KLE Technological University
Creating Value
Leveraging Knowledge

Earlier known as
B. V. B. College of Engineering & Technology

School of Computer Science and Engineering

MINOR EXAM I - Solutions

Course : Data Structures and Algorithms

SRN

0	1	F	E			B	C	S			

Course Code: 19ECSC201
Semester: III
Date of Exam: 25 Sept 2019
Duration: 1 hour 15 minutes

HAPPY BLACKOUT!

Answer any TWO full questions. There is no missing or error data in the paper. If you feel so any, well, you shouldn't! Period.

QUESTION 01

1a Identify the orders of growth for:

i) To check if there is a Hamiltonian path in the given graph (Hello GTLA!)
Non-polynomial

04 marks
CLO4
L2

ii) Finding the composition of two given relations X and Y (Composites from DMS)
Cubic

iii) A system that accepts 3-bit input and checks if the numbers are divisible by 3 or 5 (Architectures from COA)
Constant

iv) Converting a ϵ-NFA to DFA (ECLOSES from POCD)
Exponential

1b Write an algorithm for the supplied description

ALGORITHM CheckBipartite(G[V, E])
Input: A graph G [V, E]
Output: Returns 1 if the graph is bipartite, 0 otherwise
Description: Checks if the G is Bipartite using **DFS** Traversal

06 marks
CLO3
L3

Logic:
- Use two colors to color the nodes
- Call the DFS function with node u
- If the node is not yet visited, assign a color and call DFS with all the nodes connected to u.
- Keep flipping between the colors during the calls
- Until all the nodes are visited, if any adjacent nodes are same colored, the graph is not bipartite.

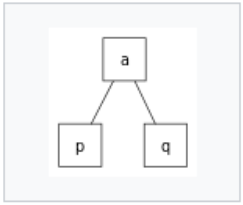
ALGORITHM CheckBipartite(G[V, E])
Input: A graph G [V, E]
Output: Returns 1 if the graph is bipartite, 0 otherwise
Description: Checks if the G is Bipartite using DFS Traversal
bipartite ← true
dfs(G, source, color1)

ALGORITHM DFSforBipartite(G[V, E], source, color)
Input: A graph G [V, E] and source
Output: Sets bipartite true if the graph is, false otherwise
Description: Traverses the tree using DFS and 2-coloring
Mark source as visited
source ← color
for i ← 0 to V do
  if there is an edge between source and i
    if i_{color} = color
    bipartite ← false
    return
  if i ≠ visited
    if color = color1
      DFSforBipartite(G, i, color2)
    else
      DFSforBipartite(G, i, color1)

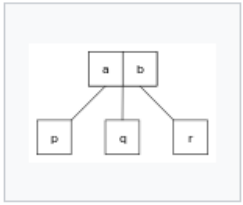
Alternate Logic:
Graph is 2-colorable if and only if there is no back edge connecting two vertices both on odd levels or both on even levels. DFS traversal needs to verify this property. Traversal can mark vertices as even or odd when it reaches them for the firsttime.

1c Answer the following questions:

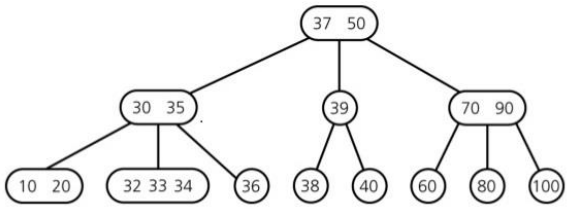
i) In a 2-3 tree, the 2-node and 3-node are as shown beside.
How does a **4-node** look like in **2-3-4 tree**?



