IMPERIAL COLLEGE LONDON

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING
EXAMINATIONS 2014

EEE PART II: MEng, BEng and ACGI

Corrected Copy

ALGORITHMS AND DATA STRUCTURES

Monday, 9 June 2:00 pm

Time allowed: 1:30 hours

There are TWO questions on this paper.

Answer BOTH questions.

Question One carries 40% of the marks. Question Two carries 60%.
This exam is OPEN BOOK.

Any special instructions for invigilators and information for candidates are on page 1.

Examiners responsible First Marker(s) : C. Bouganis
Second Marker(s) : D.B. Thomas

Special information for invigilators:
Students may bring any written or printed aids into the examination. The students are allowed to take a copy of the exam.

Information for candidates:
Marks may be deducted for answers that use unnecessarily complicated algorithms, or more than the minimum number of functions.

The Questions

1. a) Figure 1.1 shows a C++ function that calculates the value of the function described in equation (1.1), for any non-negative integer n (e.g. $f(0) = 0$).

$$f(n) = \begin{cases} 0, & n = 0 \\ f(n-1)+n, & 0 < n \leq 5 \\ f(n-1)+2, & 5 < n \end{cases} \quad (1.1)$$

Identify six syntactic or functional errors in the C++ code shown in Figure 1.1.

```
void calculateF (int N) {
    int result = -1;
    for (i=0; i <= N; i=i+2) {
        if (i ≤5)
            result = result + i/2;
        else
            result = result + 1;
    }
    return result;
}
```

Figure 1.1 calculateF() function. [6]

b) Write a recursive C++ function that performs the calculation described in part (a). [6]

[continued on the following page]

c) i) A list of numbers is inserted in an ordered binary tree (ascending ordered tree). Draw a tree for the following set assuming that the elements in the set are inserted in the order shown.

$\{5, 10, 15, 20, 25\}$ [2]

ii) State whether the tree from part (i) is balanced or not, and justify your answer. If the tree is currently unbalanced, then balance it using single and/or double rotations and draw the resulting tree. Show all intermediate steps. [6]

iii) What is the height of the tree resulting from part (ii)? [2]

iv) The same list of numbers as in part (i) is inserted into a chained hash table structure with six entries and the following hash function, where x is the inserted number:

$$H(x) = (x+1) \ mod \ 5$$

Draw the resulting hash table without any ordering imposed, assuming that the numbers in the set are inserted in the order shown. Comment on the suitability of the hash function selection for the given list of numbers. [2]

d) Draw a parse tree for the following expressions, assuming the normal priorities of the operators:

i) $10/12*3/4$ [2]

ii) $(2+6)/(5+9)*3$ [2]

[continued on the following page]

e) Consider the C++ code segment in Figure 1.2. With justification, state the values of variables *x*, *y* at points A and B of the code. With justification, state whether this code segment has a memory leak or not.

```
    int x=5;
    int y=10;
    int *p1 = new int;
    int *p2 = new int;
    x = *p1 + x;
    *p2 = y;
A:
    p1 = p2;
    x = *p1 + x;
    y = *p1 - y;
B:
```

Figure 1.2 Code segment.

[5]

[continued on the following page]

f) Figure 1.3 shows the type declaration for a dynamic linked list, where each node stores an integer in the *data* field.

```
struct Node {
    int data;
    Node * next;
};

typedef Node * NodePtr;
NodePtr hdList = NULL;
```

Figure 1.3 Linked list declaration.

i) Write a C++ function/procedure that takes as argument a pointer to the linked list and multiplies all values stored in the *data* field of the elements in the list by the constant 2. [3]

ii) Write a C++ function/procedure that takes a linked list as input and assigns the *next* field of the last element of the list to point to the first element of the list. You can assume that your function/procedure does not have to handle empty linked lists. Comment on what issues such actions can arise in processing the linked list further. [4]

2. Consider an ordered binary structure, where each node of the tree structure can store an integer number in the range $[-100, 100]$. Assume that ascending ordering has been imposed on the tree. An instance of the binary tree structure is shown below.

