

Date

EXPERIMENT- 2

21/02/23

CONVERTING REGULAR EXPRESSION TO NFA

AIM: To write program for converting regular expression to NFA.

ALGORITHM:

1. Start
2. Get input from the user.
3. Initialize separate variable and functions for postfix, Dijkstra and NFA.
4. Create separate methods for different operations like +, *.
5. By using switch case initialize different cases for the input.
6. For ' ' operator initialize different a separate method by using various stack functions. do the same for + & *.
7. Regular expression is in form a.b or a+b.
8. Display the output
9. Stop.

PROGRAM:

```
#include <stdio.h>
#include <string.h>
int main()
{
```

```

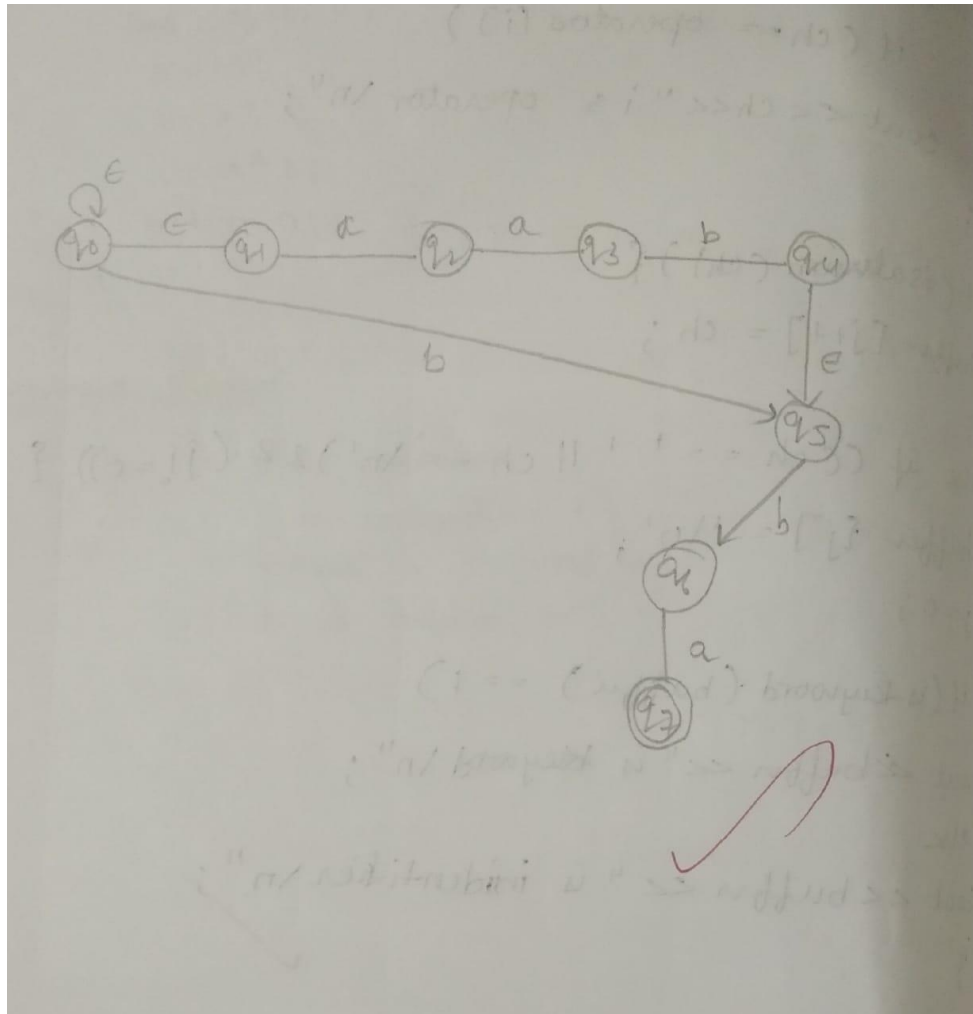
char reg[20]; int q[20][3], i=0, j=1, len, a, b;
for (a=0; a<20; a++) for (b=0; b<3; b++) q[a][b]=0;
scanf("%s", reg);
printf("Given regular exp: %s\n", reg);
len = strlen(reg);
while (i < len) {
    if (reg[i] == 'a' && reg[i+1] != '1' && reg[i+1] != '*')
        { q[j][0] = j+1; j++; }
    if (reg[i] == 'b' && reg[i+1] != '1' && reg[i+1] != '*')
        { q[j][1] = j+1; j++; }
    if (reg[i] == 'e' && reg[i+1] != '1' && reg[i+1] != '*')
        { q[j][2] = j+1; j++; }
    if (reg[i] == 'a' && reg[i+1] == '1' && reg[j+2] == 'b')
        { q[j][2] = ((j+1)*10) + (j+3); j++;
          q[j][0] = j+1; j++;
          q[j][1] = j+3; j++;
          q[j][2] = j+1; j++;
          q[j][2] = j+1; j++;
          i = i+2; }
    if (reg[i] == 'b' && reg[i+1] == '1' && reg[i+2] == 'a')
        { q[j][2] = ((j+1)*10) + (j+3); i++;
          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; }
}