2-node



3-node

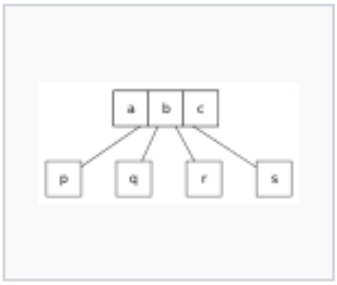


Is given beside tree a **2-3-4 tree?**

10 marks
CLO2
L2

2*5 = 10

Structure of a 4-node:

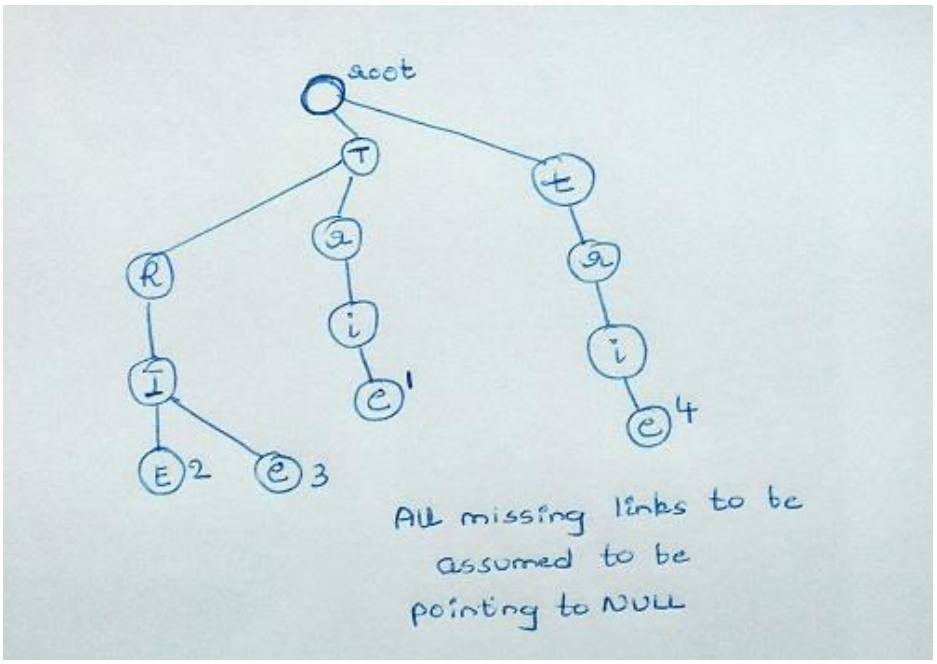


4-node

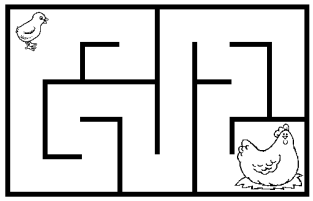
Yes. Given Tree is a 2-3-4 tree.

ii) Construct a Trie for given data by assuming suitable alphabet set:
(Trie, 1) (TRIE, 2) (TRIe, 3) (trie, 4)

The constructed trie can be seen below. The captial alphabets are assigned links first and than the small.

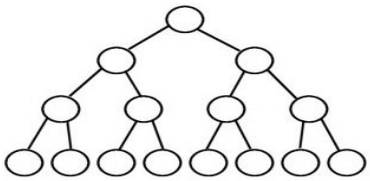


iii) Help the chicken to find its mother. Nopes, not like a primary school kid but like a graph traversal expert. (explain your approach)

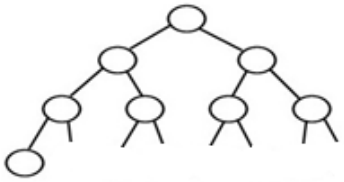


The process can be explained either using backtracking, dfs or bfs.

iv) If Thanos snaps his fingers to an almost complete binary tree given below, what would the resulting tree look like? [History: Every Thanos snap kills half of the universe population]

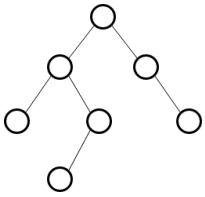


After Snap:



v) What is the maximum height of any AVL-tree with seven nodes? Assume that the height of a tree with a single node is 0.

Maximum Height = 3. The Sample can be seen below:



QUESTION 02

2a Answer the following:

i) The cities become nodes, and flood effected people need evacuation swiftly. Which method would be optimal: DFS or BFS?

BFS

ii) The Cambodian character set has 70 alphabets. What's the disadvantage if they opt to trie?

