This exam has eight (8) questions.

You may use one single-sided 8.5x11” piece of paper with whatever you want written on it. Apart from that, 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 all work.

Name: __________________________________________

Student I.D. ____________________________________
### Question 1 (2/30)

Explain the difference between static and dynamic semantic checks, and when you would perform each of them.
**Question 2 (4/30)**

For the grammar

\[
\begin{align*}
S & \rightarrow a \ A \\
A & \rightarrow b \ A \\
A & \rightarrow c
\end{align*}
\]

whose LL(1) parse table is

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>S</td>
<td>a</td>
<td>A</td>
<td>a A</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>b A</td>
<td>A</td>
<td>A c</td>
</tr>
</tbody>
</table>

show the LL(1) parser trace for the input below.

<table>
<thead>
<tr>
<th>Stack</th>
<th>Input</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>a b b c $</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Question 3 (4/30)**

For the grammar

\[
\begin{align*}
S' & \rightarrow S \\
S & \rightarrow a A a \\
A & \rightarrow b A b \\
A & \rightarrow c
\end{align*}
\]

give pseudocode for a recursive-descent parser. You may assume that either a peek/match or lex/unlex interface to your scanner exists already.
Question 4 (5/30)

Consider a “selection” statement as shown below. $S_0 \ldots S_n$ denote arbitrary statements.

```
select
  b0: S0
  | b1: S1
  | b2: S2
  ...  
  | bn: Sn
end
```

The selection statement has the following semantics. All Boolean variables $b_0 \ldots b_n$ are evaluated. If none of them are true, there is an error. If exactly one of them is true, the corresponding statement is executed. If more than one are true, one of the statements corresponding to a true Boolean is chosen at random to execute. Code executed does not “fall through” to code for the next case(s).

Show a translation of this construct into pseudo-assembly code. You may assume the existence of a $\text{rand}$ function in the run-time system that takes a positive integer $N$ as a parameter and returns a uniformly distributed pseudorandom integer between $0 \ldots N - 1$, inclusive. Your pseudo-assembly translation should be as detailed as possible.
Question 5 (7/30)

Using any method, construct the LALR(1) parse table for the grammar below. As in lectures, \( \lambda \) denotes the empty string.

- \( S' \rightarrow S \)
- \( S \rightarrow a A \)
- \( A \rightarrow b A a \)
- \( A \rightarrow \lambda \)

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.\(^1\)

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>$</th>
<th>S'</th>
<th>S</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Question 6 (1/30)

For the last question, explain how the SLR(1) table would be different, if at all. Be sure to clearly identify any conflicts that might result.

---

\(^1\)There are no marks for filling this out, but this gives you a way to check part of your answer, and gives me a way of marking this and getting you your mark relatively quickly.
This diagram is for use with Question 5.
**Question 7 (5/30)**

Show how each of the optimizations indicated would transform the code below. At each step, be sure to *only* apply the indicated optimization, and apply the optimizations in the order shown.

Assume that $w$ is a global variable; other variables are local variables and are not used after this code.

<table>
<thead>
<tr>
<th>Original code</th>
<th>#1 – constant propagation</th>
<th>#2 – constant folding</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x = 5$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$y = x + 3$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$z = w \times y$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>print $z$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>print $y$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#3 – constant propagation</th>
<th>#4 – strength reduction</th>
<th>#5 – dead code elimination</th>
</tr>
</thead>
</table>
Question 8 (2/30)

Given the following single-inheritance code below

```java
class C {
    field c1
    method m1()
    method m2()
}

class D extends C {
    field d1
    method m1()
    method m2()
}

class E extends D {
    field e1
    method m2()
    method m3()
}
```

Show the layout of class E along with its Vtbl. As in class, method overriding, polymorphism, and dynamic binding must be supported. Use the notation m_A_B to denote a method m declared in A and defined in B.