Nameerah Sec-CSB 19029001100102

Date:
Page No.:

I understand that this examination is for my learning and personal development. Hence, I pledge on my honor that I shall not give or receive any unauthorised assistance on this assignment.

Nameerah Hussain.

Section-A

Q.1

(a)

Ans: Expression: A - B/C + D*E + F

(A - B/C + D*E + F)

Character	Stack	Postfix
(	(	
A	(	A
-	(-	A
B	(-	AB
/	(-/	AB
C	(-/	ABC
+	(-+	ABC/
D	(-+	ABC/D
*	(-+*	ABC/D
E	(-+*	ABC/DE

Nameerah Sec-CSB 19029001001O2

Date:
Page No.:

+	(-++	ABC/DE*
F	(-++	ABC/DE*F
)	(	ABC/DE*F++-

(b)

⇒: The maximum number of nodes present in a binary tree of height h is $2^h - 1$.

(c)

⇒: Worst case complexity of Quick sort is $O(n^2)$ or $O(n^2)$, where n is the number of elements.

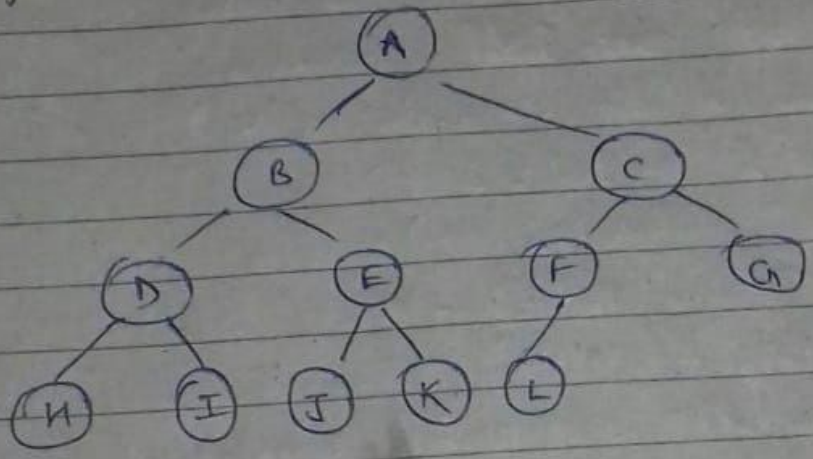
Best case complexity of Insertion sort is $O(n)$.

(d)

⇒: A complete binary tree is a binary tree in which every level, except possibly the last, is completely filled, and all nodes are as far left as possible.

Date :
Page No. :

Example of a complete binary tree is :-



(e)
⇒ : A graph becomes a tree when there is no cycle in a graph. A tree will not contain a cycle.
A tree is an undirected graph that satisfies any of the following equivalent conditions :-

* Graph is connected and acyclic (contains no cycle, as a tree is acyclic)
* Graph is connected, but would become disconnected if any single edge is removed from the graph.
* It can be checked, if the graph is connected and has (v-1) edges ; where 'v' is the no. of vertices in the graph, It would be a tree.

Nameerah Sec-CSB 1902900100102

Date:
Page No.:

ch)

A: For finding maximum height, the nodes should be minimum at each level. Assuming height as 2, minimum number of nodes required.

$N(h) = N(h-1) + N(h-2) + 1$

$\Rightarrow N(2) = N(2-1) + N(2-2) + 1$

$N(2) = N(1) + N(0) + 1$

$\Rightarrow 2 + 1 + 1 = 4$

It means, height 2 is achieved using min. 4 nodes.

Now, assuming height as 3, we have

$N(3) = N(3-1) + N(3-2) + 1$

$\Rightarrow N(2) + N(1) + 1$

$\Rightarrow 4 + 2 + 1$

$= 7$

So, Therefore, using 7 nodes, we can achieve max. height as 3

cf)

A: A priority queue is an abstract data type similar to a regular queue or stack data structure in which each element additionally has a "priority" associated with it. In a priority queue, an element

Date:
Page No.:

with high priority is served before an element with low priority.

