UNIVERSITY OF LONDON

CM2035

BSc EXAMINATION

COMPUTER SCIENCE

Algorithms and Data Structures II

Release date: Monday 22 March 2021 at 12 midday Greenwich Mean Time

Submission date: Tuesday 23 March 2021 by 12 midday Greenwich Mean Time

Time allowed: 24 hours to submit

INSTRUCTIONS TO CANDIDATES:

Section A of this assessment consists of a set of **TEN** Multiple Choice Questions (MCQs) which you will take separately from this paper. You should attempt to answer **ALL** the questions in Section A. The maximum mark for Section A is **40**.

Section A will be completed online on the VLE. You may choose to access the MCQs at any time following the release of the paper, but once you have accessed the MCQs you must submit your answers before the deadline or within **4 hours** of starting, whichever occurs first.

Section B of this assessment is an online assessment to be completed within the same 24-hour window as Section A. We anticipate that approximately **1 hour** is sufficient for you to answer Section B. Candidates must answer **TWO** out of the THREE questions in Section B. The maximum mark for Section B is **60**.

Calculators are not permitted in this examination. Credit will only be given if all workings are shown.

You should complete Section B of this paper and submit your answers as **one document**, if possible, in Microsoft Word or a PDF to the appropriate area on the VLE. You are permitted to upload 30 documents. However, we advise you to upload as few documents as possible. Each file uploaded must be accompanied by a coversheet containing your **candidate number**. In addition, your answers must have your candidate number written clearly at the top of the page before you upload your work. Do not write your name anywhere in your answers.

UL21/0265

SECTION A

Candidates should answer **ALL** of Question 1 in Section A.

Question 1

(a) Consider the following algorithm:

```
A: array of positive integer numbers
N: number of elements in A

function F1(A,N)
          if(N==0):
               return -1
          x=A[0]
          for (0 < i < N)
               if(A[i]>x):
                    x=A[i]
          return x
end function
```

Assume that the array A is equal to [3,2,1,4]. What value is returned by F1(A,4)? [4]

(b) Which of the following statements are **true**?

Select ALL statements that apply. [4]

i. Quicksort has the best worst-case time complexity possible for a comparison sort

ii. The worst-case time complexity of Mergesort on an array of length N is Θ(NlogN)

iii. The worst-case time complexity of Quicksort on an array of length N is Θ(Nlog N)

iv. Mergesort and Quicksort have the same worst-case time complexity

v. The worst-case time complexity of Quicksort on an array of length N is Θ(N)

vi. Mergesort has the best worst-case time complexity possible for a comparison sort

(c) Consider the following algorithm:

```
A: array of integer numbers
N: number of elements in A

function F2(A,N)
    if(N==0):
          return A
    mid=floor(N/2)
    for 0 <= i < mid
               aux=A[i]
               A[i]=A[N-i-1]
               A[N-i-1]=aux
    return A
end function
```

The operation floor(x) returns the largest integer smaller than or equal to x.

Select ALL statements that apply. [4]

i. The worst-case time complexity of F2 is Θ(N)
ii. The worst and best case time complexity of F2 are the same
iii. The worst-case time complexity of F2 is Θ(log N)
iv. The best-case time complexity of F2 is Θ(1)

(d) A 5-element hash table uses linear probing to deal with collisions. The hash function is h(k)=(2*k+1)%5. Assume the hash table starts empty. What is the content of it after inserting the following numbers (in this order): 23, 4, 6, 7? [4]

Choose ONE option:

i. None of the others
ii. [-1,23,6,4,7]
iii. [4,23,-1,6,7]
iv. [23,4,6,7,-1]
v. [7,-1,23,6,4]

(e) Consider the following recursive algorithm:

```
A: array of integer numbers
N: number of elements in A
L: left index of array A
R: right index of array A

function F3(A,L,R):
        if(L>=R):
              return A
        aux=A[L]
        A[L]=A[R]
        A[R]=aux
        return F3(A,L+1,R-1)
end function
```

What is the recurrence equation describing the worst-case time complexity of the function call F3(A,0,N-1)? [4]

Choose ONE option:

i. T(N)= T(N/2) + C (C is a constant)
ii. T(N)= T(N-1) + N
iii. None of the others
iv. T(N)= T(N-2) + C (C is a constant)
v. T(N)= T(N/2) + N

(f) Consider the graph represented by the following adjacency matrix:

	A	B	C	D	E
A	1	5	10	4	2
B	5	9	1	3	2
C	10	1	8	8	3
D	4	3	8	4	2
E	2	2	3	2	5

What is the cost of the minimum spanning tree? [4]

Choose ONE option:

i. 6
ii. None of the others
iii. 8
iv. 7
v. 10

(g) Which pseudocode fragment should replace Z in the following non-comparison-based sorting algorithm? [4]

```
A: array of nonnegative integer numbers
N: number of elements in A
k: maximum value stored in A

function Sort(A,N,k)
          C=new array(k+1) of zeroes
          R=new array(N) of zeroes
          pos=0
          for 0 <= j < N
               Z
          for 0 < i < k+1
               for pos <= r < pos+C[i]
                    R[r]=i
               pos=r
          return R
end function
```

Choose ONE option:

i. C[A[j]]=A[j] + 1
ii. None of the others
iii. C[A[j]]=C[A[j]] + C[A[j-1]]
iv. C[A[j]]=C[A[j]] – 1
v. C[A[j]]=C[A[j]] + 1

(h) Consider the following linked list:

head/0xA → 10/0xE → 7/0xD → 3/NULL

A new node is inserted at the start of the list. Assume the new node stores number 3 and it is allocated memory address 0xB. What are the contents of the new list? [4]

Choose ONE option:

i. head/0xA → 3/0xB → 10/0xE → 7/0xD → 3/NULL
ii. None of the others
iii. 3/0xB → head/0xA → 10/0xE → 7/0xD → 3/NULL
iv. head/0xA → 10/0xE → 7/0xD → 3/NULL → 3/0xB
v. head/0xB → 3/0xA → 10/0xE → 7/0xD → 3/NULL
vi. head/0xA → 10/0xE → 7/0xD → 3/0xB → 3/NULL

(i) The following numbers are inserted in a Binary Search Tree (in this order): 4, 3, 8, 5, 6, 2

What information in printed on screen when traversing the tree with the algorithm shown below? [4]

```
function traverse(T)
    if(T.root!=NULL)
        traverse(T.left)
        traverse(T.right)
        print(T.root)
end function
```

Choose ONE option:

i. 6 5 8 2 3 4
ii. 2 3 4 8 5 6
iii. 6 5 8 4 3 2
iv. None of the others
v. 2 3 6 5 8 4

(j) The following numbers are inserted, one by one, in a max-heap (in this order): 6, 5, 4, 8, 9, 3, 2

What is the content of the array storing the heap? Position 0 in the array is the leftmost position. [4]

Choose ONE option:

i. [8,9,4,5,6,3,2]
ii. [8,6,4,5,9,3,2]
iii. None of the others
iv. [9.4,8,3,2,5,6]
v. [9,8,5,6,4,3,2]

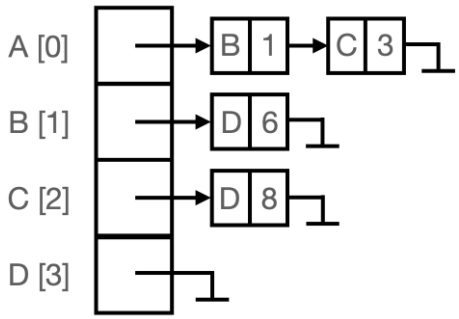
SECTION B

Candidates should answer any **TWO** questions from Section B.

Question 2

This question involves representations of graphs and finding the shortest paths through a graph.

(a) Consider the following adjacency list of a graph with four vertices, A, B, C and D, where the elements on the left index these vertices:

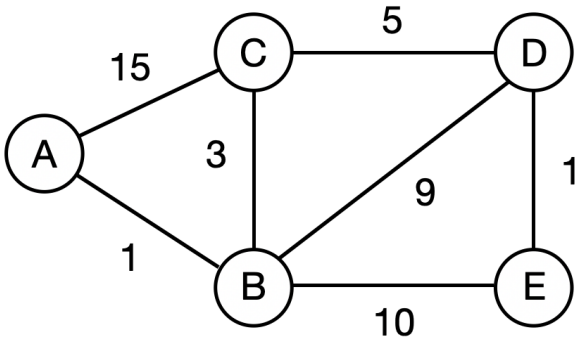


Draw the corresponding adjacency matrix for this graph. Represent no edge between vertices with a weight of zero in the matrix. [4]

