.....op.h.....

```
#include <bits/stdc++.h>
using namespace std;

set<char> input_terminals();
map<pair<char, char>, char> calculate_precedence_talbe(set<char> terminals);
void parsing_process(string input, map<pair<char, char>, char> precedence_table);
```

.....op.cpp.....

```
#include "op.h"

set<char> input_terminals()
{
   int size;
   set<char> res;

   cout << "Enter the total number of terminals: ";
   cin >> size;

   for (int i = 0; i < size; i++)
   {
       char temp;
       cout << "Enter Terminal " << i + 1 << ": ";
       cin >> temp;
       res.insert(temp);
   }

   return res;
}

map<pair<char, char>, char> calculate_precedence_talbe(set<char> terminals)
{
   map<pair<char, char>, char> res;

   terminals.insert('$');

   for (auto it = terminals.begin(); it != terminals.end(); it++)
   {
       for (auto it2 = terminals.begin(); it2 != terminals.end(); it2++)
       {
           char input;

           cout << "Enter the Precedence over " << *it << " and " << *it2 << ": ";
           cin >> input;
```

```
            res[{*it, *it2}] = input;
        }
    }

    return res;
}


void parsing_process(string input, map<pair<char, char>, char> precedence_table)
{
    stack<char> st;
    int index = 0;

    st.push('$');
    input += "$";

    cout << input << endl;


    while (input[index] != '$')
    {
        char tos;
        if (!st.empty())
            tos = st.top();

        if (precedence_table[{input[index], tos}] == '>')
        {
            st.push(input[index]);
            index++;

            cout << input[index] << " Pushed." << endl;
        }
        else
        {
            if (st.empty())
            {
                cout << "Error!" << endl;
                break;
            }
            char item = st.top();
            st.pop();

            cout << item << " Poped." << endl;
        }
    }

}
```

.....op_m.cpp.....

```cpp
#include "op.h"

int main()
{
   string input;// = "t+t";

   cout << "Enter the input string: ";
   getline(cin, input);

   set<char> terminals = input_terminals();

   map<pair<char, char>, char> table = calculate_precedence_talbe(terminals);

   parsing_process(input, table);
   return 0;
}
```

```
pawan@pawan:~/Desktop/Code/CD/LL1/New$ g++ op.cpp op_m.cpp -o op.out
pawan@pawan:~/Desktop/Code/CD/LL1/New$ ./op.out
Enter the total number of terminals: 2
Enter Terminal 1: +
Enter Terminal 2: t
Enter the Precedence over $ and $: .
Enter the Precedence over $ and +: <
Enter the Precedence over $ and t: <
Enter the Precedence over + and $: >
Enter the Precedence over + and +: >
Enter the Precedence over + and t: <
Enter the Precedence over t and $: >
Enter the Precedence over t and +: >
Enter the Precedence over t and t: .
t+t$
+ Pushed.
t Poped.
t Pushed.
$ Pushed.
pawan@pawan:~/Desktop/Code/CD/LL1/New$
```