ASSIGNMENT 4

Sub: FLAT Lab Date :14-Dec-20

Submitted by Akash Kumar reg:1801289021

Question 1.Convert It into Equivalent DFA

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

#define STATES 256
#define SYMBOLS 20

int N_symbols;
int NFA_states;
char *NFAtab[STATES][SYMBOLS];
int DFA_states;
int DFAtab[STATES][SYMBOLS];

void put_dfa_table( int tab[][SYMBOLS],
                    int nstates,
                    int nsymbols)
{


int i, j;

puts("STATE TRANSITION TABLE");

printf ("Q/E");
printf (" |");

for (i = 0; i <nsymbols; i++)
printf(" %3c",'0'+i);
printf ("\n----+--");

for (i = 0; i <nsymbols; i++)
printf ("----");
printf ("\n");

for (i = 0; i <nstates; i++)
 {
printf(" %2c | ",'A'+i);
for (j= 0; j <nsymbols;j++)
printf(" %2c ",'A'+tab[i][j]);
printf("\n");
 }

}
void init_NFA_table()

{

NFAtab[0][0] ="12";
NFAtab[0][1] ="13";
NFAtab[1][0] ="12";
NFAtab[1][1] ="13";
NFAtab[2][0] ="4";
NFAtab[2][1] ="";
NFAtab[3][0] ="";
NFAtab[3][1] ="4";
NFAtab[4][0] ="4";
NFAtab[4][1] ="4";

NFA_states = 5;
DFA_states = 0;
N_symbols = 2;
```

```
}
void string_merge(char *s, char *t)
{
char temp[STATES], *r=temp, *p=s;

while (*p && *t)
    {
if(*p ==*t)
        {
            *r++ = *p++;
t++;
        }

else if(*p < *t)
    {
        *r++ = *p++;
    }
else

    {
        *r++ = *t++;
    }
}


*r='\0';
if(*p)strcat(r,p);

else if(*t)strcat(r,t);
strcpy(s,temp);
}

void get_next_state(char *nextstates, char *cur_states, char *nfa[STATES][SYMBOLS] ,int n_nfa, int symbol)
 {


int i;
char temp[STATES];

temp[0] = '\0';
for (i = 0; i <strlen(cur_states); i++)
string_merge(temp, nfa[cur_states[i]-'0'][symbol]);
strcpy(nextstates, temp);
 }

int state_index(char *state, char statename[][STATES], int *pn)
{


int i;
if (!*state)
return -1 ;

for (i = 0; i < *pn; i++)
if (!strcmp(state, statename[i]))
return i;

strcpy(statename[i], state);

return(*pn)++;
}
```

```
122 }
123
124 int nfa_to_dfa(char *nfa[STATES][SYMBOLS],int n_nfa,int n_sym, int dfa[][SYMBOLS])
125 {
126
127 char statename[STATES][STATES];
128 int i = 0;
129 int n = 1;
130
131 char nextstate[STATES];
132 int j;
133 strcpy(statename[0], "0");
134 for (i = 0; i <n;i++)
135 {
136 for (j= 0; j <n_sym; j++)
137 {
138 get_next_state(nextstate, statename[i], nfa, n_nfa, j);
139 dfa[i][j]= state_index(nextstate, statename, &n);
140 }
141
142 }
143 return n;
144
145 }
146
147 int main()
148 {
149
150 init_NFA_table();
151
152 DFA_states = nfa_to_dfa(NFAtab, NFA_states, N_symbols, DFAtab);
153 put_dfa_table(DFAtab, DFA_states, N_symbols);
154
155 return 0;
156
157 }
158
159
```

```
STATE TRANSITION TABLE
Q/E |   0   1
----+----------
  A |   B   C
  B |   D   C
  C |   B   E
  D |   D   E
  E |   D   E

Process returned 0 (0x0)   execution time : 2.041 s
Press any key to continue.
```