Name: Rollno:

ESO207: Data Structures and Algorithms (Quiz 1)

9th September 2023

Total Number of Pages: 6 Time: 1 hr Total Points 40

Question	Points	Score
1	3	
2	3	
3	3	
4	3	
5	3	
6	3	
7	3	
8	3	
9	3	
10	3	
11	10	
Total:	40	

Instructions

1. All questions are compulsory.
2. Answer all the questions in the question paper itself.
3. MCQ type questions can have more than one correct options.
4. The symbols or notations mean as usual unless stated.
5. You may cite and use algorithms and their complexity as done in the class.
6. Cheating or resorting to any unfair means will be severely penalized.
7. Superfluous and irrelevant writing will result in negative marking.
8. Using pens (blue/black ink) and not pencils. Do not use red pens for answering.

Helpful hints

1. It is advisable to solve a problem first before writing down the solution.
2. The questions are *not* arranged according to the increasing order of difficulty.

Question 1. (3 points) We can calculate the n^{th} Fibonacci number in logarithmic time by matrix exponentiation of the matrix

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

Similarly, we can calculate the n^{th} term of any recursion using matrix exponentiation of some matrix A. Find A for the following recursion:

$$F(N) \;=\; a_1 \cdot F(N-1) \;+\; a_2 \cdot F(N-2)$$

Solution: $A = \begin{bmatrix} a_1 & a_2 \\ 1 & 0 \end{bmatrix}$.

The transpose of this matrix will also be considered as a correct solution.

Question 2. (3 points) Rank the following functions in increasing order of growth.

(a) $O(n!)$
(b) $O(2^n)$
(c) $O(\log^3 n)$
(d) $O(n)$
(e) $O(n^{1/2})$

Solution: $(c) < (e) < (d) < (b) < (a)$

Question 3. (3 points) Suppose a K-ary tree is a tree data structure in which each node has at most K children (assume $K > 1$). Define functions f and g as:

$f(K, H)$ is the maximum number of nodes in a K-ary tree of height H. (Note that for any tree with single node, has a height $H = 0$.)

$g(K, L)$ is the maximum number of nodes at any level L in a K-ary tree. (Note that for any tree, the level of the root is taken as $L = 0$.)

Then $f(K, H)$ is $\underline{\;\frac{K^{H+1}-1}{K-1}\;}$ and $g(K, L)$ is $\underline{\;K^L\;}$.

Question 4. (3 points) Given the following relation for an algorithm, where $T(n)$ denotes the time taken to execute the algorithm on an input of size n. What is the best bound for $T(n)$ in Big-O notation? (Assume $T(1) = 1$)

$$T(n) = T(n/2) + n$$

- ✓ $\mathcal{O}(n)$
- ☐ $\mathcal{O}(\log n)$
- ☐ $\mathcal{O}(n \log n)$
- ☐ $\mathcal{O}(n^2 \log n)$

Question 5. (3 points) Which of the following statements are correct?

- ☐ A stack can be implemented using a two queues, that performs the push and pop operations in constant time.

- ☐ A stack can be implemented using a single queue, that performs the push and pop operations in constant time.
- ☐ A queue can be implemented using a single stack, that performs the enqueue and dequeue operations in constant time.
- ☐ A queue can be implemented using two stacks, that performs the enqueue and dequeue operations in constant time.

Solution: As stated, the question is incorrect. To implement a stack using two queues or a queue using two stacks, you require $O(n)$ time for one of the operations, in worst case. So you will be awarded full points for this question.

Question 6. (3 points) You have an array A of length n that is strictly increasing upto some index i where ($1 \leq i \leq n$ and is strictly decreasing after that. For such an array, you have the following two statements:

I We can find the maximum element of the array in $\mathcal{O}(\log n)$ time using binary search.

II We can search for any element in the array in the $\mathcal{O}(\log n)$ time using binary search.

Which of the following option is correct?

- ☐ None of the statements are correct
- $\surd$ **Both the statements are correct**
- ☐ Only statement I is correct
- ☐ Only statement II is correct

Question 7. (3 points) Which among the following is/are correct?

