Compilers [Fall 2017]
Test II

NAME: _____

Instructions:

1) This test is 7 pages in length.

2) You have 75 minutes to complete and turn in this test.

3) Essays will be graded on how clearly you've communicated the necessary ideas. Write in complete English sentences.

4) This test is closed books, notes, papers, friends, phones, neighbors, smartwatches, etc.

5) Use the backs of pages in this test packet for scratch work. If you write more than a final answer in the area next to a question, circle your final answer.

6) Write and sign the following: "I pledge my Honor that I have not cheated, and will not cheat, on this test."

_____

_____

Signed: _____

1. [25 points] [Essay]
Define a lexically and syntactically valid, but semantically invalid, DJ program P, and write an essay explaining how something "insecure" could happen if P were executed.

2. [20 points]
Write a flex rule (i.e., regular expression) for tokenizing floating-point literals, which are defined as follows:

- A float may optionally begin with a plus or minus sign.
- A float may optionally end with an exponent portion. Exponent portions of floats must begin with the character e or E, then may contain a plus or minus sign, and then must contain a nonempty sequence of digits.
- A float may contain any natural number of digits but must contain at least one digit.
- A float must contain at least one digit before the exponent portion, if one is present.
- A float may optionally contain a decimal point. Any natural number of digits may appear before a decimal point, but at least one digit must appear after a decimal point. If the float contains both a decimal point and an exponent portion, then at least one digit must appear between the decimal point and the exponent portion.
- The only sequence of digits within a float allowed to have leading 0s is the portion after a decimal point (but before an exponent portion if present). Note that 00 and 02 have leading 0s but 0 and 2 do not.

Keep your response simple enough that it cannot be simplified in any significant ways.

3. [25 points]
For each kind of conflict an LALR parser may have, define a grammar G such that G causes an LALR parser to have that conflict. After defining each G, show enough of G's LALR DFA to pinpoint the conflict. Finally, show one more grammar for which the LALR parser has no conflicts, and show an LALR parse trace of a valid input according to this final example grammar. Always ensure that your example grammars are free of duplicate rules and rules of the form N::=N.

4. [30 points]

G' is:
```
0    S -> E$
1    E -> E * E
2    E -> n
3    E -> P
4    P -> (E)O
5    O -> P
6    O -> ε
```

a) Which of the grammatical transformations discussed in class has G' undergone?

b) Draw an LL(1) parse table for G'.

c) Define pseudocode to implement an LL(1) parser and AST-builder for the P and O nonterminals in G'. Use whatever reasonable helper functions you believe should exist.

G' is:
```
0    S -> E$
1    E -> E * E
2    E -> n
3    E -> P
4    P -> (E)O
5    O -> P
6    O -> ε
```

d) Define G'', which is G' with the left-recursion removed.

e) G'' has unnecessary complexity. With just one simplification G'' will be in LL(1). Simplify G'' to obtain an equivalent G''' that is in LL(1).

Undergraduates stop here. The remaining problem is for graduate students.

G' is:

```
0    S -> E$
1    E -> E * E
2    E -> n
3    E -> P
4    P -> (E)O
5    O -> P
6    O -> ε
```

f) [5 points]
Prove or disprove that G' is in RR(1).