Ques 1

Way 1

a) hashmap
key: character
value: frequency

b) map[ch] = map[ch] + 1

c) loop over all keys and find highest frequency value.

Way 2

array
on ascii index store the frequency of character. } 2

array[ch] = array[ch] + 1 } 3

loop over entire array and find highest value, then print character corresponding to the index. } 3

Ques 2

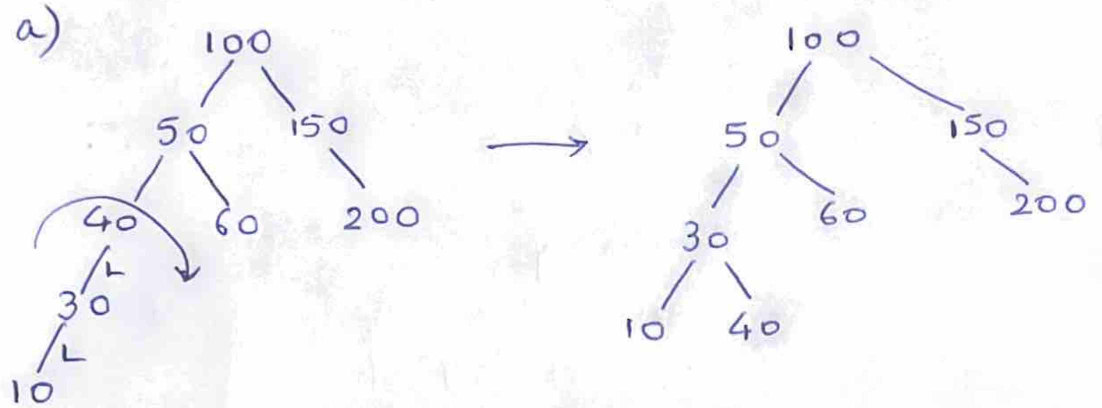
```
void createEmptyStack()
{
    head = NULL;
}
```
} 2

```
void push(int item)
{
    Node *nn = new Node(item);

    if(head == NULL)
        head = nn;
    else
    {
        nn->next = head;
        head->prev = nn;
        head = nn;
    }
}
```
} 3

```
void pop()
{
    if(head == NULL)
        return;

    if(head->next == NULL &&
       head->prev == NULL)
        head = NULL;
    else
    {
        Node *oldhead = head;
        head = head->next;
        head->prev = NULL;
        delete(oldhead);
    }
}
```
} 3

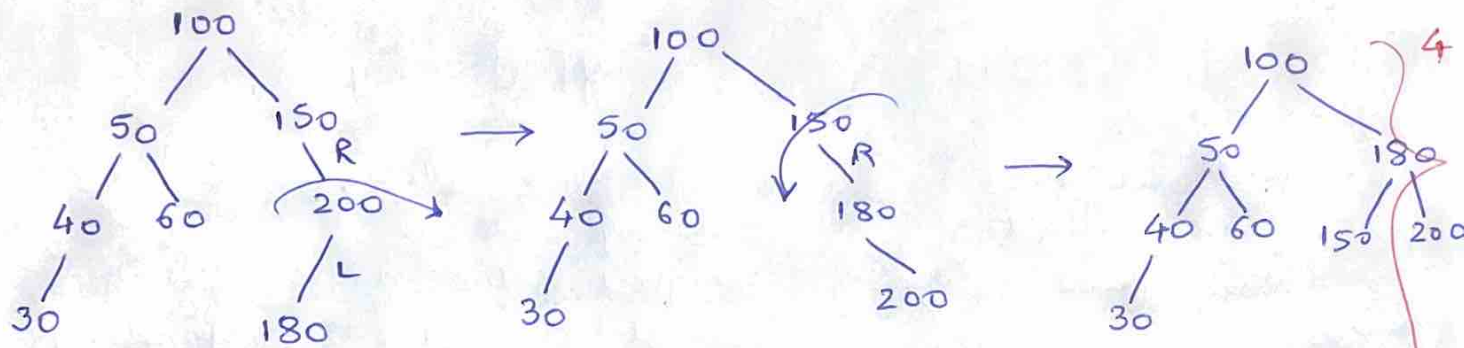
Ques 3

a)



4

b)



4

Ques 4

```
bool DFS(int src, int dst, bool *visited, string path)
{
    visited[src] = true;
    if (src == dst)
    {
        cout << path << endl;
        return true;
    }

    for (j=0; j<V; j++)
    {
        if(matrix[src][j] != 0 && !visited[j])
        {
            if(DFS(j, dst, visited, path + to_string(j)) == true)
                return true;

        }
    }
    return false;
}
```

