

112Exam2

Wednesday, November 13, 2019 6:36 PM



CS112OldMi
dtermExa...

CS 112

MidtermExam A

65 minutes

Closed Book and no Aid allowed

15th November 2017

Instructor: Abbas Attarwala

Boston University

Note: It is a violation of the Honor Code to discuss this midterm exam question with anyone until after everyone in CS 112 (A and B section) has taken the exam.

Enter your First Name: _____

Enter your Last Name: _____

Enter your Student#: _____

"I pledge my honor that I have not violated the Honor Code during this examination."

Your Signature: _____

Question:	1	2	3	4	Total
Points:	13	8	20	0	41
Bonus Points:	0	0	0	3	3
Score:					

DO NOT TURN THIS PAGE UNTIL YOU HAVE BEEN ASKED TO DO SO

- Given the following Linked List class¹. You will find the sub-questions on the next page.

```
public class LinkedList<E extends Comparable<E>> {
    private static class Node<E>
    {
        private E data;
        private Node<E> next;

        public Node(E d)
        {
            data=d;
            next=null;
        }
    }
    private Node<E> head;
    private Node<E> tail;

    public boolean increasingSequence() throws EmptyLinkedList
    {
        //TODO: Complete the rest of this method.
    }
}
```



```

    }

    /*
     * n must be non-null when _increasingSequence is called.
     * You must use recursion here.
     */
    private boolean _increasingSequence(Node<E> n)
    {
        if (n.next==null)
            return true;
        Node<E> previous=n;
        Node<E> next=n.next;
        //TODO: Complete the rest of this method.

    }

}

```

¹The template for the `LinkedList` class is exactly the same as seen in lecture

- (a) Complete the method `increasingSequence2` such that the method returns back a `true` when the `LinkedList` is in an increasing order and returns back `false` when the `LinkedList` is not in an increasing order. This method returns back an exception `EmptyLinkedList3` when the `LinkedList` is empty (i.e. head points at `null`). You can call the `compareTo` method to compare two `Nodes`. We provide the following 9 `LinkedLists` and their expected output.

 <code>3 --> 7 --> 5 --> null</code>	<i>must return FALSE</i>
<code>3 --> 4 --> 5 --> 6 --> null</code>	<i>must return TRUE</i>
<code>3 --> 5 --> 7 --> 9 --> null</code>	<i>must return TRUE</i>
<code>3 --> 3 --> 3 --> null</code>	<i>must return FALSE</i>



(7)

<code>3 -- > 4 -- > 5 -- > 7 -- > null</code>	<i>must return TRUE</i>
<code>6 -- > 5 -- > 4 -- > 3 -- > null</code>	<i>must return FALSE</i>
<code>- 1 -- > 0 -- > null</code>	<i>must return TRUE</i>
<code>3 -- > null</code>	<i>must return TRUE</i>
<code>< EMPTYLINKEDLIST ></code>	<i>must throw EmptyLinkedList Exception</i>

You can use the space provided in the code to write your answer.

On .java file

- (b) Write (do not solve) the recurrence $T(n)$ for the `_increasingSequence`.

$$\Downarrow T(n) = \underbrace{T(n-1)}_{\uparrow} + C \quad \underbrace{+ C}_{\uparrow}$$

- (c) Solve the recurrence $T(n)$ for the `_increasingSequence` from b) above, in the worst case and find the $\mathcal{O}(...)$. Show us all the steps and insert your final $\mathcal{O}(...)$ answer by drawing a box around it.

$$\begin{aligned} \Downarrow T(n) &= T(n-1) + C \\ &= [T(n-2) + C] + C = T(n-2) + 2C \\ &= [(T(n-3) + C) + C] + C = T(n-3) + 3C \\ &\vdots \\ &= \end{aligned}$$

Total for Question

²This problem is very similar to `removeDuplicates` from review session.

³You do not have to create this class

2. Given the following general tree. There is no coding required for this question.

(2)

(4)

n 1: 13

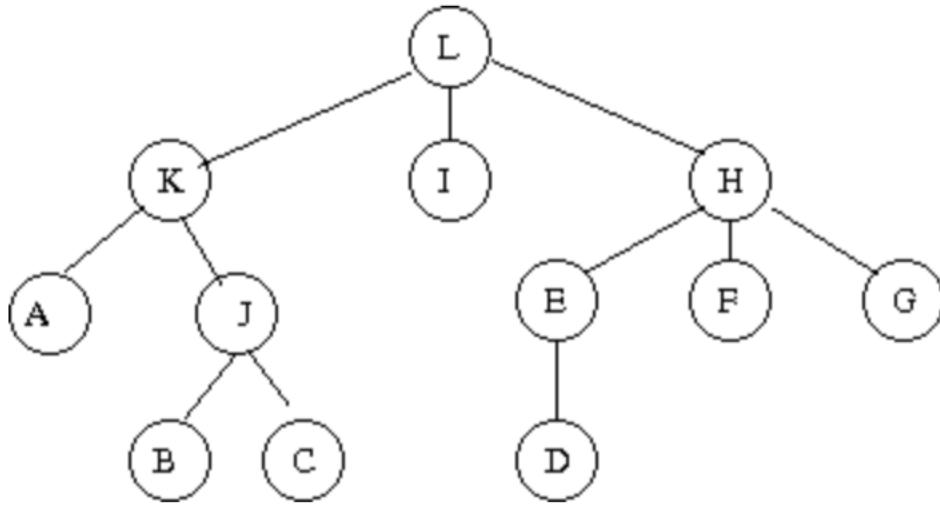


Figure 1: Tree with 12 nodes

(a) What is PreOrder traversal on the above tree?

