Name: __________________________
Student ID: ______________________

Signature: ________________________

• You have 75 minutes to write the 6 questions on this examination. A total of 75 marks is available.

• **Justify all of your answers**

• You may use one sheet of handwritten notes.

• Keep your answers short. If you run out of space for a question, you have written too much.

• The number in square brackets to the right of the question number indicates the number of marks allocated for that question. Use these to help you determine how much time you should spend on each question.

• Use the back of the page for rough work.

• Write legibly; if I cannot read it you cannot have any points!

• **Good luck**

**UNIVERSITY REGULATIONS:**

• No candidate shall be permitted to enter the examination room after the expiration of one half hour, or to leave during the first half hour of the examination.

• **CAUTION:** Candidates guilty of any of the following, or similar, dishonest practices shall be immediately dismissed from the examination and shall be liable to disciplinary action.

  1. Making use of any books, papers or memoranda, electronic equipment, or other memory aid devices, other than those authorized by the examiners.

  2. Speaking or communicating with other candidates.

  3. Purposely exposing written papers to the view of other candidates. The plea of accident or forgetfulness shall not be received.

<table>
<thead>
<tr>
<th>Question</th>
<th>Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>/19</td>
</tr>
<tr>
<td>2</td>
<td>/20</td>
</tr>
<tr>
<td>3</td>
<td>/6</td>
</tr>
<tr>
<td>4</td>
<td>/15</td>
</tr>
<tr>
<td>5</td>
<td>/9</td>
</tr>
<tr>
<td>6</td>
<td>/6</td>
</tr>
<tr>
<td>Total</td>
<td>/75</td>
</tr>
</tbody>
</table>
First & Follow Sets

Question 1 [19 points]

Consider the following context-free grammar G (Note, .. is one token):

\[
\begin{align*}
A & \rightarrow T \ [ \ I \ ] \ D \ ; \\
T & \rightarrow \text{int} \ | \ \text{real} \ | \ \text{void} \ | \ A \\
I & \rightarrow \text{number} \ .. \ \text{number} \ | \ \epsilon \\
D & \rightarrow \ = \ \text{number} \ | \ \epsilon
\end{align*}
\]

- [8/19] Compute FIRST(A), FIRST(T), FIRST(I), and FIRST(D).

- [8/19] Compute FOLLOW(A), FOLLOW(T), FOLLOW(I), and FOLLOW(D).

(Question 1 continued on next page)
First & Follow Sets

• [3/19] Consider the following productions from a context free grammar.

\[
\begin{align*}
A & \rightarrow B \ a \\
B & \rightarrow c \mid \epsilon \\
C & \rightarrow B \ B
\end{align*}
\]

It is clear that \( \text{FIRST}(B) = \{c, \epsilon\} \). Explain why \( \epsilon \notin \text{FIRST}(A) \), and why \( \epsilon \in \text{FIRST}(C) \).
Question 2 [20 points]
Recall that shift/reduce parsers have 2 different actions: either they can shift a token from input on to the stack \((\text{shift } s)\), or they can reduce the handle on the stack by a production \((\text{reduce } A \rightarrow \alpha)\).

• [4/20] Explain why an LR parser that works like this always mimics a reverse right most derivation. (Hint: What is the relationship between the location of the handle on the stack and the configuration of the parser after the reduction has been performed).

• [3/20] What is the different between a reverse right most and a left most derivation?
Recall that we can eliminate common sub expressions like the following:

\[ A \rightarrow a \ b \ h \ C \]
\[ | \quad a \ b \ g \ K \]

by changing it to

\[ A \rightarrow a \ b \ A' \]
\[ A' \rightarrow h \ C \ | \ g \ K \]

in order for the grammar to be LL(1), i.e., parsable with one look ahead.

• [4/20] Explain how the number of look aheads (for LL(1) we look ahead one, i.e. we may look at the first token on input) relates to the necessity of common subexpression elimination. (Hint. If the longest common subexpression of a grammar consists of \( k \) tokens (and we don’t have any left recursion) then is this grammar LL(\( k \))?)

