

CONVERSION FROM REGULAR EXPRESSION TO NFA

NAME: RISHAL RAMESH

EXP NO: 2

REG NO: RA1911030010084

DATE: 27/01/2022

AIM:

To study and perform regular expression to NFA(non deterministic automata) conversion in any of the programming languages.

LANGUAGE USED:

C++

ALGORITHM:

- Start
- Get the input from the user
- Initialise separate variables and functions for Postfix, Display and NFA.
- Create separate methods for different operators like +,*,.
- By using Switch case initialise different cases for the input
- For '.' operator initialise a separate method by using various stack functions. Do the same for other operators like *,+
- Regular expression is in the form of a.b(or) a+b
- Display the output
- Stop

CODE:

```
#include<iostream>
```

```
#include<string.h>
```

```
int main()
```

```

{
    printf("Enter the Regular Expression: ");
    char reg[20];
    int q[20][3],i,j,len,a,b;
    for(a=0;a<20;a++)
    {
        for(b=0;b<3;b++)
        {
            q[a][b]=0;
        }
    }
    scanf("%s",reg);
    len=strlen(reg);
    i=0;
    j=1;
    while(i<len)
    {
        if(reg[i]=='a'&&reg[i+1]!='|'&&reg[i+1]!='*')
        {
            q[j][0]=j+1;
            j++;
        }
        if(reg[i]=='b'&&reg[i+1]!='|'&&reg[i+1]!='*')
        {
            q[j][1]=j+1;
            j++;
        }
        if(reg[i]=='e'&&reg[i+1]!='|'&&reg[i+1]!='*')
        {
            q[j][2]=j+1;
            j++;
        }
    }
}

```

```

}
if(reg[i]=='a'&&reg[i+1]=='|'&&reg[i+2]=='b')
{
    q[j][2]=((j+1)*10)+(j+3);
    j++;
    q[j][0]=j+1;
    j++;
    q[j][2]=j+3;
    j++;
    q[j][1]=j+1;
    j++;
    q[j][2]=j+1;
    j++;
    i=i+2;
}
if(reg[i]=='b'&&reg[i+1]=='|'&&reg[i+2]=='a')
{
    q[j][2]=((j+1)*10)+(j+3);
    j++;
    q[j][1]=j+1;
    j++;
    q[j][2]=j+3;
    j++;
    q[j][0]=j+1;
    j++;
    q[j][2]=j+1;
    j++;
    i=i+2;
}
if(reg[i]=='a'&&reg[i+1]=='*')
{

```

```

        q[j][2]=((j+1)*10)+(j+3);
        j++;
        q[j][0]=j+1;
        j++;
        q[j][2]=((j+1)*10)+(j-1);
        j++;
    }
    if(reg[i]=='b'&&reg[i+1]=='*')
    {
        q[j][2]=((j+1)*10)+(j+3);
        j++;
        q[j][1]=j+1;
        j++;
        q[j][2]=((j+1)*10)+(j-1);
        j++;
    }
    if(reg[i]=='')&&reg[i+1]=='*')
    {
        q[0][2]=((j+1)*10)+1;
        q[j][2]=((j+1)*10)+1;
        j++;
    }
    i++;
}

printf("Transition function \n");
for(i=0;i<=j;i++)
{
    if(q[i][0]!=0)
        printf("\n q[%d,a]-->%d",i,q[i][0]);
    if(q[i][1]!=0)
        printf("\n q[%d,b]-->%d",i,q[i][1]);
}

```

```

    if(q[i][2]!=0)
    {
        if(q[i][2]<10)
            printf("\n q[%d,e]-->%d",i,q[i][2]);
        else
            printf("\n q[%d,e]-->%d & %d",i,q[i][2]/10,q[i][2]%10);
    }
}

return 0;
}

```

OUTPUT:

The screenshot shows the OnlineGDB beta web interface. The code editor contains the following C++ code:

```

1 #include<iostream>
2 #include<string.h>
3 int main()
4 {
5     printf("Enter the Regular Expression: ");
6     char reg[20];
7     int q[20][3],i,j,len,a,b;
8     for(a=0;a<20;a++)
9     {
10         for(b=0;b<3;b++)
11         {
12             q[a][b]=0;
13         }
14     }
15
16     // Transition function
17     q[0,e]-->4 & 1
18     q[1,a]-->2
19     q[2,b]-->3
20     q[3,e]-->4 & 1
21
22     ...Program finished with exit code 0
23     Press ENTER to exit console.

```

The output window shows the following text:

```

Enter the Regular Expression: (a+b)*c
Transition function
q[0,e]-->4 & 1
q[1,a]-->2
q[2,b]-->3
q[3,e]-->4 & 1
...Program finished with exit code 0
Press ENTER to exit console.

```

RESULT:

The regular expression to NFA conversion was successfully executed in C++.