2019A7PS0020U Dhruv Maheshwari

Lab 8

Ex 1:

Function	Big O

bool isOperator(char c)
{
  if(c == '+' || c == '-' || c == '*' || c == '/' || c == '^')
  {
  return true;
  }
  return false;
}
	O(1)[if-else, return]

void inOrder(et *t)
{
  if(t){
  inOrder(t->left);
  cout<<t->val<<' ';
  inOrder(t->right);
  }
}
	Since every node is visited only once, T(n)=O(n) where n is the number of nodes in the tree

et* newNode(char v){
  et *temp = new et;
  temp->left = NULL;
  temp->right = NULL;
  temp->val = v;
  return temp;
}
	O(1)[declaration, assignment]

et* constructTree(string pf){
  stack<et *> st;
  et *t,*t1,*t2;
  for(char c:pf)
{
  if(!isOperator(c))
  {
  t = newNode(c);
  st.push(t);
  }
  else{
  t = newNode(c);
  t1 = st.top();
  st.pop();
  t2 = st.top();
  st.pop();
  t->right = t1;
  t->left = t2;
  st.push(t);
  }
  }
  t = st.top();
	O(n)[for loop, where n is the number of elements in the postfix expression]

```
    st.pop();
    return t;
}
```

```
Enter expression in postfix form: ab-ef*d-
The Expression Tree is: e * f - d
```