Nothing really. Space in Trie is a trade-off over time.

04 marks
CLO2
L2

iii) The minimum height and number of nodes in a binary tree are connected by $\log_2 n$. Explain.

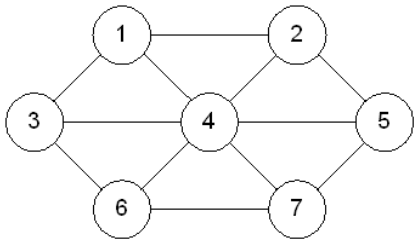
To make this height minimum, the tree most be fully saturated (except for the last tier) i.e. if a specific tier has nodes with children, then all nodes on the parent tier must have two children. This can be expressed in geometric series as:

$$\text{nodes} = 1 + 2 + 2^2 + 2^3 + \ldots + 2^{\text{depth}} = \sum_{k=0}^{\text{depth}} 2^k = \frac{1 - 2^{\text{depth}+1}}{1 - 2}.$$

On re-arranging and solving, we have:

$$\begin{aligned} \text{nodes} + 1 &= 2^{\text{depth}+1} \\ \log_2(\text{nodes} + 1) &= \log_2(2^{\text{depth}+1}) = \text{depth} + 1 \\ \log_2(\text{nodes} + 1) - 1 &= \text{depth}. \end{aligned}$$

iv) Perform DFS for the graph below with source as vertex 4:



DFS Traversal: 4 1 2 5 7 6 3

2b Explain the hash algorithm given below:

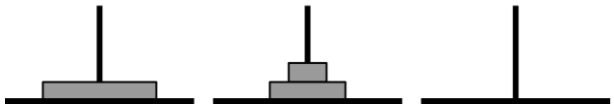
```
HASH-INSERT( T[0…m], k )
 i = 0
repeat
   j = hash(k, i)
   if T[j] == NIL
      T[j] = k
      return j
   else
      i = i + 1
until i == m
error "hash table overflow"
```

The algorithm is the process of linear probing/progressive overflow collision resolution technique.
It searches for next available position to insert during collision.

06 marks
CLO2
L1

2c Answer the following questions:

i) Given below is a snapshot of disk movement of 3 disk Towers of Hanoi. How many best case and worst case disk moments were made to reach the given state?



Best Case = 3 disk movements

Worst Case = 5 disk movements (solution target disk movement)

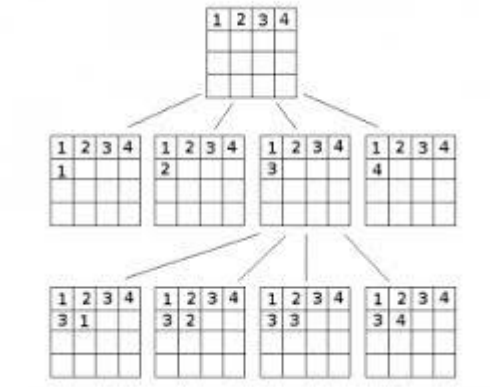
10 marks
CLO4
L3

2*5 = 10

ii) For the given sudoku grid, demonstrate your understanding of how it can be solved using backtracking. Demonstrate for any one case with 2-3 steps.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Any sample cells can be taken (full or 3x3) and demonstrated the following way:



iii) Consider the below given recursive function eff():

```
int eff(int x) {
    if(x > 1)
        return x - 1;
    else
        return (eff(eff(x+5)));
}
```

What will be the result when you call eff(0). Show the stack trace.

Stack trace for variable x:

0

x

5
*
0

x

4
0

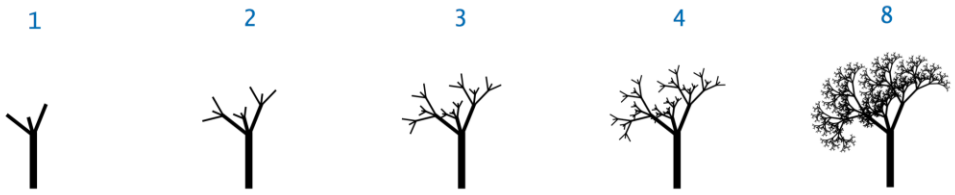
x

3

x

The final returned value is 3

iv) Consider the recursive tree shown below:



Tree-1 has 3 open branches. How many open branches will Tree-16 have?

