UNIVERSITY OF LONDON

CM2035

BSc EXAMINATION

COMPUTER SCIENCE

Algorithms and Data Structures II

Release date: Wednesday 15 September 2021 at 12:00 midday British Summer Time

Submission date: Thursday 16 September 2021 by 12:00 midday British Summer Time

Time allowed: 24 hours to submit

INSTRUCTIONS TO CANDIDATES:

Section A of this assessment paper consists of a set of **10** 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 paper 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** written clearly at the top of the page before you upload your work. Do not write your name anywhere in your answers.

SECTION B

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

Question 2

Numbers in binary can be represented as arrays of single bits, e.g. the number 35 in decimal in binary is 100011, and the corresponding array is [1,0,0,0,1,1]. This question is about multiplying integers in terms of these binary arrays. That is, given two arrays of bits representing two integers, produce a new array that is the corresponding binary representation of the two integers multiplied. For instance, given [1,0,0,0,1,1] and [1,1,0], which are 35 and 6 respectively, an algorithm should produce [1,1,0,1,0,0,1,0], which is 210, the product of 35 and 6.

We can assume that the integers have binary representations both of length N. This can be always be achieved by padding the beginning of the array with extra zeroes. In the example above the two input arrays can be made [1,0,0,0,1,1] and [0,0,0,1,1,0].

The first pseudocode function we consider adds together two N-length arrays:

```
function Add(A,B,N)
   if(N==0):
      return empty array
   C1=new array(N+1) of zeroes
   C2=new array(N+1) of zeroes
   i=N-1
   while i >= 0
      C1[i+1]=A[i]+B[i]+C2[i+1] mod 2
      if(A[i]+B[i]+C2[i+1]<2):
         C2[i]=0
      else:
         C2[i]=1
      i=i-1
   if(C2[0]==0):
      C3=new array(N) of zeroes
      for 0 < i <= N
         C3[i-1]=C1[i]
      return C3
   else:
      C1[0]=1
      return C1
end function
```

(a) What is the worst-case time complexity of the function `Add` in terms of the length of the arrays N? Explain the worst-case inputs arrays are of length N (2 marks), use Theta notation (1 mark) and briefly explain your reasoning (4 marks).

(7 marks)

The following pseudocode function multiplies two arrays of length N storing bits:

```
function Multiply(A,B,N)
   if(N == 0):
      return empty array
   C1=new array(2N) of zeroes
   C2=new array(2N) of zeroes
   for 0 <= i < N
      for 0 <= j < N
         C1[(2N-1)-j-i]=A[N-1-j]*B[N-1-i]
      C2=Add(C1,C2,2N)
   return C2
end function
```

(b) What is the worst-case time complexity of the function `Multiply` in terms of the length of the arrays N? Use Theta notation (1 mark) and briefly explain your reasoning (4 marks).

(5 marks)

(c) One of your colleagues says that no algorithm will be able to compute the multiplication of two arrays of bits of length N in time better than Ω(N). Do you agree or disagree with them? Explain your answer.

(4 marks)

Consider the following piece of pseudocode:

```
function NewMultiply(A,B,N)
   if(N == 0):
      return 0
   else if(N == 1):
      return A[0]*B[0]
   h=floor(N/2)
   x=N mod 2
   A1=new array(h+x) of zeroes
   B1=new array(h+x) of zeroes
   for x <= i < h
      A1[i]=A[i]
      B1[i]=B[i]
   A2=new array(h+x) of zeroes
   B2=new array(h+x) of zeroes
   for h <= i < N
      A2[i]=A[i]
      B2[i]=B[i]
   m1=NewMultiply(A1,B1,h+x)
   m2=NewMultiply(A1,B2,h+x)
   m3=NewMultiply(A2,B1,h+x)
   m4=NewMultiply(A2,B2,h+x)
   return m1*(2^(2h+2x)) + (m2+m3)*(2^(h+x)) + m4
end function
```

This piece of pseudocode computes the (decimal) integer obtained from multiplying the two arrays of length N storing bits. The operation `floor(x)` returns the largest integer smaller than or equal to x.

(d) What is the worst-case time complexity of the function `NewMultiply` in terms of the length of the arrays N? Use Theta notation (1 mark) and show your working to arrive at this answer (6 marks). Assume that the floor of a number and x mod 2 can be computed in a constant number of time-steps.

(7 marks)

(e) Another colleague claims that the algorithm in `NewMultiply` has the best worst-case time complexity for any algorithm that takes two arrays of bits and computes the (decimal) integer corresponding to the multiplication of the two integers in the arrays. Do you agree or disagree? Explain your answer. You can use any form of research for your answer as long as you appropriately reference your resources.

