CS 164, Spring 2019 **CS 164: Homework #3** **P. N. Hilfinger**

Due: Wednesday, 20 February 2018 at 2400

To merge the skeleton files into your master homework repositorry, use the comnands

```
git checkout master  # If not already on that branch
git fetch shared
git merge shared/hw3
```

This assumes you have set up your repository according to instructions for using Git in CS164 and that you have first added and committed any modified files.

Unless the problem specifies otherwise, please put your solutions in a file named `hw3/hw3.txt`.

1. [From Aho, Sethi, Ullman] Indicate what language is described by each of the following grammars. In each case, S is the only non-terminal. Some symbols are quoted to make it clear that they are terminals.

a. S ::= 0 S 1 | 0 1

b. S ::= + S S | - S S | a

c. S ::= S `"("` S `")"` S | ϵ

d. S ::= a S b S | b S a S | ϵ

e. S ::= a | S + S | S S | S `"*"` | `"("` S `")"`

2. Identify each ambiguous grammar in problem 1 above, and give an unambiguous grammar that recognizes the same language (any such grammar—don't worry about associativity or precedence, since there are no semantic actions.)

3. For 1d above, give two distinct leftmost derivations for the string *abab*. For each derivation, show the corresponding parse tree and the rightmost derivation for that same parse tree.

4. [From Aho, Sethi, Ullman] Show that all binary (base 2) numerals produced by the following grammar denote numbers that are divisible by 3:

N ::= 11 | 1001 | N 0 | N N

Does this grammar generate all positive binary numerals that are divisible by 3?

5. [From Aho, Sethi, Ullman] Try to design a context-free grammar for each of the following languages (it is not always possible). Whenever possible, make it a regular grammar.

a. The set of all strings of 0's and 1's where every 0 is immediately followed by at least one 1.

b. Strings of 0's and 1's with an equal number of 0's and 1's.

c. Strings of 0's and 1's with an unequal number of 0's and 1's.

d. Strings of 0's and 1's that do not contain the substring 011.

e. Strings of 0's and 1's of the form xy where x and y are equal-length strings and $x \neq y$.

f. Strings of 0's and 1's of the form xx.

6. Write a BNF grammar describing the language of boolean expressions whose value is `true`. The terminal symbols are '1' (true) , '0' (false), '`*`' (logical and), '`+`' (logical or), unary '`-`' (logical not) and left and right parentheses (for grouping). Assume the usual precedence rules, with logical "not" having highest precedence. That is, `1`, `1*1`, `1+0`, `1*1*-(0+1*0)`, and `-0` are all in the language, while `0`, `0+0`, prog—0*1—, and `1*1*(0+1*0)` are not. Your grammar may be ambiguous (that is, you may specify operator precedence and associativity separately). Put your solution in a file `6.y`. Start with the skeleton in `6.y` in the files for this homework assignment.

This problem uses C++, BISON and FLEX, which you'll have to install if you don't have them, or simply use your instructional account (you can ssh in and run it).

7. The skeleton file `7.cc` provides a C++ framework for a top-down parser that accepts the same tokens as for problem 6, above. The lexer interface—the functions `next`, `scan`, and `ERROR`—are defined as in lecture. Fill in the skeleton with a recursive-descent parser that effectively does the same thing as the grammar you developed for problem 6. However, rather than build a recognizer where false statements cause a parsing error, build one that returns true or false as the semantic value of the top level node, depending on whether the sentence is true or false.