(j)

A: The worst case time complexity of searching in BST (or Binary search Tree) is O(h) where h is the height of tree.

Nameerah Sec-CSB 19029001001012

Date :
Page No. :

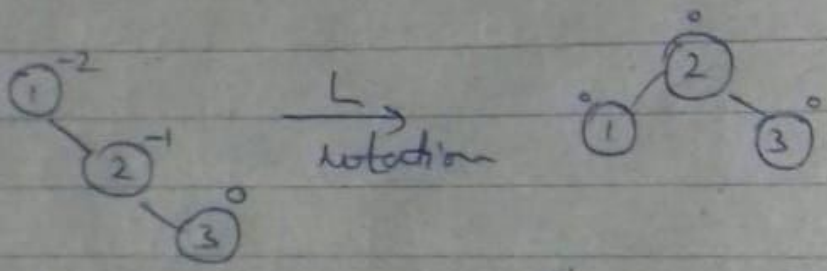
Section-B

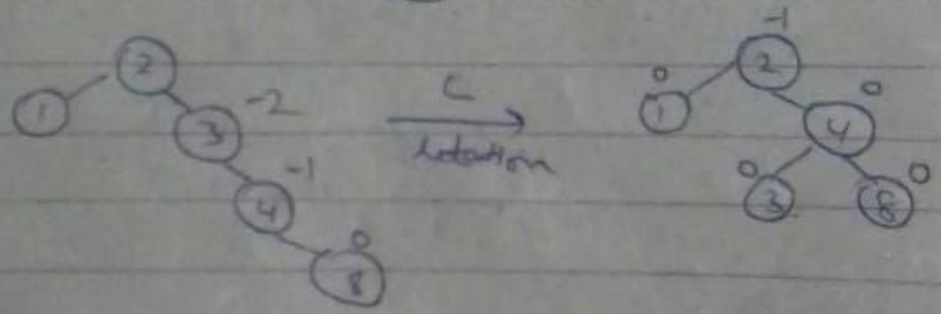
Q2

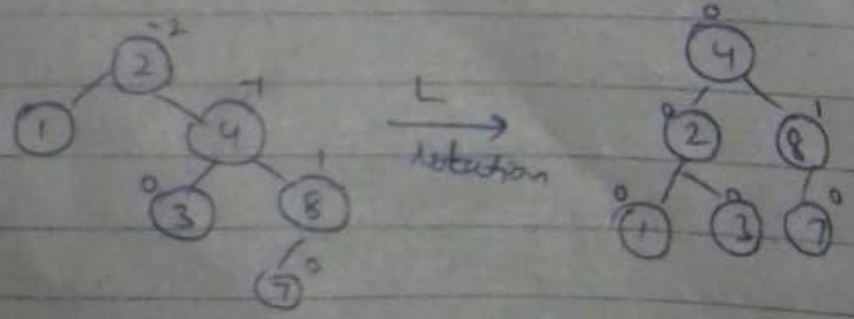
(a)

⇒ · AVL tree is a binary search tree (or BST) in which the difference of heights of left and right subtrees of any node is less than or equal to one. It is basically, a self balancing binary tree.