(7 marks)

Question 3

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

root: the root of a binary search tree (BST) storing positive integer numbers greater than 0
N: the number of nodes in the BST rooted at root

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

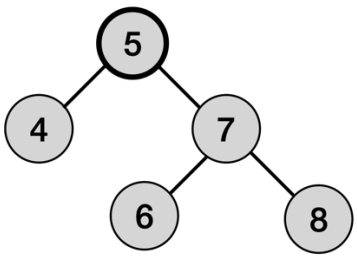
ALGORITHM 1

```
function A1(root,N)
   if(N == 0):
      return -1
   S=new Stack()
   A=new array(N) of zeroes
   i=0
   x=root
   while !ISEMPTY(S) or x != NULL do
      while x != NULL do
         PUSH(S,x)
         x=x->left
      x=PEEK(S)
      POP(S)
      A[i]=x->data
      i=i+1
      x=x->right
   return A[floor(N/2)]
end function
```

ALGORITHM 2

```
function A2(root,N)
   if(N == 0):
      return -1
   A=new array(N) of zeroes
   Q=new Queue()
   ENQUEUE(Q,root)
   while !ISEMPTY(Q) do
      x=PEEK(Q)
      A[0]=x->data
      i=0
      while A[i]>A[i+1] and i<N-1 do
         t=A[i]
         A[i]=A[i+1]
         A[i+1]=t
         i=i+1
      ENQUEUE(Q,x->left)
      ENQUEUE(Q,x->right)
      DEQUEUE(Q)
   return A[floor(N/2)]
end function
```

Consider the following BST:



(a) For this BST, what is returned by the function call A2(root,6)?

(1 mark)

(b) What is the task performed by algorithms A1 and A2?

(3 marks)

(c) What is the worst-case time complexity of A2 for a BST of N nodes? Use Theta notation (1 mark) and briefly explain your reasoning (4 marks).

(5 marks)

(d) What is the worst-case time complexity of A1 for a BST of N nodes? Use Theta notation (1 mark) and briefly explain your reasoning (4 marks).

(5 marks)

(e) Explain how you could improve algorithm A1 so that it could use less memory (space) and/or need fewer operations.

(5 marks)

(f) Rewrite the pseudocode of algorithm A1 so that it uses recursion instead of using a stack. You will not get any marks if your attempt uses a stack.

(5 marks)

(g) One of your colleagues has claimed that they can find an algorithm that performs the same task as A1 and A2, but the worst-case time complexity is $\Theta(\log N)$, for N nodes in a BST. Do you agree or disagree with your colleague? Explain the reasoning behind your opinion.

(6 marks)

Question 4

This question is about hashing and heaps.

(a) Briefly explain the concept of a collision in the context of a hash function (4 marks). Give an example demonstrating this concept (2 marks).

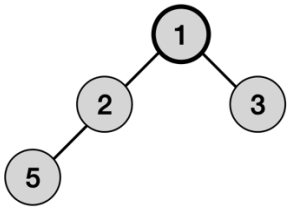
(6 marks)

(b) One of your colleagues claims that collisions are unavoidable for any hash function. Do you agree or disagree? Briefly explain your reasoning.

(6 marks)

A software developer wants to use hash functions to implement a min heap. Given a heap in its tree representation, the value in the root is stored in a separate variable `root`. There are two arrays, called `left` and `right`, and a hash function f and for each node with value x, we have the hashed value f(x). The values of the left child and right child of each node are stored at index f(x) in the `left` and `right` arrays respectively. If there is no child then the value stored at this element is -1.

For instance, consider the following min heap:



The corresponding arrays `left` and `right` are assumed to have five elements initially storing `-1` in each element. Assume the hash function is f(x) = 2x mod 5 so `root` stores `1`, and thus the arrays `left` and `right` are `[-1,-1,2,-1,5]` and `[-1,-1,-1,3,-1]` respectively to represent this tree.

(c) For simplicity assume the hash function used does not have collisions for the range of values to be stored in the min heap. Also assume that all the values to be stored in the min heap are distinct, positive integers (not equal to 0). Describe a method to insert a value into the heap such that the min-heap property is satisfied after the method is completed. The method can be described in words, with pseudocode used to aid the description.

(10 marks)

(d) In the case where a hash function has collisions, briefly explain why linear probing for resolving collisions is not helpful in implementing the min heap based

on hashing (4 marks). Briefly explain how the extend and re-hash method could be more useful in this context (4 marks).

(8 marks)

END OF PAPER