```cpp
#include <iostream>
#include <stack>
using namespace std;

struct TreeNode {
    char data;
    TreeNode* left;
    TreeNode* right;
};

TreeNode* newNode(char data) {
    TreeNode* node = new TreeNode;
    node->data = data;
    node->left = node->right = NULL;
    return node;
}

TreeNode* constructExpressionTree(string prefix) {
    stack<TreeNode*> s;
    for (int i = prefix.length() - 1; i >= 0; i--) {
        char c = prefix[i];
        if (isdigit(c) || isalpha(c)) {
            TreeNode* node = newNode(c);
            s.push(node);
        } else {
            TreeNode* node = newNode(c);
            node->left = s.top();
            s.pop();
            node->right = s.top();
            s.pop();
            s.push(node);
        }
    }
    return s.top();
}

void postorderTraversal(TreeNode* root) {
    stack<TreeNode*> s;
    TreeNode* lastNodeVisited = NULL;

    while (!s.empty() || root != NULL) {
        if (root != NULL) {
            s.push(root);
            root = root->left;
        } else {
            TreeNode* peekNode = s.top();
            if (peekNode->right != NULL && lastNodeVisited != peekNode->right) {
                root = peekNode->right;
            } else {
                cout << peekNode->data << " ";
                lastNodeVisited = peekNode;
                s.pop();
            }
        }
    }
}

void deleteTree(TreeNode* root) {
    if (root == NULL) return;
    deleteTree(root->left);
    deleteTree(root->right);
    delete root;
}

int main() {
    int choice;
    string prefix;
    TreeNode* root = NULL;

    do {
        cout << "Menu" << endl;
        cout << "1. Enter prefix expression" << endl;
        cout << "2. Postorder traversal" << endl;
        cout << "3. Delete tree" << endl;
        cout << "4. Exit" << endl;
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1:
                cout << "Enter prefix expression: ";
                cin.ignore();
                getline(cin, prefix);
                root = constructExpressionTree(prefix);
                break;
            case 2:
                if (root == NULL)
                    cout << "Tree not constructed yet" << endl;
                else {
                    cout << "Postorder traversal of expression tree: ";
                    postorderTraversal(root);
                    cout << endl;
                }
                break;
            case 3:
                if (root == NULL)
                    cout << "Tree not constructed yet" << endl;
                else {
                    deleteTree(root);
                    root = NULL;
                    cout << "Tree deleted" << endl;
                }
                break;
            case 4:
                cout << "Exiting program" << endl;
                break;
            default:
                cout << "Invalid choice" << endl;
                break;
        }
        cout << endl;
    } while (choice != 4);

    return 0;
}
```