1, 2, 3, 4, 8, 7, 6, 5, 11, 10, 12

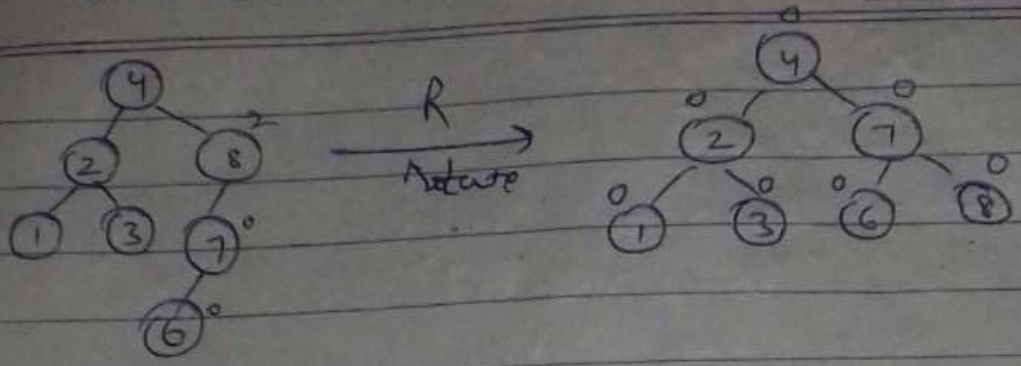


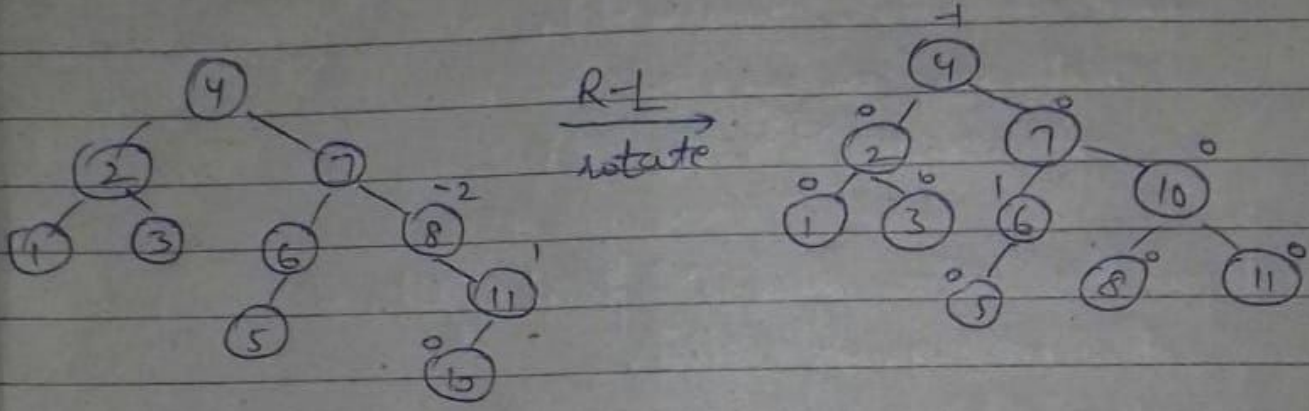


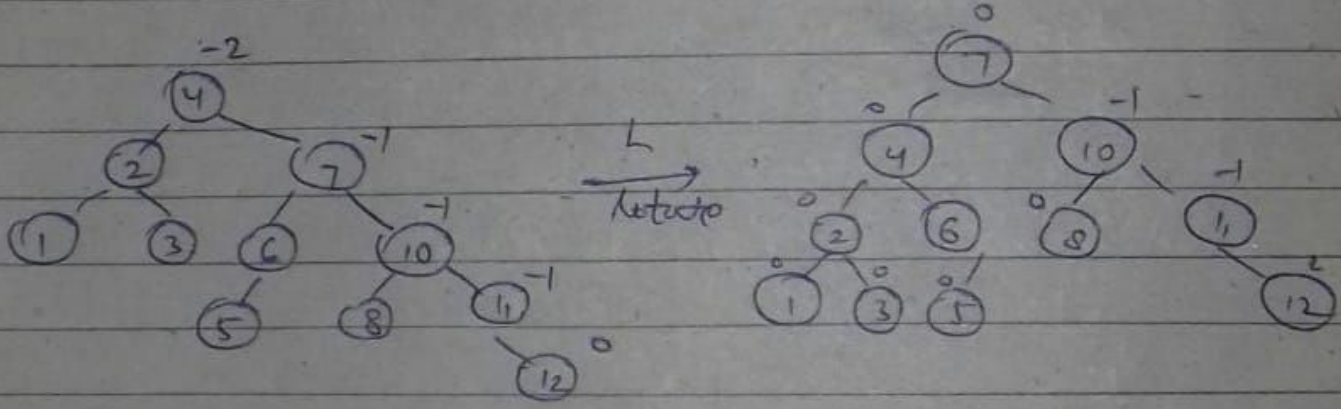


Name: Cse 1902900100102

Date:
Page No.:







