Homework #5

For all those questions requiring diagrams or text, attach a PDF file named `hw5.pdf` with your answers.

Question 1 (6 pt.)

Consider regular expression `a*a(b|c)`.

a) Use the McNaughton-Yamada-Thompson algorithm to construct an NFA that accepts the language represented by this regular expression. Do not skip any steps of the algorithm, even if you think that some of the inserted states or transitions might be redundant.

b) Convert the generated NFA into an equivalent DFA, using the algorithm presented in class, based on the *ε-closure* and *move* operations on the NFA states. Construct the DFA transition table, where each set of reachable NFA states is associated with an equivalent DFA state, and give the transition graph for the resulting DFA.

c) Write a C++ program that takes an input string as an argument and prints whether this string is matched by the regular expression above. Use class `DFA` (included in header file `DFA.h` available in the additional material) to simulate the automaton you constructed in the previous question. Attach a file named `q1.cc` with your code. Your source code should build without modifications when linked together with file `DFA.cc`.

Question 2 (4 pt.)

Write a Flex program that scans the input code shown below, and splits it into tokens. For each token, the program should print its name and the associated lexeme. Upload your code in a file named `q2.l`.

You can load the Flex input from a text file, instead of the standard input, by opening the file with an `std::ifstream` object, and passing a reference to it in the constructor of class `yyFlexLexer`:

```
std::ifstream f("code.txt");
yyFlexLexer lexer(&f);
```

Also, you can add `#include` files in the first section of the Flex file using a block headed by `%{` and ended by `%}`, as follows:

```
%{
#include <fstream>
%}
```

This is the program to scan, which you can store, for example, in a file named `code.c`:

```
/* Main function */
int main( )
{

	float m, n;
	printf("\nEnter a number: ");
	scanf("%f", &m);

	// Function call
	n = square(m);
	printf("\nSquare of the given number %f is %f", m, n);
}
```