Homework #4

Due Wednesday, September 19

1. Read section 1.4 of the Enderton handout, which discusses unique readability for propositional formulas. Start reading section 1.3 of van Dalen.

2. Consider the following inductive definition of the set of all "AB-strings":

   - $\emptyset$, the empty string, is an ab-string
   - if s is an AB-string, so is $f_1(s)$
   - if s is an AB-string, so is $f_2(s)$.

   In the "correct" interpretation, the underlying set U is a set of strings, and f_1 and f_2 are functions that prepend the letters "A" and "B" respectively. However, if instead we take U' to be the set of strings of stars (e.g. "*****"), let f_1 be a function that prepends one star, and let f_2 be the function that prepends two stars, then the smallest subset of U' the contains $\emptyset$ and is closed under f_1 and f_2 is *not* generated freely.

   Come up with better functions f_1 and f_2, so that they still act on the underlying set U', but make the resulting set of "ab-strings" freely generated.

⋆ 3. Recall the definition of "arithmetic expressions" I gave in class:

   - any string of digits that doesn't start with "0" is an arithmetic expression
   - if s and t are arithmetic expressions, so is "$(s+t)$" (more precisely, "("$\hat{}s\hat{}$"+"$\hat{}t\hat{}$")")
   - if s and t are arithmetic expressions, so is "$(s \times t)$".

   Let $length(s)$ denote the length of s, and let $val(s)$ denote the evaluation function I defined in class. Prove by induction that for every expression s, the inequality $val(s) \leq 10^{length(s)}$ holds.

∘ 4. What would happen to the previous theorem if we were to add exponentiation, $a \uparrow b$?

⋆ 5. The set of propositional formulas in prenex form is defined inductively, as follows (the underlying set consists of strings of variables and logical symbols):

- $\perp$ is a prenex formula
- any variable p_i is a prenex formula
- if φ is a prenex formula, so is $\neg\varphi$
- if φ and ψ are prenex formulas, so is $\wedge\varphi\psi$
- if φ and ψ are prenex formulas, so is $\vee\varphi\psi$
- if φ and ψ are prenex formulas, so is $\rightarrow\varphi\psi$

Intuitively, this is just another notation for propositonal formulas in which the connectives come *in front* of the arguments, instead of *in between* them. For example, one writes $\wedge p_1 p_2$ instead of $(p_1 \wedge p_2)$. Notice, however, that in this representation no parentheses are used.

a. Convert $\vee\neg\rightarrow\wedge p_1 p_2 p_3 \wedge p_4 p_5$ to a regular propositional formula.

b. Convert $((p_1 \wedge p_2) \rightarrow p_3) \vee (\neg p_3 \rightarrow p_1)$ to a prenex formula.

c. Define a function recursively that maps prenex propositional formulas to regular ones (you can assume that the set of prenex formulas is freely generated).

d. Define a function recursively that maps regular propositional formulas to prenex ones.

6. Do problems 1 and 2 on page 14 of van Dalen.

$\star$ 7. Do problem 3 on page 14. In other words, show that if φ is a subformula of ψ, and ψ is a subformula of θ, then φ is a subformula of θ. (Hint: say that a subset A of $PROP$ is "closed under subformulas" if whenever a formula φ is in A, every subformula of φ is also in A. Show by induction formulas θ, that the set of subformulas of θ is closed under subformulas.)

8. Do problem 4 on page 14. In other words, show that if φ is a subformula of ψ and $\theta_0, \theta_1, \ldots, \theta_k$ is a formation sequence for ψ, then for some $i \leq k$, $\varphi = \theta_i$. Be precise: use the definitions of $PROP$, formation sequences, and subformulas presented in class.

$\circ$ 9. Do problem 5 on page 15.

10. Do problems 6 and 7 on pages 14–15 of van Dalen.

$\star$ 11. Do problem 9 on page 15 of van Dalen.

$\circ$ 12. Do problem 11 on page 15.

$\circ$ 13. Definition 2.3.1 says that C is freely generated (as a subset of U), if, *when restricted to C*, each f_i is injective and the ranges of the f_i's are disjoint from each other and from B. Certainly, if the functions f_i have these properties on U, they also have it on C; but give an example where the functions do *not* have these properties on U, but C is still freely generated.

∘ 14. Generalize the recursion theorem (2.3.2) so that in defining $F(s)$ one can use all elements of C that are "shorter" than s (for a given "length" function).

∘ 15. Prove unique readability for prenex formulas, i.e. that the set of prenex formulas is freely generated. This amounts to showing that there is only one way to "parse" a given formula.

∘ 16. In the programming language of your choice, define a data structure to represent propositional formulas as trees. (That is, a propositional formula is either a variable, or an operation with pointers to its arguments). Write a parser for propositional formulas, that is, a program that takes a string as input and turns it into a parse tree. The routine should print "ok" if successful, or "error" if the string is not a formula.

Now write routines that convert a formula to prenex form; that take an assignment of truth values to the variables as input and determine whether or not the resulting formula is true; and that determine whether there is *any* assignment that makes the formula true.