THEORY OF COMPUTATION
ASSIGNMENT

Q1. Given a DFA, write a program to find the minimum state DFA

BASIC STRUCTURE

1. **DFA.h** is the header file that contains all the variables and the function prototypes
2. **DFA_MIN.cpp** is where the implementation of the minimisation algorithm takes place.
3. **main.cpp** is the driver function where the user inputs the tuples of a DFA and corresponding transition and subsequently call the minimisation algorithm functions in DFA_MIN.cpp

1. DFA.h

The header file contains all the variables and the function prototypes used further.

VARIABLES

TYPE	NAME	DESCRIPTION
int	num_symbols	Number of symbols in the alphabet
int	num_states	Number of states in the DFA
char	final_states[STATES]	All the final states
int	DFAMat[STATES][SYMBOLS]	The transition matrix that holds where a current state (row) goes on input symbol (column)
char	StateName[STATES][STATES+1]	The state name table
int	new_states	Number of optimized DFA states
int	Upd_DFA[STATES][SYMBOLS]	The New Updated DFA Transition matrix
char	NEW_finals[STATES+1]	The new final states

FUNCTIONS

1. **DisplayDFA_Matrix() -** Print state-transition table. State names: 'A', 'B', 'C', ... Symbols: 0,1,2......
2. **Next_State()** - Get next-state string for current-state string.
3. **Class_Index()** - Get index of the equivalence states for state 'ch' Equiv. class id's are '0','1', '2', ...
4. **is_one_nextstate()** - Check if all the next states belongs to same equivalence class.
   Return value:
      If next state is NOT unique, return 0.
      If next state is unique, return next state --> 'A/B/C/...'
   's' is a '0/1' string: state-id's
5. **Divide_Class()** - Divides the DFA states into finals and non-finals.
6. **Update_DFA()** - Get optimized DFA 'newdfa' for equiv. class 'stnt'.
7. **append()** - char 'ch' is appended at the end of 's'.

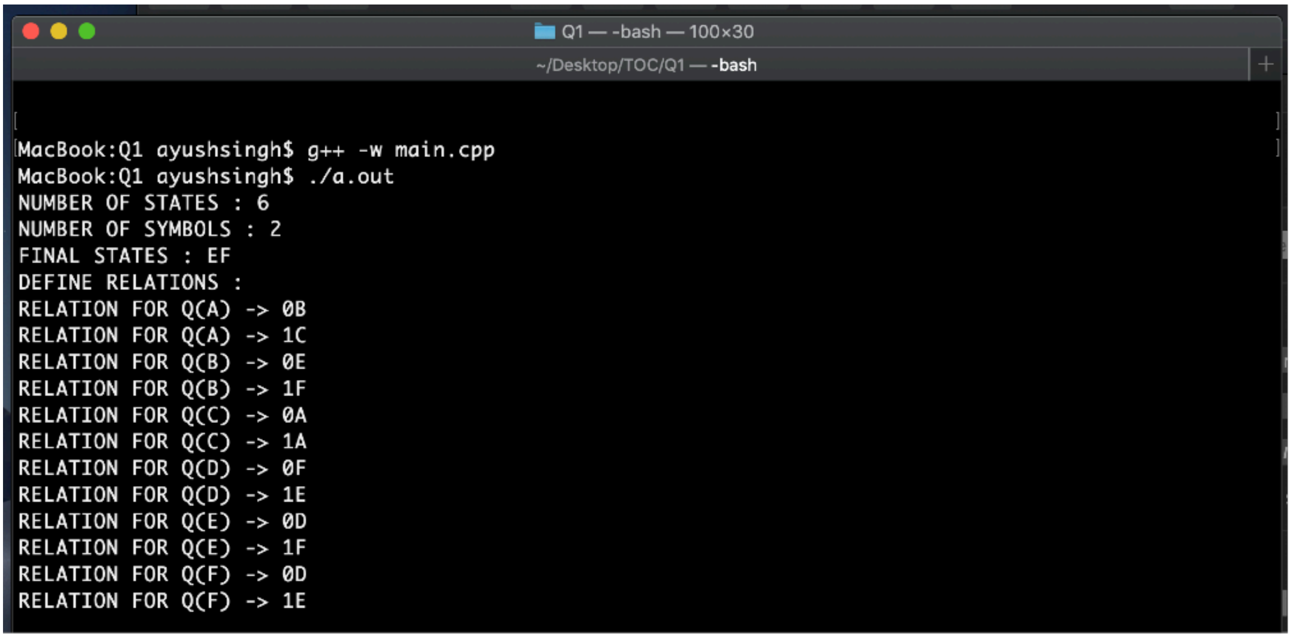
8. **Split_Class() -** Divide first equivalent class into subclasses.
   stnt[i1] : equiv. class to be segmented
   stnt[i2] : equiv. vector for next state of stnt[i1]

