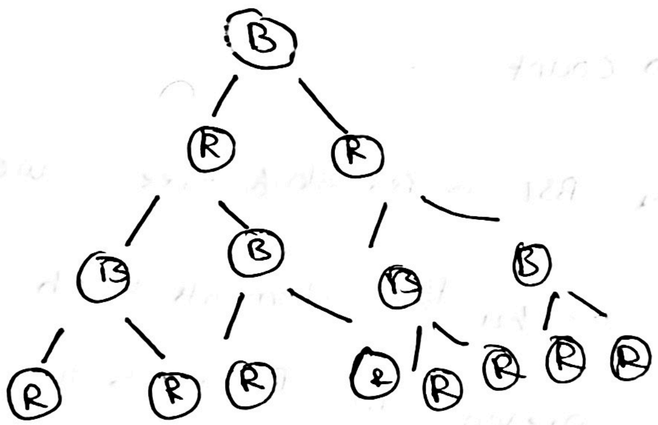
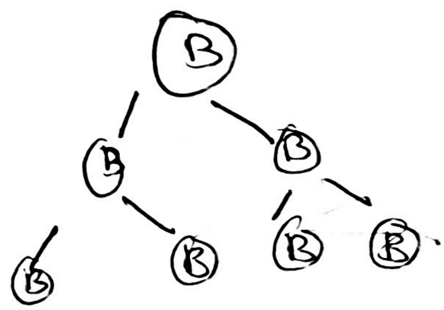
Assignment 01

1) a) The largest number of possible internal nodes for a Red Black tree with black height k is $2^{2k}-1$. This is obtained when the tree is perfectly balanced and black node has two red nodes as child nodes. $(2^{2k}-1)$



The smallest number of possible internal nodes posible in Red-Black tree with black height k is 2^k-1. It is obtained when Red Black tree has all black nodes as children. (2^k-1)



2) a)

Pseudocode: (Recursive Program)

```
BSTInRange (root, a, b)

    if root is null
        then "False"

    if root.value >= a and root.value <= b
        then "True"

    if root.value < a
        BSTInRange (root.right, a, b)

    else
        BSTInRange (root.left, a, b)
```

Given, that the BST is balanced we are traversing over BST. The time Complexity of our algorithm will be O(h) (or) O(logn), where h is the height of Tree and n are the number of nodes in BST.

3) a)

Pseudocode :-

```
COUNT (D, x)

Count = 0
root = D.root

while    root ≠ null

    if    root.value > X
          Count = count + root.right.size
          root = root.left

    else
          root = root.right

return count
```

Given, that the BST is red Black tree we are traversing and checking whether the elements which are greater than n are present in BST. As we know that the complexity of traversing in BST is O(logn) (or) O(h) where h is height and n is the number of nodes.

4)a)

```
Primality (root, x)

    if n<=1
        return 0

    for i=2 to √n +1
        if n%i==0
            return 0

    if TREE-SEARCH(root, x) != NIL
        return 1

    TREE-INSERT(root, x)
```

Note:-

TREE-SEARCH & TREE-INSERT are functions referred from textbook.