```

output : $a(a+b)^*ba$

Transition table

Current state	Input	Next state
$q[0]$	ϵ	$q[5], q[1]$
$q[1]$	a	$q[2]$
$q[2]$	a	$q[3]$
$q[3]$	b	$q[4]$
$q[4]$	ϵ	$q[5], q[1]$
$q[5]$	b	$q[6]$
$q[6]$	a	$q[7]$



Online C Compiler

programiz.com/c-programming/online-compiler/

Programiz
C Online Compiler

Interactive C Course

```

1 #include<stdio.h>
2 #include<string.h>
3 int main()
4 {
5     char reg[20]; int q[20][3], i=0, j=1, len, a, b;
6     for(a=0; a<20; a++) for(b=0; b<3; b++) q[a][b]=0;
7     scanf("%s", reg);
8     printf("Given regular expression: %s\n", reg);
9     len=strlen(reg);
10    while(i<len)
11    {
12        if(reg[i]=='a' && reg[i+1]!='|' && reg[i+1]!='*') { q[j][0]=j+1; j++; }
13        if(reg[i]=='b' && reg[i+1]!='|' && reg[i+1]!='*') { q[j][1]=j+1; j++; }
14        if(reg[i]=='e' && reg[i+1]!='|' && reg[i+1]!='*') { q[j][2]=j+1; j++; }
15        if(reg[i]=='a' && reg[i+1]=='|' && reg[i+2]=='b')
16        {
17            q[j][2]=(j+1)*10+(j+3); j++;
18            q[j][0]=j+1; j++;
19            q[j][2]=j+3; j++;
20            q[j][1]=i+1; i++;
  
```

Output

/tmp/DL58L4twhv.o
a(a+b)*ba
Given regular expression: a(a+b)*ba
Transition Table

Current State	Input	Next State
q[0]	e	q[5], q[1]
q[1]	a	q[2]
q[2]	a	q[3]
q[3]	b	q[4]
q[4]	e	q[5], q[1]
q[5]	b	q[6]
q[6]	a	q[7]

Get Started!

Compiler design...html
33/136 B

Type here to search

Earnings upcoming

10:08 PM
07-02-2023

RA2011028010099
DURGA CHANDANA SREE M

```

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][0] = 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("\n \t transition table \n");
printf("current state \t input \t next state");
for (i=0; i<=j; i++)
{
    if (q[i][0] != 0) printf("\n q[x.d] \t ");
    if (q[i][1] != 0) printf("\n q [1.d] \t | b | q[0.d]");
    if (q[i][2] != 0) printf("\n q [2.d] \t | c | q[0.d]");
    if (q[i][2] < 10) printf("\n q [1.d] \t | e | q[0.d]");
    else printf("\n q [1.d] \t | e | q [1.d], q [2.d]");
}

return 0; }

```

RESULT: Hence the program to convert regular expression to NFA is implemented successfully.