Compiler Design Lab

Date: 19-12-2025 **Name:** S Vishwajith
Assignment No.: 4 **Register No.:** 23BCE1145

Aim:

To write a C++ program that converts the regular expression (a|b)*a(a|b) to NFA with epsilon-transition and check whether the given string is accepted or rejected by the NFA.

Algorithm:

1. For the regular expression (a|b)*a(a|b), construct a transition table for NFA with epsilon (null) transitions containing the current state, symbol for transition and the new state.
2. Initiate states = {0} where 0 is the initial state and take the null closure of states in the variable and store it in the states variable itself.
3. For each symbol ‘c’ in the string:
    a. Take a new empty variable ‘next_states’ or clear it if it exists.
    b. For each ‘st’ in states:
        i. If there are any possible transitions, add them to ‘next_states’
    c. Take the null closure of ‘next_state’.
    d. Replace state with ‘next_state’.
4. If the states set contains the final state number 3, then the given string is accepted. Else, the given string is rejected.

Code:

```
#include <iostream>

#include <string>

#include <set>

#include <vector>

#include <map>

using namespace std;


map<int, vector<pair<char, int>>> get_transition_table()
```

```
{
    map<int, vector<pair<char, int>>> tt;

    tt[0].push_back({'a', 0});
    tt[0].push_back({'b', 0});
    tt[0].push_back({'e', 1});
    tt[1].push_back({'a', 2});
    tt[2].push_back({'a', 3});
    tt[2].push_back({'b', 3});

    return tt;
}

int get_next_state(vector<pair<char, int>> transitions, char symbol)
{
    for (auto &[c, i] : transitions)
    {
        if (c == symbol)
            return i;
    }
    return -1;
}

void null_closure(set<int> &states, map<int, vector<pair<char, int>>> tt)
{
    bool changed;
    do
    {
```

```
        changed = false;
        set<int> snapshot = states;
        for (int s : snapshot)
        {
            int e = get_next_state(tt[s], 'e');
            if (e != -1 && !states.count(e))
            {
                states.insert(e);
                changed = true;
            }
        }
    } while (changed);
}

int main()
{
    string s;
    cout << "Enter a string: ";
    cin >> s;

    map<int, vector<pair<char, int>>> tt = get_transition_table();
    cout << endl
         << "Current State\tSymbol\tNew State" << endl;
    for (auto &[old_state, transitions] : tt)
    {
        for (auto &[symbol, new_state] : transitions)
        {
            cout << old_state << "\t\t" << symbol << "\t\t" << new_state << endl;
```

```
    }
}
cout << endl;

set<int> states = {0};
null_closure(states, tt);
for (char c : s)
{
    set<int> next_states;
    for (int st : states)
    {
        int ns = get_next_state(tt[st], c);
        if (ns != -1)
            next_states.insert(ns);
    }

    states = next_states;
    null_closure(states, tt);
}

for (int i : states)
{
    if (i == 3)
    {
        cout << "The given string is accepted." << endl;
        return 0;
    }
}
```

```
    cout << "The given string is rejected." << endl;
    return 0;
}
```

Inputs:

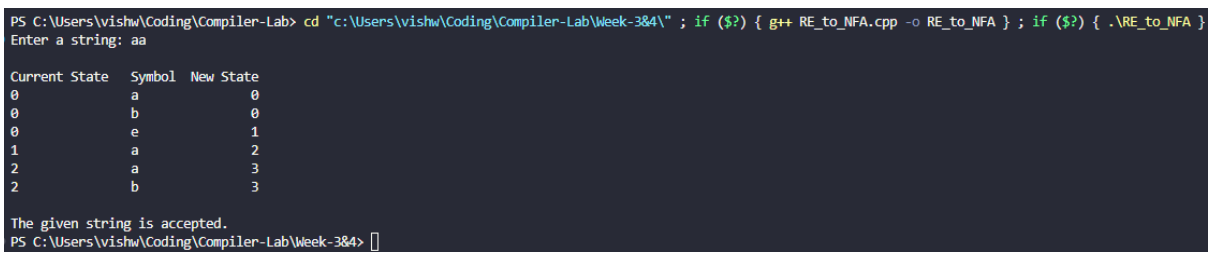
1. aa
2. ab
3. abba

Outputs:

Enter a string: aa

Current State	Symbol	New State
0	a	0
0	b	0
0	e	1
1	a	2
2	a	3
2	b	3

The given string is accepted.



Enter a string: ab

Current State	Symbol	New State

0	a	0
0	b	0
0	e	1
1	a	2
2	a	3
2	b	3

The given string is accepted.

```
PS C:\Users\vishw\Coding\Compiler-Lab> cd "c:\Users\vishw\Coding\Compiler-Lab\Week-3&4\" ; if ($?) { g++ RE_to_NFA.cpp -o RE_to_NFA } ; if ($?) { .\RE_to_NFA }
Enter a string: ab

Current State   Symbol  New State
0               a               0
0               b               0
0               e               1
1               a               2
2               a               3
2               b               3

The given string is accepted.
PS C:\Users\vishw\Coding\Compiler-Lab\Week-3&4>
```

Enter a string: abba

Current State	Symbol	New State
0	a	0
0	b	0
0	e	1
1	a	2
2	a	3
2	b	3

The given string is rejected.

```
PS C:\Users\vishw\Coding\Compiler-Lab> cd "c:\Users\vishw\Coding\Compiler-Lab\Week-3&4\" ; if ($?) { g++ RE_to_NFA.cpp -o RE_to_NFA } ; if ($?) { .\RE_to_NFA }
Enter a string: abba

Current State   Symbol  New State
0               a               0
0               b               0
0               e               1
1               a               2
2               a               3
2               b               3

The given string is rejected.
PS C:\Users\vishw\Coding\Compiler-Lab\Week-3&4>
```

Result:

The C++ program to convert the regex (a|b)*a(a|b) into NFA with epsilon transitions and check whether the given string is accepted or not was successfully run and the results were verified.