(d) Ans: Quick sort program

```
#include <stdio.h>
void quicksort (int array[], int left, int right, int size);
void printarray (int array [], int size);
int main ()
```

Date:
Page No.:

```
    scanf("%d", &size);
    int array[size+1];
    for(int i=0; i<size; i++) {
        scanf("%d", &array[i]);
    }
    quicksort(array, 0, size-1);
    printarray(array, size);
    return 0;
}
int partition(int array[], int left, int right, int size)
    int pivot = array[left];
    int i = left+1;
    int j = right;
    int temp;
    while(i<=j) {
    while(array[i]<=pivot && i<size) {
        i++;
    }
    while(array[j]>pivot) {
        j--;
    }
    if(i<j) {
        temp=array[i];
        array[i] = array[j];
        array[j] = temp;
    }
```

Date :
Page No. :

```
    }
    temp = array[i-1];
    array[i-1] = pivot;
    array[left] = temp;
    return i-1;
}
void quicksort(int array[], int left, int right, int size){
    if(left<right){
    int pi = partition(array, left, right, size);
    quicksort(array, left, pi-1, size);
    quicksort(array, pi+1, right, size);
    }
}
void printarray(int array[], int size){
    for(int i=0; i<size; i++){
        printf("%d", array[i]);
    }
}
```

(e)

Ans: Dequeue function

```
#include <stdio.h>
#include <stdlib.h>

struct queue {
```

Nameerah Sec-CSB 19029001001O2

Date:
Page No.:

Class

```
    int data;
    struct queue * next;
}
```

Initial value

```
struct queue * front = NULL;
struct queue * rear = NULL;
```

Dequeue function

```
void dequeue () {
if (front == NULL && rear == NULL) {
    printf ("UNDERFLOW \n");
    return;
}
else {
    front = front → next;
}
}
```

Section-C

Q.4

(a)

⇒:

```
void main ()
{
int A[20], i, n, temp
```

O(1) ← printf ("Enter the number of term in the array");
O(1) ← scanf ("%d", &n);
(1) ← printf ("Enter the list items");

Date:
Page No.:

```
O(n) ← for (i=0; i<=n; i++)
       {
       scanf("%d", &A[i]);
       }
       // Bubble sort
let S6 ← for (i=1; i<=n-1; i++)
       {
S7 ← for (j=0; j<=n-i-1; j++)
       {
S8 ← if A[j] > A[j+1]
       {
       temp = A[j];
       A[j] = A[j+1];
       A[j+1] = temp;
       }
       }
       }
O(1) ← printf("Sorted list :\n");
O(n) ← for (i=0; i<=n-1; i++)
       {
O(n) ← printf("%d", A[i]);
       }
       }// End main()
```

Nameerah Sec-CSB 19029001000102

Date:
Page No.:

S_6	S_7	S_8
for i=1 (1 time)	(n-2)+1 = n times	(n-1) times
for i=2 (1 time)	(n-2)+1 = (n-1) times	(n-2) times
for i=3 (1 time)	(n-3)+1 = (n-2) times	(n-3) times
⋮	⋮	⋮
for i=n-1 (1 time)	1	0
for i=n (1 time)	0 times	0 times.

* Total freq. of $S_6 = 1+1+1+\dots+1 = n$ times $= n$
Total time taken $= n \cdot t_6$ (where t_6 is execution time of S_6)
for large value of n $T \propto n$ or $T = O(n)$

* Total freq. of $S_7 = n+(n-1)+(n-2)+\dots+2+1$
$= 1+2+3+\dots+n = \frac{n(n+1)}{2}$

$\frac{n^2+n}{2}$.