Tree-1 - $3^1 = 3$

Tree-2 = $3^2 = 9$

Tree-8 = $3^8 = 6561$

Tree-16 = $3^{16} = 4,30,46,721$

v) The POP_ALL() operation on a stack has an efficiency of O(n), where 'n' is the number of items in the stack. Are we talking of array representation of stack or list representation of stack?

It's the same with either of the representation.

QUESTION 03

3a The keys in the sequence:

12, 18, 13, 2, 3, 23, 5 and 15

are inserted into an initially empty hash table of length 10 using the hash function

h(k) = k mod 10

using linear probing/progressive overflow.

What is the resultant hash table?

04 marks CLO1 L2

The resulting hash table will look as following:

0	
1	
2	12
3	13
4	2
5	3
6	23
7	5
8	18
9	15

3b With appropriate code snippets or pseudo-code, explain the case of deleting a node with two children from a binary search tree.

Draw appropriate diagrams while you present your case.

06 marks CLO2 L3

Finding the inorder successor:

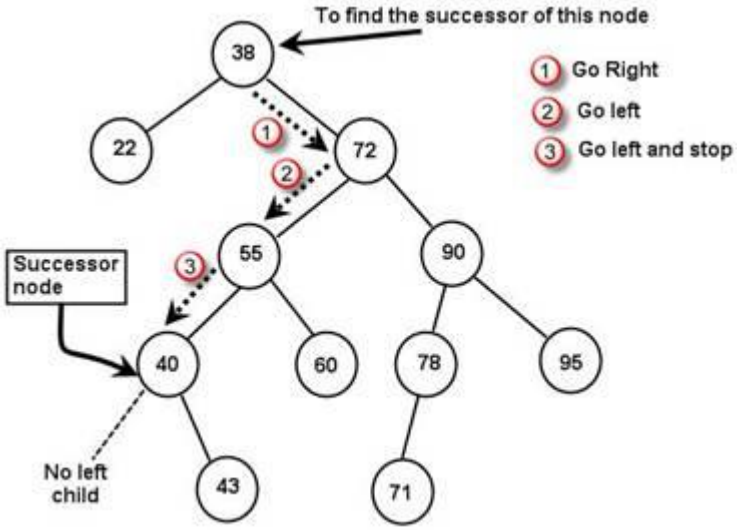
```
successor = currnode->right;
while(successor->left != NULL)
   successor = successor->left;

successor->left = currnode->left;
p = currnode->right;

Rearrangement:
if(currnode == parent ->left)
  parent->left = p;
else
  parent->right = p;

free(currnode);
```

Sample diagram if we were deleting 38:



Then, Attach 22 to left of 40. Delete 38 and make 72 as parent.

3c Consider the given algorithm:

ALGORITHM Even(k)
Input: *k*, a positive integer
Output: *k*th even natural number (the first even being *0*)
Description: Algorithm for finding the ***k***th even natural number
if *k = 1*
  return *0*;
else
  return Even(*k-1*) + *2*

10 marks
CLO4
L2

i) Identify the basic operation and set up a recurrence relation

BO - addition.

E(k) = 0 if k = 1
E(k-1) + 1 otherwise

ii) Compute the number of times the basic operation is executed

E(k) = E(k-1) + 1 → (1)
Substitute k = k - 1 in (1)
E(k-1) = E(k-2) + 1 → (2)
Substitute (2) in (1)
E(k) = E(k-2) + 2
= E(k-3) + 3
= …..
= E(k-k+1) + k – 1
= E(1) + k – 1

= k – 1
E(k) ϵ O (k)

iii) Identify the orders of growth
Linear

iv) Identify the metric to measure the size of the input
Number of bits used to represent k

v) Trace the algorithm for the input k = 4
Even(4)
|------ Even(3) + 2
|------ Even(2) + 2
|------ Even(1) + 2
|------ 0
returns the value 6