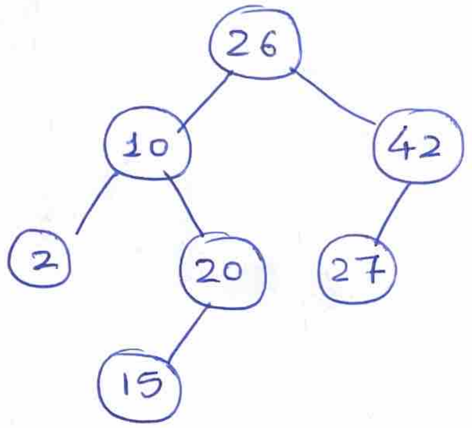
Written something: 1
not clean 4.
rec written with vars.
printed always: 6
not done pop_back: 6
iterative k (point all k not finished): 4

8

```
bool DFS (int src, int dst)
{
    bool visited [V];
    for (int i=0; i<V; i++)
        visited[i] = false;

    return DFS (src, dst, visited, "" + to_string(src));

}
```

Ques 5

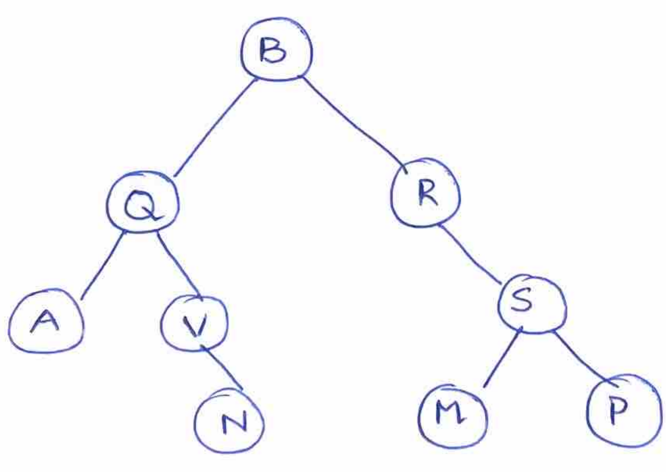
a)



4

b) Inorder: AQVNBRMSP   LNR

Pre-Order: BQAVNRSMP   NLR



4

Only code written: 1

rest part right: 1

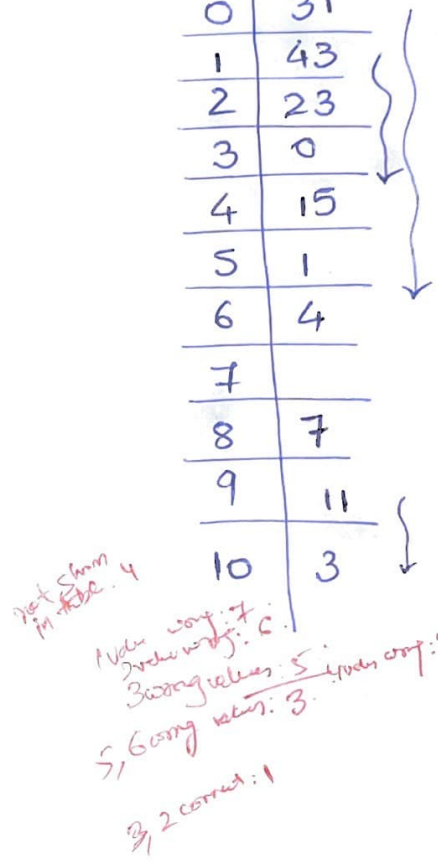
Ques 6

8

(4)

key	(key+7)*(key+7)	x/16	x+key	x%11
43	2500	156	199	1
23	900	56	79	2
1	64	4	5	5
0	49	3	3	3
15	484	30	45	1
31	1444	90	121	0
4	121	7	11	0
7	196	12	19	8
11	324	20	31	9
3	100	6	9	9

0	31
1	43
2	23
3	0
4	15
5	1
6	4
7	
8	7
9	11
10	3



Ques 7

a) A node has multiple keys and children. } 4

b). Order = max no. of children

Order = n

max children = n	min children = $\lceil \frac{n}{2} \rceil$
max keys = n-1	min keys = $\lceil \frac{n}{2} \rceil - 1$

} 4