If t_7 is the time required to execute S_7

Date :
Page No. :

then, total time required = $\left(\frac{n^2+n}{2}\right) t_7$

for larger value of n. $T \propto n^2$
Time complexity $\boxed{T = O(n^2)}$

Total frequency of $S_8 = (n-1)+(n-2)+\cdots+2+1$

$= (n-1)+((n-1)-1)+((n-1)-2)+\cdots+2+1$

$= \frac{(n-1)((n-1)+1)}{2}$

$= \frac{(n-1)^2+(n-1)}{2}$

$= \frac{n^2+1-2n+n-1}{2} = \frac{n^2+n-2n}{2}$

$= \frac{n^2-n}{2}$

$T \propto n^2$; $\boxed{T = O(n^2)}$

Therefore, overall time complexity.

$T = \max(O(n), O(n^4), O(n^2))$
$S_6 \quad S_7 \quad S_8$

$\boxed{T = O(n^2)}$

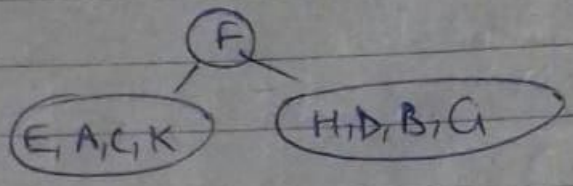
Nameerah Sec-CSB 19029001 00102

Date:
Page No.:

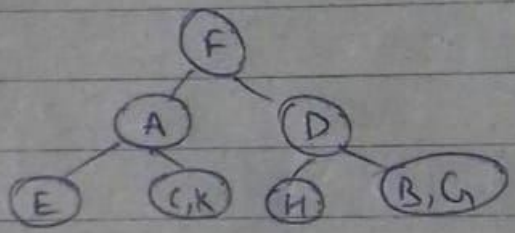
Q.7
(a)

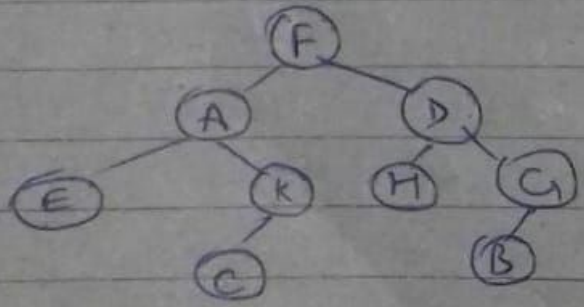
⇒: Inorder = E, A, C, K, F, H, D, B, G
Preorder = F, A, E, K, C, D, H, G, B

From pre order traversal (F) is the root node.
So, (E, A, C, K) is left subtree and (H, D, B, G) is right subtree



Now,





Date:
Page No.:

Q.6
(b)

⇒: Binary Search Tree (or BST) is a binary tree in which almost two children are possible of a node, but with slight change, every child of left subtree must be smaller than the parent node and every right child must be greater than the parent node.

For Example [Insertion]
We want to insert element in the tree
[c, e, d, b, a]

Step1 If BST is empty, Insert first element (root) as it is.

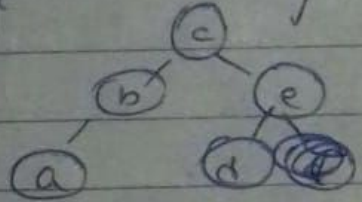
(c)

Step2 Now, while inserting next element check if it's greater or smaller than the root element

(c)
(e)

[In this case e > c]

Step3 Follow step 2 for every element.



