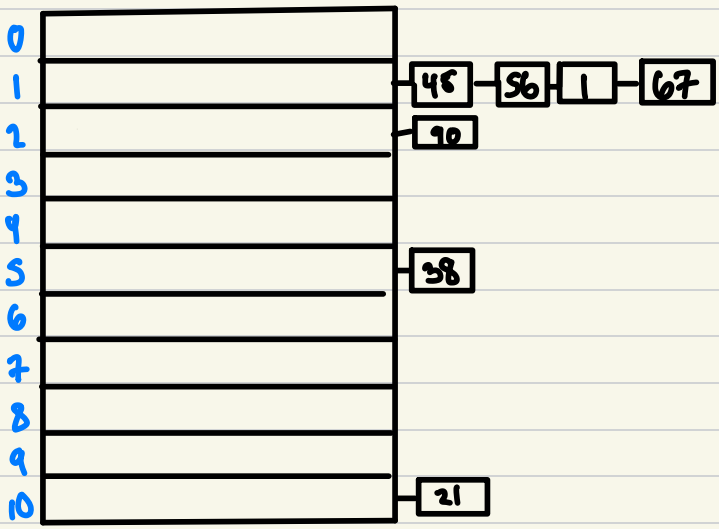
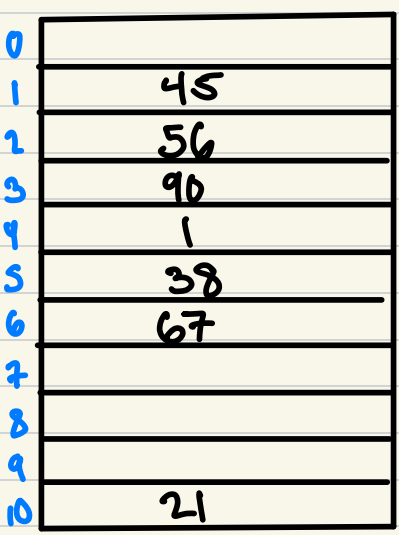
2210 Assignment #3

21 mod 11 = 10
45 mod 11 = 1
56 mod 11 = 1
90 mod 11 = 2
1 mod 11 = 1
38 mod 11 = 5
67 mod 11 = 1

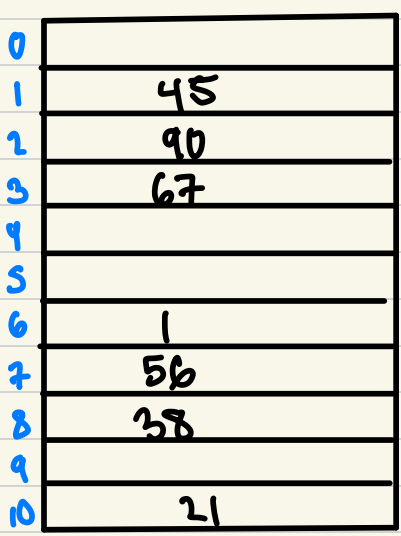
1) i)



ii)



iii)



ROUGH WORK: $h'(56) = 7 - (56 \bmod 7 + 1) = 6$

$h'(1) = 7 - (1 \bmod 7 + 1) = 5$ // $h'(38) = 7 - (38 \bmod 7 + 1)$

$h'(67) = 7 - (4 + 1) = 2$ $= 7 - 4 = 3$

2a) 1, 5, 1, 6, 7, 3, 7, 3

b) 1, 5, 7, 6, 3, 1, 3, 7

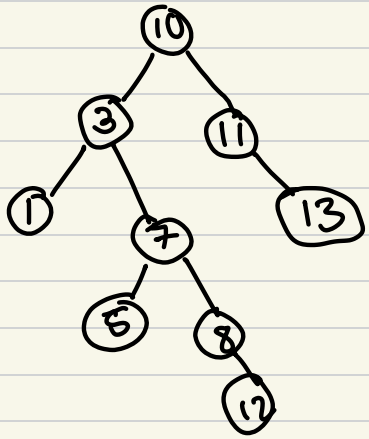
c) 1, 7, 8, 6, 5, 3, 7, 1

3) i)

```
INITIALIZE index = 0

FUNCTION buildBST(preorder, lower, upper):
    IF index >= LENGTH(preorder):
        RETURN NULL

    value = preorder[index]

    IF value < lower OR value > upper:
        RETURN NULL

    index = index + 1

    root = NEW Node(value)
    root.left = buildBST(preorder, lower, value)
    root.right = buildBST(preorder, value,
upper)

    RETURN root
```

ii) Each node is processed O(1) and since there are n nodes, the time complexity is O(n).

iii.)



4.)

```
FUNCTION CountPaths(root, k):
    RETURN countPathsFromNode(root, 0, k)

FUNCTION countPathsFromNode(node, currentSum, k):
    IF node IS NULL:
        RETURN 0

    # Update the current path sum
    currentSum = currentSum + node.value

    # If it's a leaf node, check if the currentSum equals k
    IF node.left IS NULL AND node.right IS NULL:
        IF currentSum == k:
            RETURN 1
        ELSE:
            RETURN 0

    # Recursively count paths in left and right subtrees
    leftPaths = countPathsFromNode(node.left, currentSum, k)
    rightPaths = countPathsFromNode(node.right, currentSum, k)

    RETURN leftPaths + rightPaths
```

ii) There are also n number of nodes and each node is O(1) so the time complexity is O(n).