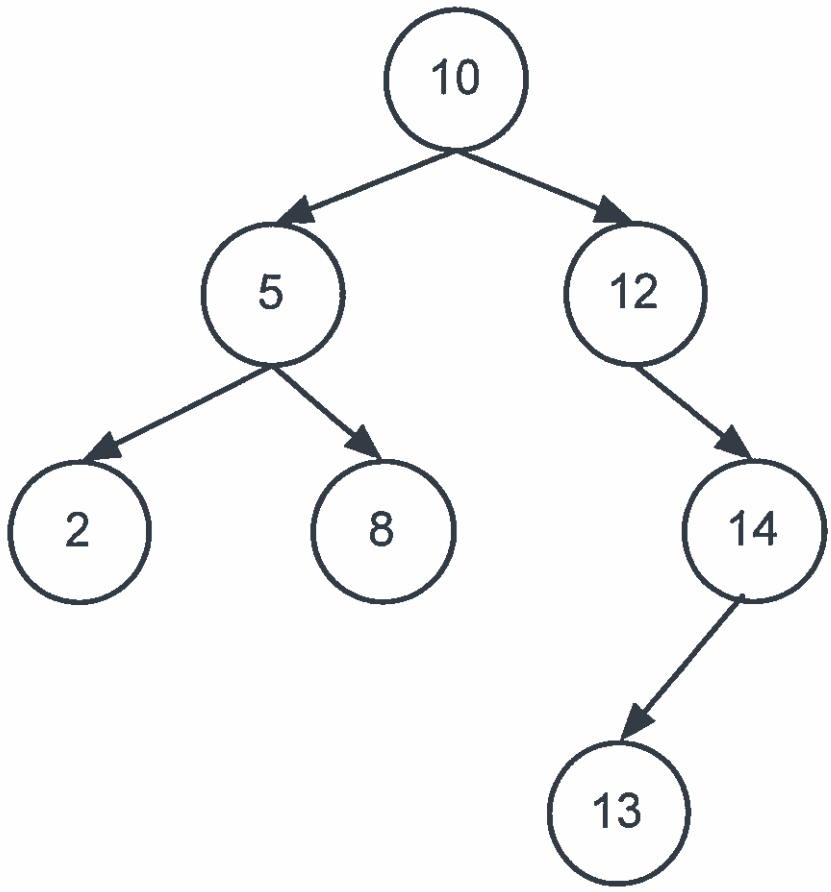

Figure 2.1 Tree structure.

a) Define a structure *treeNode* capable of representing a node of the tree. [5]

b) Write a recursive function/procedure that takes as input the pointer to the root of the tree and returns the largest number stored in the tree. Show how your recursive function/procedure will be invoked. You can always pass extra input arguments in your function/procedure. [10]

[continued on the following page]

c) Define a linked list structure capable of storing the same information as in the binary tree (no order is imposed on the linked list). Write a recursive function/procedure that takes as input a pointer to the root of the tree, and returns a pointer that points to the head of a linked list that contains the elements stored in the tree structure. Show how your recursive function/procedure will be invoked. You can always pass extra input arguments in your function/procedure. [10]

d) Write a recursive function/procedure that performs a similar operation to part c), but now the elements are stored in the list in descending order. Show how your recursive function/procedure will be invoked. You can always pass extra input arguments in your function/procedure. (Hint: It can be done by using a single function). [10]

e) Extend the structure *treeNode* by adding fields to store the number of nodes on the left and right subtrees of each node. Please provide the resulting structure. Write a recursive function/procedure that takes as input a pointer to the root of the tree, and updates each node's fields regarding the nodes on the left and right subtrees. Show how your recursive function/procedure will be invoked. You can always pass extra input arguments in your function/procedure. (Hint: You can use more that one functions/procedures to achieve this). [10]

f) Write a function/procedure that returns the median value of the elements stored in the tree structure. The extended Tree structure is assumed here, where all the fields in each node are updated. The median is the value that separates the higher half of the data from the lower half. In the case where the number of elements is even, the median value is defined as the mean of the two middle values. You can pass to your function the pointer to the root of the tree or the pointer to the linked list generated using one of the functions defined above. Show how your function/procedure will be invoked. You can always pass extra input arguments in your function/procedure. [15]