Faculty of Computer Science

DALHOUSIE UNIVERSITY
Inspiring Minds

Midterm Exam

Term: Winter 2005 (Jan4-Apr8)

Student ID Information

Last name: ____________________ **First name:** ____________________

Student ID #: ____________________ **CS.Dal.Ca userid:** ____________________

Course ID:		**CSCI 3136**
Course Title:		Principles of Programming Languages
Instructor:		Vlado Keselj
Date of Exam:		2 Mar 2005
Time Period:	Start:	15:30
	End:	17:30
Duration of Exam:		2 hours
Number of Exam Pages: (including this cover sheet)		11 pages
Exam Type:		**Closed Book**
Additional Materials Allowed:		**No**

Grade Table

Question	Score
1	/8
2	/10
3	/14
4	/16
5	/10
6	/10
7	/9
8	/10
Σ	/87

1. (8 points) Introduction to Programming Languages

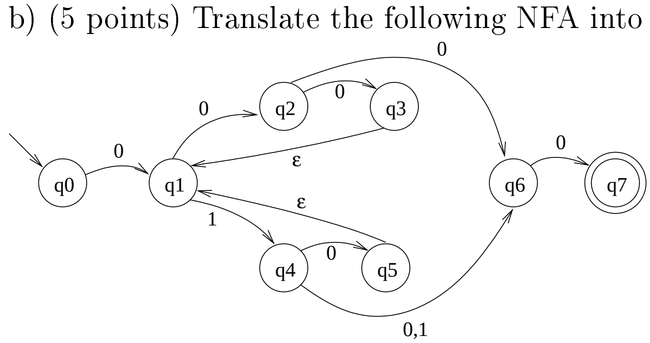
a) (4 points) What are the benefits of using intermediate forms in compilation?

b) (4 points) What is the difference between compilers and interpreters?

2. (10 points) Regular expressions and Automata

a) (5 points) Construct a DFA or NFA which accepts all non-empty strings of letters **a** and **b**, such that each **a** is followed either by a **b** or by the sequence **aba**, and each **b** is either at the end of the string or it is followed by an **a**.

b) (5 points) Translate the following NFA into a DFA.



3. (14 points) Context-Free Grammars

The following context-free grammar is given:

$$
\begin{array}{lcl}
S & \rightarrow & ABB \\
A & \rightarrow & aA \\
A & \rightarrow & \epsilon \\
B & \rightarrow & bA \\
B & \rightarrow & \epsilon
\end{array}
$$

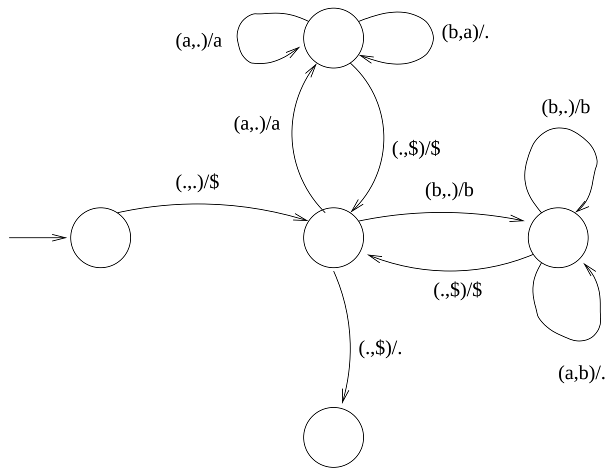
where S is the start symbol, S, A, and B are nonterminals, the tokens are a and b.

a) (10 marks) Calculate FIRST, FOLLOW, and PREDICT tables and verify whether the grammar is LL(1) or not.

b) (4 marks) Is the string `aabab` accepted by the grammar? If yes, then write its left-most derivation.

4. (16 points) Push-Down Automata

a) (8 points) What language is accepted by the following PDA?



a) (8 points) Prove that a language accepted by empty stack by a PDA is also accepted by empty stack by a PDA with only one state. In other words, any PDA can be simulated by a one-state PDA.

5. (10 points) Scheme

a) (5 marks) What function is defined with the following code? What is the value of `(f 5)`?

```
(define f (lambda (x) (if (< x 1) 0 (+ (f (- x 2)) (* x x)))))
```

b) (5 points) What is the value of `x` after evaluating each of the following two Scheme expressions? Present results as structures of the cells.

```
(define x (cons '(1 2 3 4) '(5 (6) ) ))
(set-cdr! (car (cdr (cdr x)))
          (cdr (cdr (cdr (car x))))
)
```

6. (10 points) Scoping

a) (5 points) What are two kinds of scoping used in programming languages? Briefly describe difference between them.

b) (5 points) What is the output of the following program if one kind of scoping is used and what if the other is used?

```
program main
int x;
procedure A() { print x }
procedure B(int x) {
  print x;
  A();
}
procedure C() {
  int x = 20;
  print x;
  A();
}

x = 10;
print x;
B(5)
C();
A();
```

7. (9 points) Shallow and Deep Binding

a) (4 points) Explain the difference between shallow and deep binding.

b) (5 points) The linear congruence method for generating random numbers starts with a value x as a seed. Each time, a new random number is generated using the formula $x \leftarrow (a \cdot x + 1) \bmod n$, where a is a constant chosen with great care, and n is given by the user. The result is a pseudorandom value between 0 and $n - 1$.

Read the following Scheme code carefully and show how to use it:

```
(define random-const 31415821) ;; this is the value a
(define random-integer
  (lambda (seed)  ;; seed is the x in (ax+1) mod n
    (lambda (n)
      (set! seed
            (modulo (+ 1 (* random-const seed)) n))
      seed)))
```

8. (10 points) Data Abstraction

a) (5 marks) What is the difference between modules, module types, and classes?

b) (5 marks) What output is produced after the function call `f(2)`, where function `f` is defined in C in the following way:

```
void f(int n) {
  int x = 0;
  static int y = 0;
  x = x - n;
  y = y + n;
  if (n > 0) f(n-1);
  printf("%d %d ", x, y);
}
```

(extra blank page provided)

(extra blank page provided)