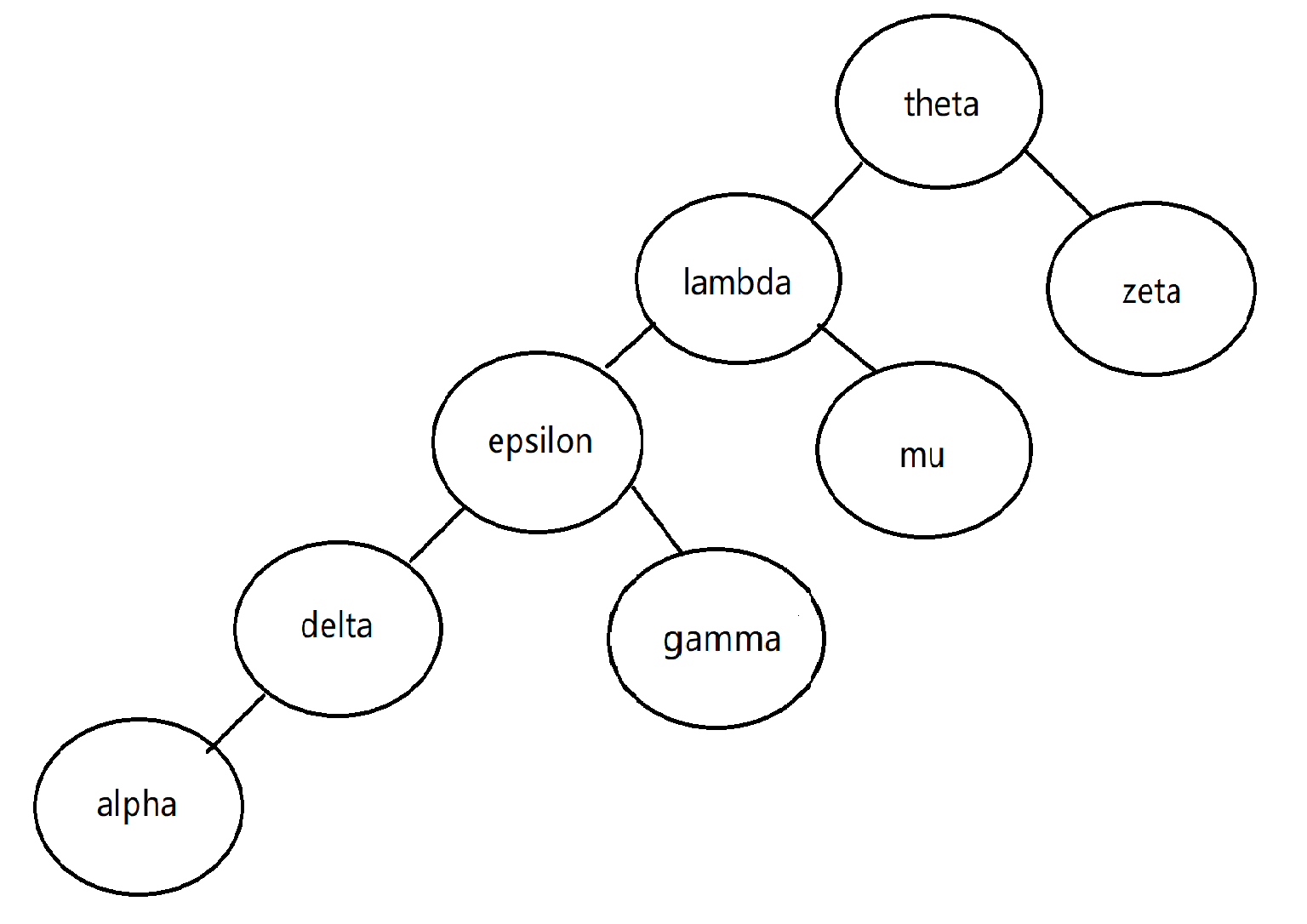
**(2)**

*Award* ***[3 max]****.*

*Example answer 1 (time efficiency):*The data in the binary search tree(BST) is ordered;  
Such that as each node is checked for the item, half of the remaining nodes  
are ignored;  
But each element in the stack has to be checked;  
Which, for large data sets, may be inefficient compared to the BST;

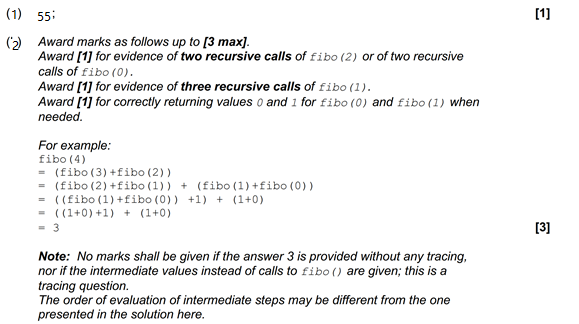
*Example answer 2 (memory efficiency):*Each element on the stack has to be popped off/removed from the stack to  
be checked for the searched item;  
When the item is found the stack will be empty/stack contents will be changed;  
When an element in the BST is checked for the item it is not removed from the  
BST;  
So there is no need to create an additional copy of the BST (but a real copy of  
the stack must be created which for large data sets may be inefficient);