   Algorithm:
   - stnt[i1] is splitted into 2 or more classes 's1/s2/...'
   - old equiv. classes are NOT changed, except stnt[i1]
   - stnt[i1]=s1, stnt[n]=s2, stnt[n+1]=s3, ...

   Return value: number of NEW equiv. classes in 'stnt'.
9. **combine_class()** - Equiv. classes are combined to get NEW equiv. classes.
10. **optimize_DFA()** - State-minimization of DFA: 'dfa' --> 'newdfa'
    Return value: number of DFA states.
11. **get_NEW_finals()** - New finals states of reduced DFA.

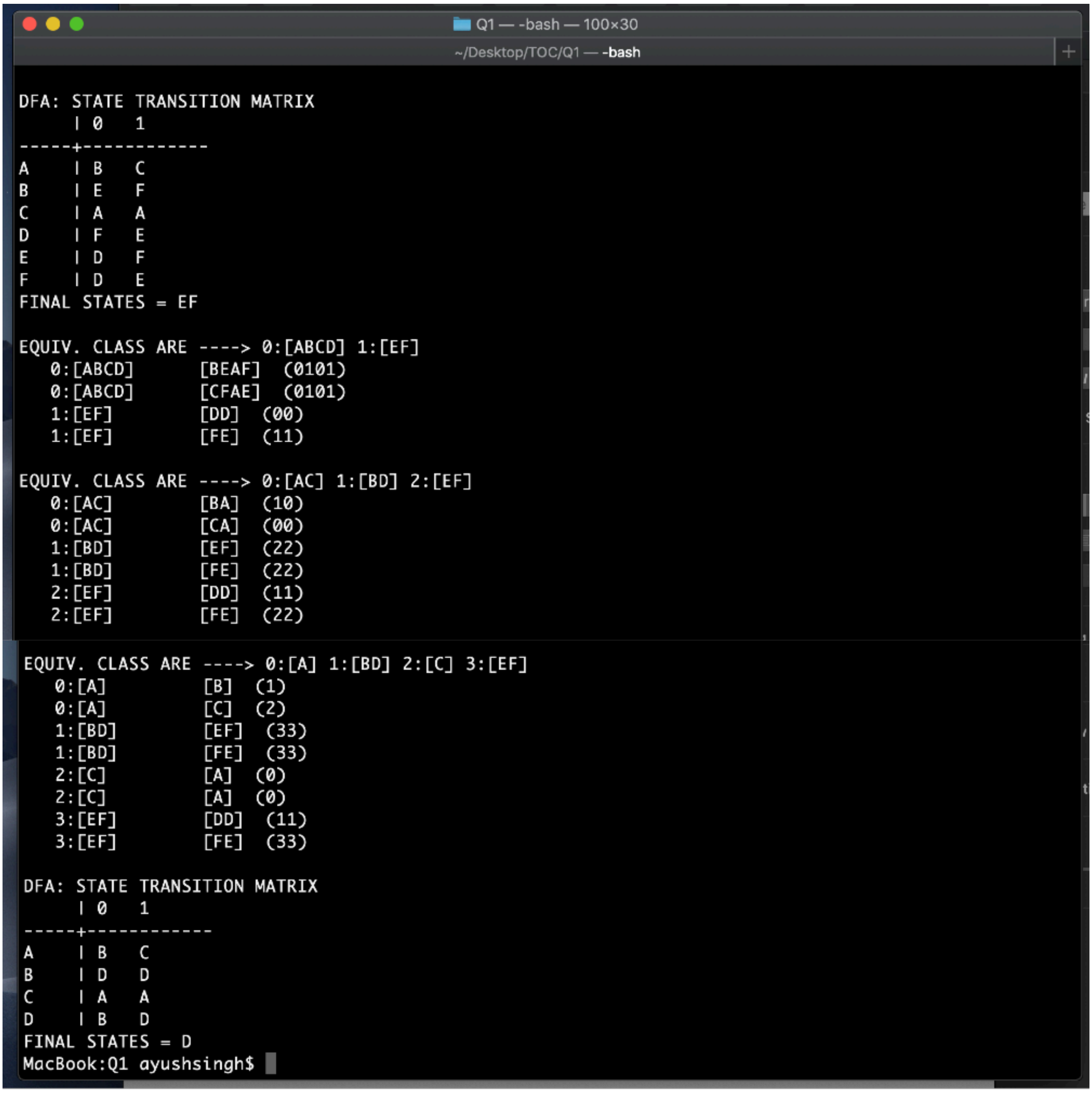
INPUT

Assumption : The states and symbols are implicit, i.e. states start from 'A', 'B', 'C' ... and so on and
symbols are '0', '1', '2' ...



```
MacBook:Q1 ayushsingh$ g++ -w main.cpp
MacBook:Q1 ayushsingh$ ./a.out
NUMBER OF STATES : 6
NUMBER OF SYMBOLS : 2
FINAL STATES : EF
DEFINE RELATIONS :
RELATION FOR Q(A) -> 0B
RELATION FOR Q(A) -> 1C
RELATION FOR Q(B) -> 0E
RELATION FOR Q(B) -> 1F
RELATION FOR Q(C) -> 0A
RELATION FOR Q(C) -> 1A
RELATION FOR Q(D) -> 0F
RELATION FOR Q(D) -> 1E
RELATION FOR Q(E) -> 0D