Consider the grammar from the first/follow set question again:

\[ A \rightarrow T \ [ \ I \ ] D ; \]
\[ T \rightarrow \text{int} \ | \ \text{real} \ | \ \text{void} \ | \ A \]
\[ I \rightarrow \text{number} \ .. \ \text{number} \ | \ \epsilon \]
\[ D \rightarrow = \ \text{number} \ | \ \epsilon \]

• [3/20] Show a rightmost derivation of the string \texttt{real [ 8 .. 19 ] = 89 ;}
Derivations and Parsing

• [3/20] Show a leftmost derivation of the same string.

• [3/20] Are the parse trees the same? (Justify your answer)

(End of question 2)
Closures

**Question 3** [6 points]

• [3/6] Let $G$ be a context free grammar. Explain why

$$\text{closure}(\{[A \to \alpha a\beta]\}) = \{[A \to \alpha a\beta]\}$$

• [3/6] Recall that in the definition of the kernel for an item set we determined that no items of the form $[A \to \cdot \alpha]$ were ever included, except for one, namely the item $[S' \to \cdot S]$; Explain why no other items of this form are included in the kernels, and why the one with the unique start symbol $S'$ is. You might want to start by explaining what the kernel of an item set is.

(End of question 3)
LL(1) Parsers

**Question 4** [15 points]

Let $G$ be the following context free grammar in EBNF form:

$$
I \rightarrow \text{id} \ (P) \\
P \rightarrow E \ P' \mid \epsilon \\
P' \rightarrow , \ E \ P' \mid \epsilon \\
E \rightarrow 0 \mid 1 \mid \text{id}
$$

And the following FIRST and FOLLOW sets:

<table>
<thead>
<tr>
<th>$X$</th>
<th>$FIRST(X)$</th>
<th>$FOLLOW(X)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I$</td>
<td>{ id }</td>
<td>{ $ }$</td>
</tr>
<tr>
<td>$P$</td>
<td>{ 0, 1, id, $\epsilon$ }</td>
<td>{ ( ) }</td>
</tr>
<tr>
<td>$P'$</td>
<td>{ , , $\epsilon$ }</td>
<td>{ ( ) }</td>
</tr>
<tr>
<td>$E$</td>
<td>{ 0, 1, id }</td>
<td>{ , , }</td>
</tr>
</tbody>
</table>

| $\text{id} (P)$ | { id }                  |                     |
| $E \ P'$        | { 0, 1, id }            |                     |
| $,$ $E \ P'$    | { , }                   |                     |
| $0$             | { 0 }                   |                     |
| $1$             | { 1 }                   |                     |
| $\text{id}$     | { id }                  |                     |

- [10/15] Fill in the following parse table:

<table>
<thead>
<tr>
<th></th>
<th>id</th>
<th>( )</th>
<th>,</th>
<th>0</th>
<th>1</th>
<th>$$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P'$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$E$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Question 4 continued on next page)
LL(1) Parsers

- [5/15] show the parsing of $f(0, 1)$ using the table that you constructed on the previous page. The column labelled ‘Production’ should list the production that you use.

<table>
<thead>
<tr>
<th>Stack</th>
<th>Input</th>
<th>Production</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I$</td>
<td>$f(0, 1)$</td>
<td>$I$</td>
</tr>
</tbody>
</table>

(End of question 4)
Espresso

**Question 5** [9 points]

- [3/9] Phase 3 of the Espresso compiler consists of 2 passes, explain why we cannot do the necessary checking in just one pass.


- [2/9] Even though it is strongly typed, if we were to be completely honest, there are a few things in the language that break the strongly typed properties. Can you name at least one thing that cannot be checked at compile time?

- [2/9] If Espresso used dynamic scoping rather than static scoping, how would that affect Phase 3 (Name Checking)?

(End of question 5)
LR Parsing

Question 6 [6 points]

• [3/6] Using an item set, explain (show) what causes a shift/reduce conflict.

• [3/6] Can you explain the necessary conditions for a reduce/reduce conflict between the two items:

[A → α·]

and

[B → β·]