**Explanation of Test Case (input1.txt)**

Hello,

This document explains the first test case (input1.txt) for the assignment.

**Overview**

The program uses two queues:

1. **Command Queue (Priority Queue):** Stores commands with their assigned priority values.
2. **Message Queue (Regular Queue):** Stores messages and applies transformations based on commands from the priority queue.

**Initial Setup**

- Commands are inserted into the priority queue. You can implement this using a custom class or STL.
- The priority of each command is denoted within parentheses (e.g., (#)).
- The last line of the input file specifies the tree traversal method.

Pqueue = [

DECODE:[?vi#fzm2f##faf@2zvaf#zbf22nfz42fcfo424](1),

REPLACE:[?,t](1),

ADD:[@,g](1),

REPLACE:[#,s](2),

REPLACE:[v,h](3),

SWAP:[f,2](4),

REPLACE:[z, ](4),

REPLACE:[f,e](5),

REMOVE:[@](6),

REMOVE:[2](6),

REPLACE:[4,d](6),

BST:(7)

]

Queue = []

**Processing Commands**

Commands are executed sequentially from the priority queue. If a command modifies the front message in the queue, the processed message is moved to the back.

1. **DECODE:[?vi#fzm2f##faf@2zvaf#zbf22nfz42fcfo424]:**
    - Add ?vi#fzm2f##faf@2zvaf#zbf22nfz42fcfo424 to the message queue.
    - Queue = [?vi#fzm2f##faf@2zvaf#zbf22nfz42fcfo424]
2. **REPLACE:[?,t]:**
    - Replace all ? with t.
    - Queue = [tvi#fzm2f##faf@2zvaf#zbf22nfz42fcfo424]
3. **ADD:[@,g]:**
    - Insert g after each @.
    - Queue = [tvi#fzm2f##faf@g2zvaf#zbf22nfz42fcfo424]
4. **REPLACE:[#,s]:**
    - Replace # with s.
    - Queue = [tvisfzm2fssfaf@g2zvafszbf22nfz42fcfo424]
5. **REPLACE:[v,h]:**
    - Replace v with h.
    - Queue = [thisfzm2fssfaf@g2zhafszbf22nfz42fcfo424]
6. **SWAP:[f,2]:**
    - Swap occurrences of f and 2.
    - Queue = [this2zmf2ss2a2@gfzha2szb2ffn2z4f2c2o4f4]
7. **REPLACE:[z, ]:**
    - Replace z with a space.
    - Queue = [this2 mf2ss2a2@gf ha2s b2ffn2 4f2c2o4f4]
8. **REPLACE:[f,e]:**

- Replace f with e.
- Queue = [this2 me2ss2a2@ge ha2s b2een2 4e2c2o4f4]

9. **REMOVE:[@]:**
    - Remove all @.
    - Queue = [this2 me2ss2a2ge ha2s b2een2 4e2c2o4f4]
10. **REMOVE:[2]:**
    - Remove all 2.
    - Queue = [this message has been 4eco4f4]
11. **REPLACE:[4,d]:**
    - Replace 4 with d.
    - Queue = [this message has been decoded]
12. **BST:**
    - Insert the front message into a Binary Search Tree (BST), sorted by string length, and remove it from the queue.
    - Queue = []
    - BST = { this message has been decoded }
    - Since only one node exists, this message becomes the root.

**Final Output**

Using the specified inorder traversal:

**Ans1.txt:**

this message has been decoded

**Additional Notes**

- Ensure output format matches the expected answer file **exactly**.
- Remove newline (\n) and carriage return (\r) characters from input to avoid OS-related discrepancies during grading.

By following these steps, the expected output will be produced accurately.