(b) Is the graph represented by the adjacency list in part (a) of this question directed or undirected? [1]

(c) What are the circumstances in which it is better to use an adjacency list to represent a graph rather than using an adjacency matrix? Explain your reasoning. [6]

(d) Consider the following graph:



By hand, go through an implementation of Dijkstra's algorithm on this graph to find the shortest path between vertices A and D. Your implementation should produce both the path's length and the nodes in it. [9]

(e) Given an undirected (connected), weighted graph, the **shortest-path tree** with its root at vertex v is a spanning tree of the graph such that the path from vertex v to every other vertex in the tree is the shortest path in the graph. Describe a method based on Dijkstra's algorithm for producing the shortest-path tree of a given graph. Explain why the method works. [10]

Question 3

The following two algorithms claim to solve the same problem. To do so, they receive as an input argument:

root: the root of a binary search tree (BST) storing nonnegative integer numbers

ALGORITHM 1

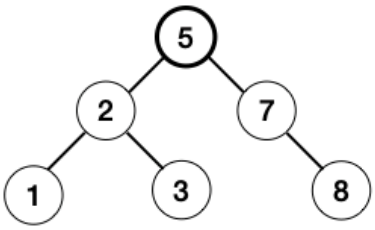
```
function A1(root)
   if(root == NULL)
        return -1
   x=root->data
   Q=new Queue()
   ENQUEUE(Q,root)
   while !ISEMPTY(Q) do
        t=PEEK(Q)
        if(t->data < x)
            x=t
        else
            ENQUEUE(Q,t->left)
            ENQUEUE(Q,t->right)
            DEQUEUE(Q)
   end while
   return x
end function
```

Note: the function ENQUEUE only inserts a new element in the queue if this element is different from NULL

ALGORITHM 2

```
function A2(root)
   if(root == NULL)
      return -1
   t=root
   while(t->left != NULL)
      t=t->left
   return t->data
end function
```

(a) For the following BST:



What is returned by the function call A2(root)? [2]

(b) For the BST in part (a), what is the content of the queue **immediately before returning** from the execution of A1(root)? [2]

(c) In pseudocode rewrite the function A2 using recursive function calls. You may not use any form of iteration. [6]

(d) For a fully balanced BST of N elements (with all levels fully populated), write down the recurrence relation describing the running time of your recursive function in part (c) of this question. Explain your reasoning. [5]

(e) For a fully balanced BST of N elements (with all levels fully populated), what is the worst-case time complexity of the function A1 in terms of N? Use Theta notation (for one mark) and explain your reasoning (for three marks). [4]

(f) For a general BST of N elements, what is the worst-case time complexity of the function A1 in terms of N? Use Theta notation (for one mark) and explain your reasoning (for three marks). [4]

(g) For a general BST of N elements, which of the two algorithms A1 and A2, should you use? Give your choice and explain your reasoning (for at most four marks). At most three additional marks are given for the quality of the explanation. [7]

Question 4

This question is about linked lists and hashing functions.

(a) Consider the following linked list:

head/1xA → 2/0xB → 4/NULL

A pointer called temp stores the memory address 0xB. Explain how a new node storing the value 5 can be added to the end of this linked list utilising the pointer temp. You may use diagrams. [5]

(b) For a general linked list, write a function INSERT(head,x) in pseudocode that inserts a new node at the end of a linked list storing the value x. The input arguments are head, the pointer to the head of the list, and x, the value being stored at the end of the list. [8]

(c) What is the worst-case time complexity in N for inserting a new node with value x at the head of a linked list with N nodes? Use Theta notation. [1]

(d) What is the worst-case time complexity in N for searching an unordered linked list with N nodes for a value x? Use Theta notation. [1]

(e) An N-element hash table using linear probing can be used to store M values, such that the location of the value in the table is determined by a hashing function applied to that value. What is the worst-case time complexity in N of searching an N-element hash table for a value x? Use Theta notation [for one mark] and explain your reasoning [for four marks]. [5]

(f) The following text is a proposal to use a hash table to simulate the data organisation and manipulation of a linked list. Explain the shortcomings of this proposal: [10]

> The value stored in the head of a list can be stored in bucket 0 of a hash table, then to find the next value in the corresponding linked list, a hash function is applied to the value in the current bucket: the hashed value will dictate into which bucket the value is stored.

END OF PAPER