LKAJBCJHEDFG

(b) What is PostOrder traversal on the above tree?

ABCJKIEDFGHL

(c) What is breadth-first order traversal or levelwise order traversal on the above tree?

L KIH AJEFG BCD

(d) How many leaf nodes does the above tree contain?

7

(e) What is the height of the above tree?

4

Total for Question

(2)

(2)

(2)

(1)

(1)

on 2: 8

3. Given the following `BinaryTree` class

```
public class BinaryTree<T> {
    private static class Node<T>
    {
        private int data;
        private Node<T> left;
        private Node<T> right;

        public Node(int d)
        {
            data=d;
        }
    }
    private Node<T> root; //points at the root of the tree

    /* Returns the height of the tree. The height of an empty tree OR
     * a tree with just a single node is 0. You must use recursion
     * on _getHeight to answer this
     * question. */
    public int getHeight()
    {
        //TODO
    }

    private int _getHeight(Node<T> r)
    {
        //TODO
    }
}
```



```
}
```

```
/* Returns the clone of BinaryTree represented by the reference 'this'  
 *Pay careful attention to the return types of clone  
 *and _clone. */  
public BinaryTree<T> clone ()  
{  
    BinaryTree<T> ret=new BinaryTree<>();  
    //TODO  
  
}  
private Node<T> _clone(Node<T> r)  
{  
    if (r==null)  
        return null;  
    //TODO  
  
}  
}
```

- (a) In the lecture, we calculated the height of a general Tree using recursion. You are now asked to find the height of a BinaryTree⁴. You will use recursion and complete getHeight and _getHeight. You can use the space in the code to answer this question.

(7)

On .java file

- (b) In your assignment3 you cloned a Set and then in your assignment6 you cloned a LinkedList.
In this question, we ask you to clone a BinaryTree. You will use recursion and complete `clone` and `_clone`. You can use the space in the code to answer this question.

on .java file.

⁴Hint: You can use the `max` function from the `Math` class

Page 6 of 10

- (c) In the main function provided, write code so that you create the following `BinaryTree`⁵ using the above `BinaryTree` class.

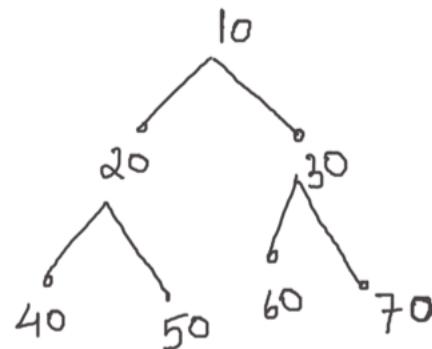


Figure 2: BinaryTree with 7 nodes

```
//You can assume that this main function belongs inside the above  
//BinaryTree class.  
public static void main(String[] args)  
{
```

(7)

(6)

}

Total for Question

⁵In lecture we created similar trees by hardcoding values to test various algorithms. You can use similar technique when answering this question

Page 7 of 10

4. (a) Given the following code⁶:

```
for (i=1; i<=n; i++)
    for (j=1; j<=i; j++)
        k=k+1;
```

(1 1/2 (b)

You are asked to count the actual number of times the instruction $k = k + 1$ is executed and then find the $\mathcal{O}(\dots)$. Your friend from NorthEastern performs the following series of steps:

$$\begin{aligned}&= \sum_{i=1}^n \sum_{j=1}^n 1 \\&= \sum_{i=1}^n n \\&= n \sum_{i=1}^n 1\end{aligned}$$

n 3: 20

onus))

$$= n * n$$

$$= n^2$$

i.e. n^2 is $\mathcal{O}(n^2)$. The final $\mathcal{O}(n^2)$ answer is correct. However, there is a flaw in the above steps.
What is the flaw and repeat the steps⁷ such that the flaw is corrected.

$$\begin{aligned}
 &= \sum_{i=1}^n \sum_{j=1}^i 1 \\
 &= \sum_{i=1}^n n \\
 &= n \cdot n \\
 &= n^2
 \end{aligned}$$

⁶Similar to Piazza post @199

⁷You must also use the \sum in your step(s) similar to what your friend did.

- (b) What is the $\mathcal{O}(\dots)$ for the following piece of code? Make sure that you explain your final answer! 1.1/2 (b)

```
public void someFunction(int n){
    int i, j, k, count=0;
    for (i=n/2; i<=n; i++)
        for (j=1; j<=i; j++)
            for (k=1; k<=j; k++)
                count++;
    }
```

$\text{Time complexity: } O(n \cdot \frac{n}{2}) = O(n^2)$

onus))

```

{
    for (j=1; j<=n; j=2*j)  $\leftarrow O(\log n)$ 
    {
        for (k=1; k<=n; k=k*2)  $\leftarrow O(n \log n) < O(n \cdot \frac{n}{2}) = O(n)$ 
        {
            count++;
        }
    }
}

```

$$O(n) \cdot \left[O(\log(n)) \cdot [O(n)] \right] = O(n^2 \log(n))$$

Extra Rough Page

