EECS 665 – Fall 2016
Assignment 1
nfa2dfa – Subset construction to convert NFA to DFA

Your assignment is to implement the subset construction algorithm (Figure 3.32 in the text) to convert an NFA to the equivalent DFA. You will also need to implement other algorithms, such as those for the *move* (Figure 3.31) and ϵ–*closure* (Figure 3.33) operations, required for subset construction.

Both the input NFA and the output DFA will be represented as a transition table. You should represent the NFA and DFA states as integer numbers, starting at 1. A special symbol 'E' will be used to indicate the ϵ–transition.

The input will come from *standard input* and the output should be printed to *standard output* (monitor). You should not make any assumptions regarding the maximum number of states in the input NFA or the output DFA. You should attempt to match your output as closely to my output as possible.

This assignment should be implemented in either C or C++. Provide a *Makefile* that will include commands to build your program.

This assignment will be due on **Friday, October 7**. You should comment your program so that others (e.g. the grader) can understand it. You should also have comments at the top of the file indicating your name, this course, and the assignment. 5 points (out of 100) will be designated for *style*, which will include code structure, code comments, and output formatting (to match my output format). Your code should run on the EECS `cycle` server machines.

E-mail a single '.tar' file containing your program source file(s) and the Makefile to your respective Lab TA (either Brad – `brad.torrence@ku.edu`, or Tyler – `t982w485@ku.edu`) and CC it to '`prasadk@ku.edu`' before the beginning of class on Friday, Oct. 7^{th}.

Example input NFA:

```
Initial State: {1}
Final States: {11}
Total States: 11
State   a       b       E
1       {}      {}      {2,5}
2       {3}     {}      {}
3       {}      {4}     {}
4       {}      {}      {8}
5       {}      {6}     {}
6       {7}     {}      {}
7       {}      {}      {8}
8       {}      {}      {9,11}
9       {10}    {}      {}
10      {}      {}      {9,11}
11      {}      {}      {}
```

Corresponding output DFA:

```
E-closure(I0) = {1,2,5} = 1

Mark 1
{1,2,5} --a--> {3}
E-closure{3} = {3} = 2
{1,2,5} --b--> {6}
E-closure{6} = {6} = 3

Mark 2
{3} --b--> {4}
E-closure{4} = {4,8,9,11} = 4

Mark 3
{6} --a--> {7}
E-closure{7} = {7,8,9,11} = 5

Mark 4
{4,8,9,11} --a--> {10}
E-closure{10} = {9,10,11} = 6

Mark 5
{7,8,9,11} --a--> {10}
E-closure{10} = {9,10,11} = 6

Mark 6
{9,10,11} --a--> {10}
E-closure{10} = {9,10,11} = 6

Initial State: {1}
Final States: {4,5,6}
State   a       b
1       {2}     {3}
2       {}      {4}
3       {5}     {}
4       {6}     {}
5       {6}     {}
6       {6}     {}
```