Assignment 3 - Solutions

Q1 a) A possible solution:

```
InsertionSort (arr):
    int [] sorted_a;
    int j ← 0;
    int k ← arr.size-1;
    for (int i=0; i<arr.size; i++):
        if (arr[i] == 0)
            sorted_a[j] ← arr[i];
            j++;
        else
            sorted_a[k] ← arr[i];
            k--;
    return sorted_a;
```

Another possible solution:

```
InsertionSort (arr):
    int[] sorted_a;
    int j ← 0;
    for(int i=0; i<arr.size; i++):
        if (arr[i] == 0)
            sorted_a[j] ← arr[i];
            j++;
    while (j<arr.size):
        sorted_a[j] ← 1;
        j++;
    return sorted_a;
```

b) A possible solution:

```
InsertionSort(arr):
    int[] sorted ← arr;
    int i ← 0;
    int j ← arr.size - 1;
    while (j > i):
        if (arr[i] > arr[j])
            swap(sorted, i, j);
            i++;
            j--;
        else if (arr[j] > arr[i])
            i++;
            j--;
        else if (arr[j] == 0)
            i++;
        else if (arr[i] == 1)
            j--;
```

```
swap(arr, i, j):
    int temp ← arr[i]
    arr[i] ← arr[j];
    arr[j] ← temp;
```

Q_2 We use a Stack to Traverse the Tree non_recursively. The algorithm is shown below, where T is a non_empty binary tree with n_nodes

Algorithm 1, A possible Solution

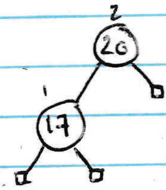
```
let node be the root of T;
Let S    be an initially empty Stack;
 flag ← True
 while flag do
     if node ≠ None then
         S.push(node);
         node ← node.left;
     else
       if !(S.empty()) then
           node ← S.pop();
           print(node.item);
           node ← node.right;
       else
        flag ← false
```

Each node of T is pushed into S once and will be eventually removed from S.

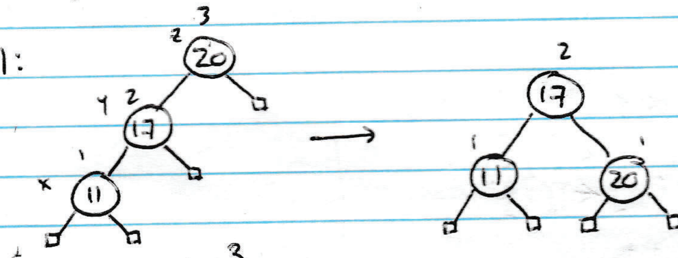
Since each statement inside the while_loop Can be done in $O(1)$ the algorithm runs in $O(n)$ overall time

Q3a) Insert 20:

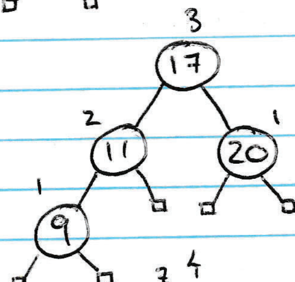
Insert 17:



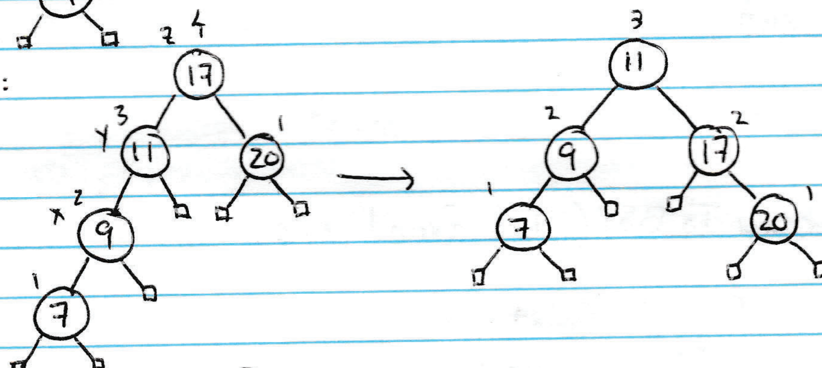
Insert 11:



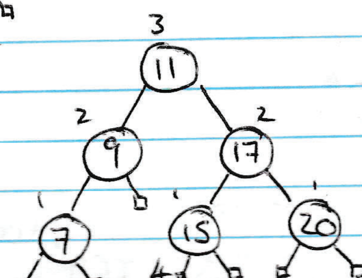
Insert 9:



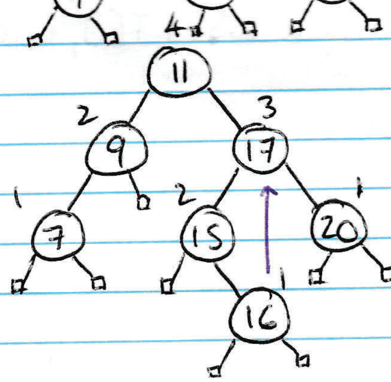
Insert 7:



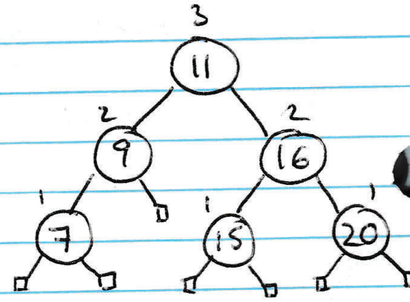
Insert 15:



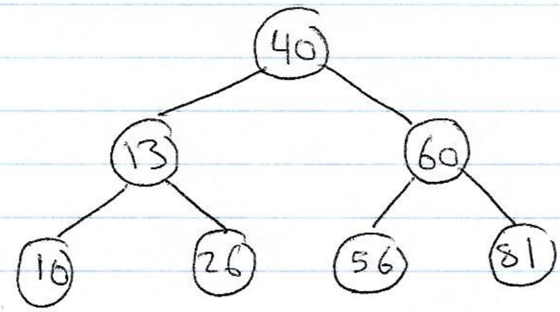
Insert 16:



b)

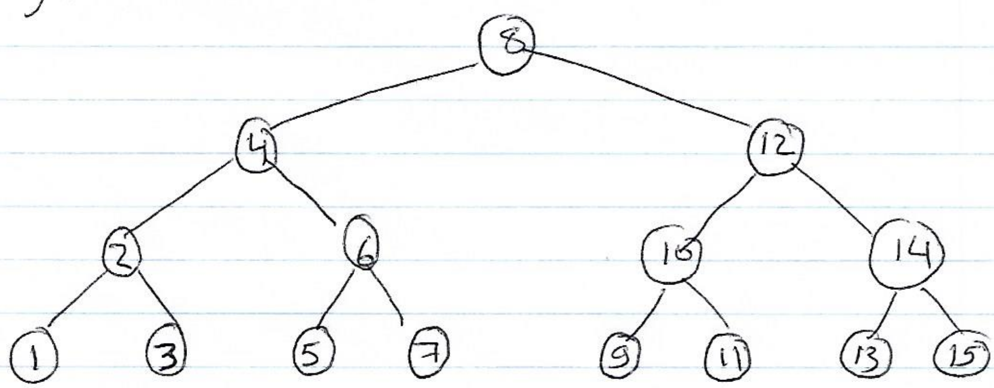


Q4

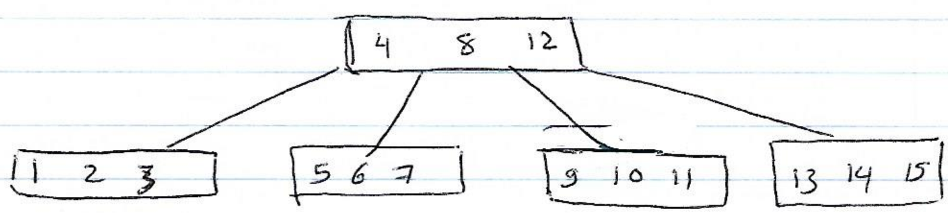


Q5

max height

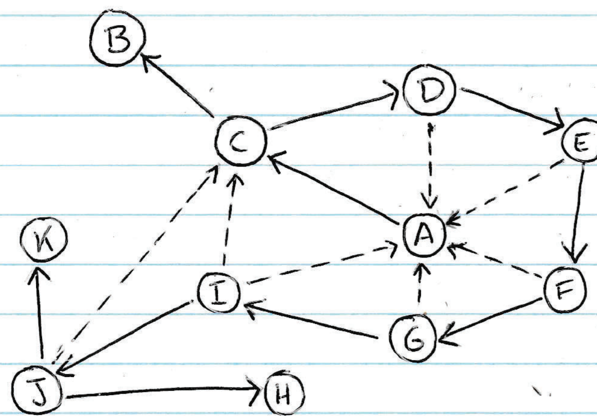


min height



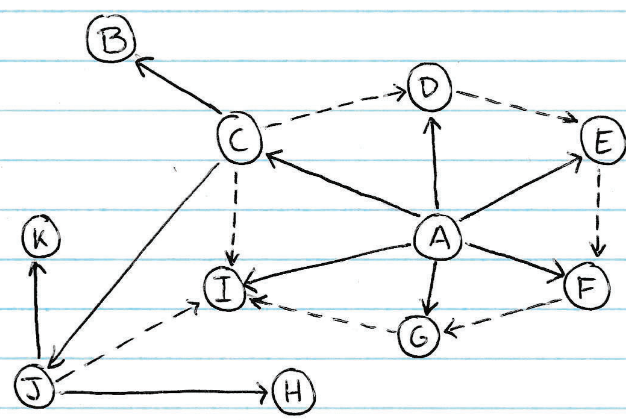
Q6 b) A, C, B, D, E, F, G, I, J, H, K

DFS



a) A, C, D, E, F, G, I, B, J, H, K

BFS



c) A, G, I, D, F, E, J, C, H, K, B

Dijkstra's