UNIVERSITY OF CALGARY  
FACULTY OF SCIENCE  
FINAL EXAMINATION  
COMPUTER SCIENCE 411 L01

20 April 2005  
DURATION: 2 hours

- This exam has six (6) questions.
- This exam is closed book. No notes, books, calculators or electronic devices, or other assistance may be used.
- Write all your answers directly in the exam.
- Write your full name and student I.D. number in the spots provided in the exam.
- State any assumptions you make.
- Show rough work.

Name: ________________________________
Student I.D. ________________________________

FOR MARKING USE ONLY:

<table>
<thead>
<tr>
<th>Question</th>
<th>Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>/60</strong></td>
</tr>
</tbody>
</table>
Question 1 (8/60)

It is not possible to construct a DFA that will recognize nested comments in a program, yet you can write a scanner using `lex` that recognizes nested comments.

A. Explain how this is possible.

B. Sketch a solution using `lex` pseudocode.
Question 2 (8/60)

Draw a binary AST for the following code. Make sure your AST nodes are labeled so that the correspondence to the code is evident.

```c
switch (2 + 3 * 5) {
    case 11:
    case 13:
        blarg();
        break;
    case 17:
        goto X;
    default:
        continue;
}
```


**Question 3 (8/60)**

Draw the scope stack and symbol tables for the following code at the point marked with a dagger (†).

```c
void foo(char ch) {
    int i;
    struct {
        int i;
        int j;
    } s1;
    struct {
        int k; †
        int m;
    } s2;
    unsigned int len;
}
```
Question 4 (8/60)

One type of goto statement in Fortran ‘… allows for a multi-way decision based upon the value of an integer expression, and takes the form

\[ \text{GOTO (label1, label2, ..., labeln), integer expression} \]

This is called a \textit{computed} GOTO and causes control to be transferred to the statement labelled label1 if the value of the integer expression is 1, to the one labelled label2 if it is 2, and so on. If the value of the integer expression is negative or zero, or if it is greater than the number of labels specified, then the computed GOTO has no effect and the next statement is obeyed.\(^1\)

Show a translation of the following code into pseudo-assembly code. As in lectures, S1–S4 denote arbitrary statements.

\begin{verbatim}
1   S1
    GOTO (42, 1, 2112), 2+1
  S2
42  S3
2112 S4
\end{verbatim}

Do not optimize the code in any way. Your pseudo-assembly translation should be as detailed as possible.

\(^1\)T. M. R. Ellis, \textit{A Structured Approach to Fortran 77 Programming}, Addison-Wesley, 1982, p. 79.
Question 5 (8/60)

Design a grammar to describe the computed GOTO in the last question. You may assume that you are supplied a rule for “integer expression” and that the scanner returns tokens for:

- GOTO keywords
- left parentheses
- right parentheses
- commas
- natural numbers
Question 6 (20/60)

Construct the LALR(1) parse table for the grammar below. Show the alternate grammar and its FOLLOW sets as well.

0 \( S' \to S \)
1 \( S \to a \ B \ c \)
2 \( B \to Q \ d \ B \)
3 \( B \to c \)
4 \( Q \to B \)

Use the diagram on the opposite page for your state machine, and be sure to use the state numbers from the diagram when building your table.
Remember to fill in the diagram fully, including labeling the edges.

---

***************