- ☐ Searching an element in a sorted linked list takes $O(\log n)$ time.
- $\surd$ **Searching an element in an unsorted linked list takes $O(n)$ time.**
- ☐ If a singly-linked list contains n nodes, only with a pointer to the first node and a pointer to the last node in the list, then the last node can be deleted in the worst case $O(1)$ time.
- $\surd$ **Time complexity of an algorithm to check if the array of n elements is sorted, is same as the one for checking if a linked list of n elements is sorted**

Question 8. (3 points) There is an input stream of numbers, and you have a stack and queue. You can choose to insert the next number in the input stream, into either the stack or the queue. Once the input stream is finished, you output the elements, first by popping all elements of the stack one by one, and then once the stack is empty, output by dequeueing all elements of the queue one by one.

Given the input sequence to be 1, 2, 3, 4, 5

Which of the following output sequences is/are possible?

- ☐ 5 1 4 2 3
- $\surd$ **4 2 1 3 5**
- $\surd$ **5 4 1 2 3**
- ☐ 1 2 5 4 3

Question 9. (3 points) Consider the following pseudo-code for a function **Mystery**(T, n) where T is a binary search tree and n is a positive integer.

Algorithm 1: Mystery(T, n)

```
if n == 1 then
  | return;
else
  | Insert(T, n);
  | if n%2 then
  |   | n ← 3 * n + 1;
  | else
  |   | n ← n/2;
  | end
  | Mystery(T, n);
end
```

where **Insert**(T, n) is a function that inserts a number n in the given BST T.

Suppose you start with an empty BST, B, and call the function **Mystery**$(B, 17)$. Answer the following with respect to BST B thus formed

(a) Height of B is ____**5**____.

(b) Right-most element of B is ____**52**____.

> **Solution:** Although the correct answer to this question is 52, we will also consider 40 as an answer since the definition of rightmost element was a bit confusing to some students.

(c) Is B perfectly balanced (Yes/No)? ____**No**____

Question 10. (3 points) Consider the pseudo-code for a function **MyInsert**

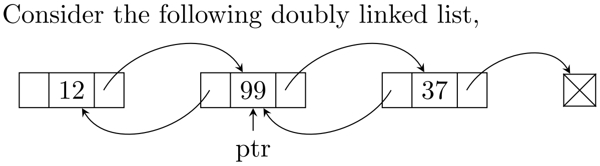
Algorithm 2: MyInsert(L, val, p)

$q \leftarrow createNewNode(val)$;
$p.prev = q$;
$q.next = p$;
$q.prev = p.prev$;
$p.prev.next = q$;

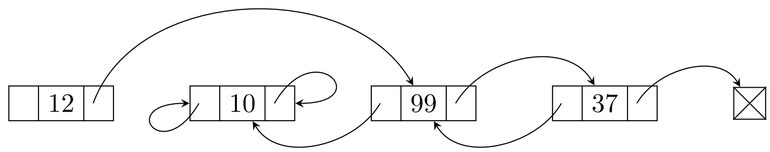
where, L is a doubly linked list, val is the new value to be inserted, and p is the pointer to the node at which insertion is to be done.

Now, Consider the following doubly linked list,

L :



What will the list look like, if we call **MyInsert**$(L, 10, ptr)$, where L and ptr are from the figure above



Solution:

Question 11. In this question we calculate the median of the values stored in a BST having height h. Each node of the BST stores the following information.

1. $left$: Pointer to the left child
2. $right$: Pointer to the right child
3. val : Value stored at this node
4. $size$: Size of the subtree rooted at this node

(a) (5 points) Give the pseudocode for the function **Insert**(T, val) in $O(h)$ time, that inserts a node with value val in the tree T, and correctly updates the $size$ attribute of the concerned nodes.

Solution:

Algorithm 3: Insert$(T, value)$

```
p ← T;
while p ≠ NULL do
    p.size ← p.size + 1;
    if p.val > value then
        p ← p.left;
    else
        p ← p.right;
    end
end
p.val ← value;
p.left ← NULL;
p.right ← NULL;
p.size ← 1;
```

(b) (5 points) Give the pseudocode for the function **Median**(T) to find the median of the BST T in $\mathcal{O}(h)$ time. For simplicity, you can assume there are an odd number of elements in the BST.

Solution:

Algorithm 4: Median(T)

```
index ← n/2 + 1;
p ← T;
while p.left.size ≠ index − 1 do
    if p.left.size > index − 1 then
        p ← p.left;
    else
        index ← index − (p.left.size + 1);
        p ← p.right;
    end
end
return p.val;
```