RELATION FOR Q(E) -> 1F
RELATION FOR Q(F) -> 0D
RELATION FOR Q(F) -> 1E
```

<u>OUTPUT</u>



```
DFA: STATE TRANSITION MATRIX
     | 0   1
-----+------------
A    | B   C
B    | E   F
C    | A   A
D    | F   E
E    | D   F
F    | D   E
FINAL STATES = EF

EQUIV. CLASS ARE ----> 0:[ABCD] 1:[EF]
   0:[ABCD]      [BEAF]  (0101)
   0:[ABCD]      [CFAE]  (0101)
   1:[EF]        [DD]  (00)
   1:[EF]        [FE]  (11)

EQUIV. CLASS ARE ----> 0:[AC] 1:[BD] 2:[EF]
   0:[AC]        [BA]  (10)
   0:[AC]        [CA]  (00)
   1:[BD]        [EF]  (22)
   1:[BD]        [FE]  (22)
   2:[EF]        [DD]  (11)
   2:[EF]        [FE]  (22)

EQUIV. CLASS ARE ----> 0:[A] 1:[BD] 2:[C] 3:[EF]
   0:[A]         [B]  (1)
   0:[A]         [C]  (2)
   1:[BD]        [EF]  (33)
   1:[BD]        [FE]  (33)
   2:[C]         [A]  (0)
   2:[C]         [A]  (0)
   3:[EF]        [DD]  (11)
   3:[EF]        [FE]  (33)

DFA: STATE TRANSITION MATRIX
     | 0   1
-----+------------
A    | B   C
B    | D   D
C    | A   A
D    | B   D
FINAL STATES = D
MacBook:Q1 ayushsingh$ 
```