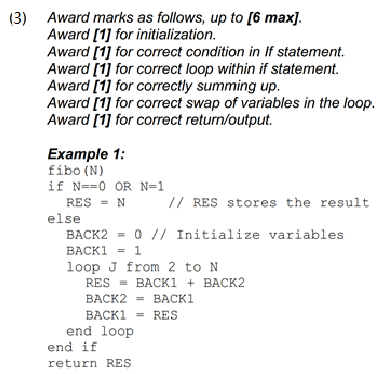
1. *Award* ***[4 max]****.  
   Example answer:*The (next) stack item is (placed into a new node and) compared (alphabetically)  
   to the root;  
   If the root is empty it would be placed here (and this recursive procedure  
   terminates);  
   Else, depending upon the comparison, it would look to the node to the left or right  
   and the (recursive) procedure calls itself again (with the new root);  
   If it is lower than the root, then the left child of the root becomes the new root;  
   If it is higher than the root, then the right child of the root becomes the new root;
2. *Award* ***[3]****.*

**

*Award marks as follows:*Clearly a binary tree;  
Correct root;  
All values correct;

**5.**

****

****

**6.**

**(1) 5 [1]**

**(2) Award marks as follow, up to [5 max]**

**Award [1] for initialization**

**Award [1] for correct for loop**

**Award [1] for correct while loop inside loop**

**Award [1] for correct value exchange**

**Award [1] for correct out put**

**InsertSort()**

**loop i from 2 to 10**

**key = array[ i ]**

**j = i-1**

**loop while j > 0 and array[ j ] > key**

**array[ j+1 ] = array [ j ]**

**j = j - 1**

**end while**

**end loop**

**output array**

**end InsertSort**

**(3) Award marks as follow, up to [3 max]**

**Award [1] for initialization**

**Award [1] for correct operation for queue**

**Award [1] for correct operation for stack**

**Inverse()**

**myStack// empty Stack**

**loop while not Q.isEmpty()**

**myStack.push( Q.dequeue() )**

**end loop**

**loop while not myStack.isEmpty()**

**myStack.pop()**

**end loop**

**end Inverse**

**C5**

1. **Start pointer points to the first in the list; [1]**

**Each subsequent pointer points to the next in the list and last node has null pointer [1]**

1. **a. A AND B**

**b. ABCDE CBDAE**

**(1) Helen Black**

**(2) [4]**

Use a variable (counter) to keep track of/increment the number of nodes;  
Use a temporary pointer;  
Follow the pointers from the beginning of the list/from the node pointed to by  
pointer X.next;  
Until the pointer to the end of the list (pointer X) is encountered;  
***Note:*** *Accept methods that start from the end of the list (X).*

1. [3]

Static data structure has a predetermined number of elements but number of  
elements in dynamic data structure does not have to be defined in advance;  
Static data structure has limited size, the amount of memory available is the only  
limit in size of dynamic data structure, size varies;  
In static data structure elements can be directly accessed, in a dynamic data structure  
access is sequential (which is slower);

(1) Award up to [2 max].

Each node contains data and also a link to other nodes;

Links between nodes are implemented by pointers (a pointer references a

location in memory or holds a memory address);

List size is not fixed / predetermined;

(2) Award up to [6 max] as follows. (There are 7 marking points)

[1] create new node;

[1] instantiation of values and pointers in new node;

[1] state where the search starts from;

[1] how to detect position for insertion;

[1] update pointers in new node;

[1] update pointers from the node at the insertion point, to the new node;

[1] update external pointers;

Create a new node (with pointer NEWNODE) with data field Primrose and two

pointer fields (next and previous), to be inserted;

Perform a linear search, either from the beginning or end of the list (using

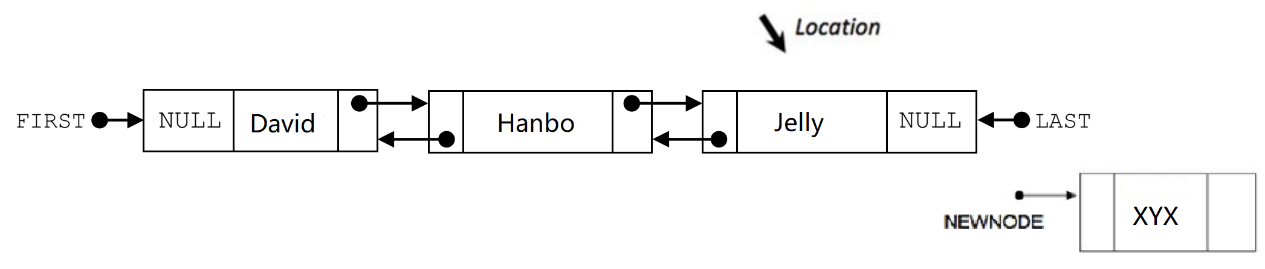
pointers FIRST and LAST, on the alphabetically order list;

The location/position of insertion, is found by comparing nodes (XYX to be

inserted after Jelly, LOCATION points to Jelly) (Accept any description

to that effect);

(At the end of this phase, the situation looks as in Figure 1)



Set the “previous” pointer in the newly created node to the current LAST / to point  
to Jelly/ to point to the node detected by LOCATION;

Change/Set/Update the Jelly “next” pointer to point to the new node / to  
link with the NEWNODE pointer (delete NULL in the field and link to the existing  
NEWNODE pointer);  
Update the LAST pointer to point to the newly created node;

