Data Structures and Fundamentals of Programming

Problem #1

In C++ implement a generic **singly-linked-list** class, called `Queue<item>`, which uses dynamic memory allocation. `item` is the type of data stored in the **queue**. This should implement the **FIFO** (First-In-First-Out) queue ADT(abstract data type). The queue should look something like the following:

$$HOQ \rightarrow X_1 \rightarrow X_2 \rightarrow \ldots \rightarrow X_n$$

where X_1 is the node in the front of the queue and X_n is at the end of the queue, HOQ is the head of queue.

Along with the class definition(s), you must implement the following methods for `Queue`:

- `Queue()` - Default constructor
- `~Queue()` - Destructor
- `insert(item)` –add a new item to the queue
- `item remove()` – removes an item from the queue.
- `item getmax()` – return the maximum item inside the queue. `MIN_ITEM` and `MAX_ITEM` are the minimum and maximum value of all the possible `items`, respectively. You can directly use an existing function `max(item1,item2)` to compute the maximal one of the two items `item1` and `item2`.

Note: Your implementation can **NOT** use STL or any other libraries (standard or otherwise).

Problem #2

A "binary search tree" (BST) is a type of binary tree where the nodes are arranged in order: for each node, all elements in its left subtree are less-or-equal to the node (<=), and all the elements in its right subtree are greater than the node (>).

Given a binary search tree. Each node is defined by a struct as

```
struct TreeNode
{
      Treenode *left;
      Treenode *right;
      ElementType data;
}
```

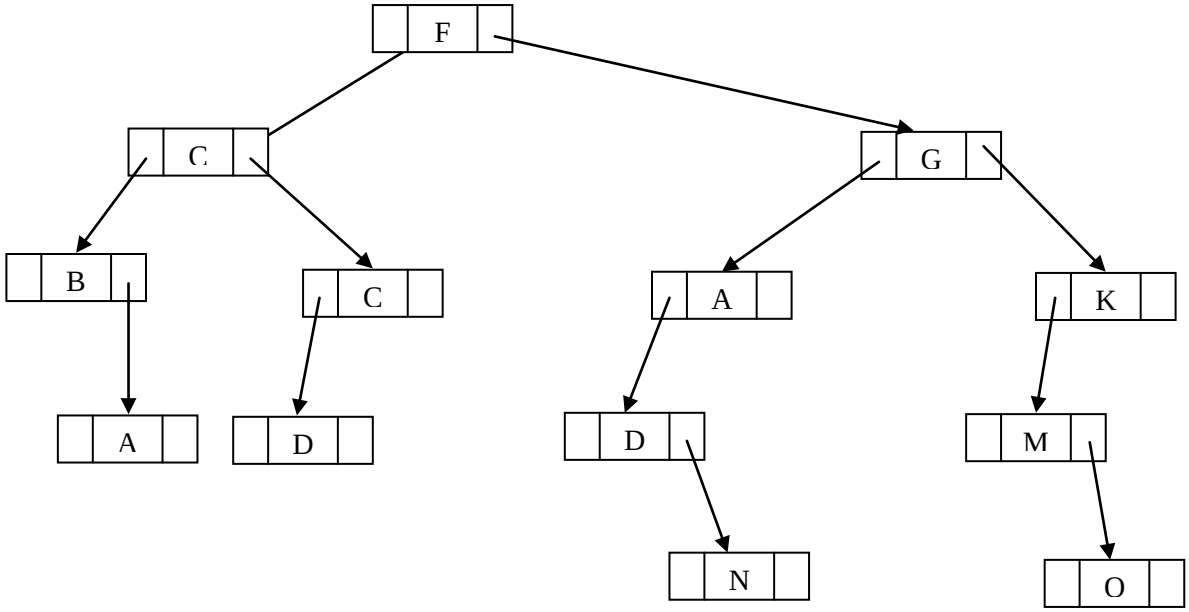
Assume there is a function, `output(data)`, can output the element data.

(1) Write a function to output all the data nodes of a given tree in an increasing order (from small nodes to large nodes).

(2) Write a function to output the minimum and maximum data values from a given tree. `MIN_DATA` and `MAX_DATA` are the minimum and maximum value of all the possible `data` values.

Problem #3

A) Give the Preorder, Postorder, and Inorder traversals of the tree below:



B) Create a binary tree given the Preorder Traversal of this tree as AEFBGCFD, and the Inorder Traversal of this tree as EFAGBFCD.

Draw this binary tree, and write the Postorder traversal of this tree.