University of Toronto
Department of Electrical and Computer Engineering

Midterm Examination

CSC467 - Compilers and Interpreters

Fall 2009

Print your name neatly in the space provided below.

Name:
ID Number:

Grade:

Question	Mark
1	
2	
3	
4	
Total	

Problem 1 (5 marks): Divide the following C++ program into tokens.

```
template <class T> class MyClass {   // my class
    public:
        bool calc( T a, T b ) {
            return a << b < 10;
            }
    };
```

Write the lexeme of each token in separate line.

Problem 2 (15 marks): Consider the following regular expressions:

```
R1: “<b>”
R2: “</b>”
R3: a a*
R4: a b*
R5: R1 (R3 | R4) R2
```

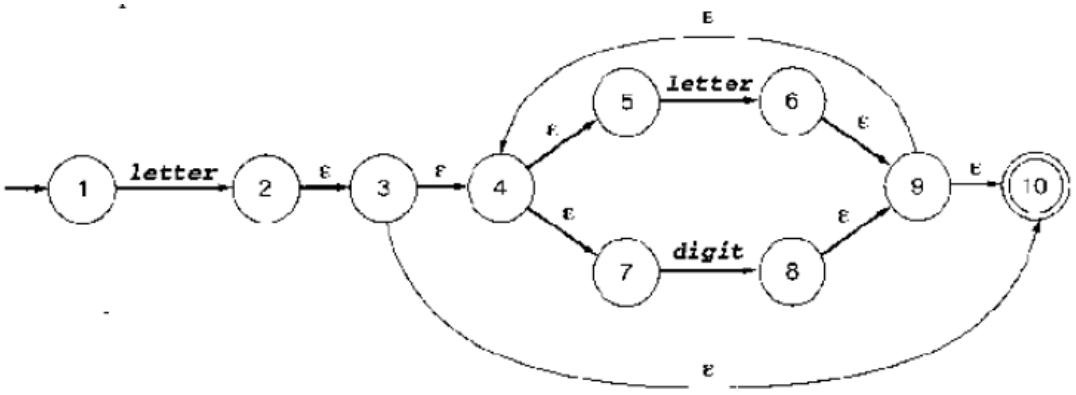
where { a, b, <, /, > } is the alphabet.

a) Draw the NFA for R1, R2, R3, R4. Clearly label the start and accepting state.

b) Draw the NFA for R5, by using the NFAs constructed in a).

c) Give 2 examples of valid inputs for R5, and 2 examples of invalid inputs for R5.

Problem 3 (10 marks): Consider the following NFA to a DFA. Clearly mark each DFA state with the corresponding NFA states.



Problem 4 (20 marks): Consider the following grammar for expressions:

```
Expr ::= 1 | 2 | ID | Expr + Expr | (Expr)
```

where { `1 2 ID + * ( )` } are the set of tokens.

a) Explain in what aspect that the grammar is not satisfactory if a top-down parser is to be constructed.

b) Assume + is right associative, rewrite the grammar below, and indicate the start symbol.

c) Compute FIRST for all grammar symbols and FOLLOW for all nonterminals.

d) Assume yylex() is available to you as the lexer, which returns an integer code distinguishing the different tokens, write a recursive descent parser (top-down) in C below. Make sure the C procedure name matches the non-terminals defined in b).