CS 453 Introduction to Compilers	**Midterm Examination**
Spring 2008	**March 12, 2008**

50 minutes (maximum)
Closed Book

- You may use one side of one sheet (8.5x11) of paper with any notes you like.
- This exam has 8 pages, including this cover page and a blank page at the end. Do all your work on these exam sheets, use the backs of the pages if needed.
- Be specific and clear in your answers. If there is any question about what is being asked, then indicate the assumptions you need to make to answer the question.
- Show all your work if you wish to be considered for partial credit.
- If you use the back pages for scratch space, cross out your scratch work before you submit the exam.

Question	Points	Score
1	10	
2	25	
3	15	
4	30	
5	20	

Name: _____

Email: _____

DO NOT TURN TO NEXT PAGE TILL YOU GET PERMISSION

1. [10 points] Terminology. Draw a diagram with the following components, which shows how the components interact:

   - ast
   - token stream
   - parser
   - lexer
   - symbol table
   - semantic analysis
   - character stream or file

   Put rectangles around the components that DO something and circles around those that indicate data structures. An arrow from an active component to a data structure indicates the production of the data structure as output. If the arrow goes the other way, it is an indication that the data structure is input to the active component.

2. [25 points] Top-Down Parsing. Construct a recursive-descent parser for the grammar:

```
E -> S
S -> + S S
S -> - S S
S -> num
```

You can assume the functions match() and panic() are available.

```
match(tok) { if(tok=lookahead) lookahead = scan();
             else throw new SyntaxException(message); }
panic( nonterminal ) {
    print error;
    while ( scan() not in (FOLLOW(nonterminal)) ) {
    }
}
```

Indicate the FOLLOW set for each nonterminal in a separate table.

3. [15 points] Syntax-directed Translation.

```
E ->  S

S ->  +     S     S

S ->  -     S     S

S ->  num
```

Associate actions with the above grammar so that the following output occurs for each of the following example strings:

string	output
+ - 5 4 + 3 2	6
- 5 10	-5
2	2
+ + + 1 2 3 4	10

Your actions can be expressed using JavaCUP syntax where the RESULT associated with E holds the value, or you can use attributes and print statements.

4. [30 points] LR Parsing Table.

   a) For the following grammar:

```
(1) E -> E AND B
(2) E -> B
(3) B -> true
(4) B -> false
```

   Draw the LR(1) states and transitions between those states. Then fill in the empty LR(1) parse table on the next page. The table may include more states than you will actually need.

	true	false	AND	$	E	B
0						
1						
2						
3						
4						
5						
6						
7						

b) Is this grammar LR(0)? Why or why not?

5. [20 points] Ambiguity.
   a) Rewrite the following grammar so that it is not ambiguous.

```
E -> E @ E
E -> E # E
E -> [ E ]
E -> id
```

   You CANNOT use the JavaCUP precedence feature. Ensure that the following low to high precedence is satisfied.

```
#
@
[ ]
```

   b) Show the parse tree using the grammar you wrote for the following string:

```
a # b @ c @ d
```

(empty page)