[Required Tree]

Nameerah Sec-CSB 1902900100102

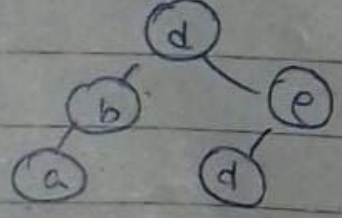
Date:
Page No.:

[Deletion in BST]

Deletion has three condition:-
(i) If the node to be deleted has 0 child.
(ii) If the node to be deleted has 1 child.
(iii) If the node to be deleted has 2 child.

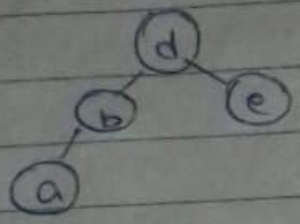
→ (i)st case i.e., if the node to be deleted has 0 child
In case one, simply free the deleted node.
In second case, connect previous of (to be deleted) node to the next of deleted node.
In third ~~case~~, and final case, we can take inorder predecessor or inorder successor and replace it with the deleted node and remove it using first case.

For Example:
From previous tree, let us suppose we have to delete node (c).
Find inorder predecessor or successor here, in this case inorder successor is (d) to replacing it with deleted node.



Date:
Page No.:

Now, removing it using first case by breaking the link of (e)-(d)



[Required Tree]

Q.3

(b)

Ans: Recursion: When a function calls itself with a slight change in arguments it is called recursion. Recursion consists of Base case. When recursion to the smallest valid argument then the base case executes and recursion terminates mean it doesn't calls itself anymore.

→ Non-recursive program to find factorial.

```
#include <stdio.h>
int main(){
  int number;
    int fact = 1;
scanf("%d", &number);
if(number==0)  printf("1
else
{
```

Date :
Page No. :

```
for(int i=number; i>=2; i--){
    fact = fact * i;
}
printf("%d\n", fact);}
    return 0;
}
```

Output : 5
120
: 0
1

→ Recursive Program to find factorial.

```
#include <stdio.h>
int factorial (int n){
if(n==0||n==1){
    return 1;
}
return n*factorial(n-1);
}
int main(){
  int n;
 scanf("%d", &n);
int ans = factorial(n);
printf("%d\n", ans);
return 0;
}
```

Base Case

function calling itself.

Date:
Page No.:

Q.5

(a)

Ans: Hash function in data structure is a function which is used to store a value in hash table in constant time (O(1)) and retrieve that value in constant (O(1)) using key. Key is a unique identity of a value in Hash Table.

Different Types of Hashing Techniques:-

* Division Method

In this method, the value is dependent on module function.

Let us suppose size of HashTable = 10 and element to be used in Hashing are:

42, 43, 49, 68, 94.

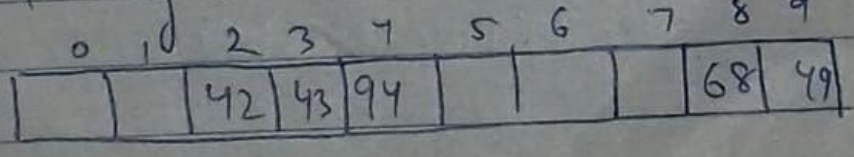
Hash key = 42 % 10 = 2

Hash key = 43 % 10 = 3

Hash key = 49 % 10 = 9

Hash key = 68 % 10 = 8

Hash key = 94 % 10 = 4.



Hash Table →

Nameerah CSB 19029001000102

Date:
Page No.:

* Mid Square method

In this method, the middle number of the square of element is taken to be the key of value.

For Example: Element to be inserted are 91, 88, 18, 10 and size of Hash Table is 100.

Hash key = 91 * 91 = 8281 = 28
Hash key = 88 * 88 = 7744 = 74
Hash key = 18 * 18 = 324 = 2
Hash key = 10 * 10 = 100 = 0

* Digit folding methods

→ In this method, a large set of elements is break into smaller element and a Hash key is generated using single mathematics.

Example: Element to be placed are
235 766 23, 346 877 34

Hash key = 235 + 766 + 23 = 1024
" " = 346 + 877 + 34 = 1257