Baruch College
PHI 3010: Symbolic Logic and Computers
Semester: Spring Semester 2013
Instructor: R. Gregory Taylor
Office: VC 5–272B
Office hours: Tuesdays from 2:30 to 4:30 p.m. and by appointment
E-mail: gregory.taylor@baruch.cuny.edu

Texts (required): Warren Goldfarb, *Deductive Logic*, Hackett Publishing, Indianapolis, 2003. This text may be purchased or "rented" at the Baruch College bookstore. Either way, it is very inexpensive by the standards of modern textbook publishing. We will use the textbook extensively during the first half of the semester, although you need not bring it to class.

Course Description from Department Listing of Elective Courses for Spring 2012: PHI 3010 Symbolic Logic and Computers/This course is a natural follow-up to any course in Logic or any course in Finite Math. The goal of the course is to explain logical systems that relation to computer programs and operations. You will learn and prove the fundamental result in Computer Science: the Church–Turing theorem, that a computer can automatically solve any problem involving "or" and "not," but cannot automatically solve every problem involving "all" and "some." This theorem determines for all time the power of computers and computer algorithms.

First Reading Assignment (over the next two weeks or so): Goldfarb, Part I, pp. 3–87

Goals for the Course (What the Student Should Expect to Learn): Having completed this course, the student should (1) understood how the method of truth-tables provides an algorithm for determining whether an arbitrary sentence of the language of truth-functional logic is truth-functionally valid; (2) understand why certain sorts of formal derivation systems for truth-functional logic do **not** provide an algorithm for truth-functional validity; (3) understand why certain sorts of systems for quantificational (first-order) logic do **not** afford us any algorithm for first-order validity; finally, (4) understand how the Church–Turing Theorem can be taken to tell us that, in fact, no algorithm or ``programmable method'' for first-order validity is possible.

Course Content: The following are the most important topics that will be covered, some rather quickly, in the first half of the course (by around April 1, it is hoped):

- Core logical concepts: truth-values, arguments, deductive validity and soundness, induction, consistency, equivalence
- Truth-functional connectives; examples of connectives that are not truth-functional, e.g., $\mathbf{Bel}_A$ … meaning that agent *A* believes that … .
- Syntax (grammar) of truth-functional logic
- Truth-value assignments (to sentences)
- Truth-tables for sentences
- Truth-functional validity, satisfiability, unsatisfiability, and indeterminacy (contingency)
- Truth-functional equivalence of sentences
- Truth-functional consistency as a property of sets of sentences
- Truth-functional implication (as when one sentence "implies" another sentence)
- Derivations within the formal derivation system *T* described in *Goldfarb* pp. 79–87
- Proofs by induction on the complexity of sentences of truth-functional logic
- The soundness and completeness concepts for *T*
- Proving the soundness and completeness of the system *T*
- The informal concept of an *algorithm*
- The method of truth-tables as an algorithm for determining whether an arbitrary sentence of the language of truth-functional logic is truth-functionally valid; the method of truth-tables as an algorithm for

determining whether an arbitrary sentence of the language of truth-functional logic is truth-functionally (un)satisfiable

- Why the formal derivation system *T* does **not** provide an algorithm for truth-functional validity or for truth-functional (un)satisfiability
- The limitations of truth-functional logic with respect to logic's deductive task and with respect to logic's descriptive task
- How quantificational logic overcomes (many of) those limitations
- The concept of quantification
- Syntax (grammar) of quantificational (first-order) logic
- Quantifier scope
- Logical (first-order) validity, satisfiability, unsatisfiability
- A natural deduction system for quantificational logic (*Goldfarb* pp. 183–85 [six rules])
- Proofs by induction on the complexity of formulae of the language of quantificational logic
- Proving the soundness and completeness of Goldfarb's natural deduction system
- Why Goldfarb's natural deduction system does **not** afford us any algorithm (''programmable method'') for first-order validity or for first-order (un)satisfiability (Note that our merely seeing the Goldfarb's system affords us no algorithm for validity leaves open the question whether there might yet be some other system that **does** provide such an algorithm, or expressed differently, whether there might yet be some C-language program, using different ideas, that can decide whether an arbitrary input formula is logically valid.)

As mentioned above, the topics listed above will take us to around April 1 and to the end of Goldfarb's text. For the remaining weeks of the semester, the instructor will post lecture handouts on BlackBoard in advance. The student should print these and bring them to class. Topics to be covered in the last half of the course will be the following:

- ❖ The formal system *N* consisting of nine first-order axioms formalizing a classical system for the natural numbers (0, 1, 2, …) and based on Goldfarb's natural deduction system for first-order logic
- ❖ A few theorems of the formal system *N*, where a ``theorem'' is any provable formula of the language of the system *N*
- ❖ The concept of a recursive number-theoretic function ("building up from below")
- ❖ The characteristic function of a set of natural numbers
- ❖ The concept of a recursive predicate
- ❖ The Church–Turing Thesis/This is philosophical claim to the effect that the number-theoretic functions that are "computable," in an intuitive sense, are all and only those that are recursive in the technical sense; it is not to be confused with the Church–Turing *Theorem*, mentioned below
- ❖ Bounded quantification
- ❖ Rules for defining new recursive functions (predicates) in terms of functions and predicates previously established as recursive
- ❖ Encoding finite sequences of natural numbers as a single natural number (the concept of "Gödel numbering")
- ❖ Arithmetization of the syntax of the formal system *N*
- ❖ Representability of recursive functions and predicates in the formal system *N* (This and the previous two items are key steps in the celebrated Incompleteness Theorems of K. Gödel [1931])
- ❖ Proof that the set of (codes of) theorems of the formal system *N* is not a recursive set of natural numbers, from which we can then show:
- ❖ The Church–Turing Theorem stating that the set of (codes of) valid formulae of the language of first-order logic is not a recursive set of natural numbers
- ❖ Finally, if we assume, reasonably, the Church–Turing Thesis mentioned above, then the Church–Turing Theorem tells us that there is no algorithm for first-order validity, thus answering negatively the question raised above (Expressed differently, the Church–Turing Theorem plus the Church–Turing Thesis tell us that it is not possible to write a C-language program that can answer the question whether an arbitrary input formula is logically valid.)

The items listed above are intended to give the student some notion of what will be covered in the course. It is not assumed that the student has any familiarity whatsoever with these ideas.

It is hoped that we would complete the proof of the Church–Turing Theorem one or two weeks before the end of the semester. That might leave a few sessions to discuss either the Incompleteness Theorems of K. Gödel or other decision problems associated with first-order logic. Regarding the latter, it has been shown that, if we restrict attention to sets of first-order formulae of limited complexity, then there *do* exist algorithms for determining whether arbitrary formulae are valid or satisfiable. (Incidentally, author Warren Goldfarb played a major role in these investigations during the 1980s.)

Grading/Tests: In addition to the two-hour final examination, held during the scheduled examination period, there will be a midterm examination roughly half-way through the semester. Grades on these two tests will make up half or more of the student's grade for the course (see below). There will also be six or seven very short quizzes.

Written homework: There will be several written homework assignments.

Grade Computation: *Grade Computation*: The grade for the course will be determined as follows. The student's grade will be computed in two ways, first using

Quizzes 25%
Midterm examination 25%
Final examination 25%
Written homework assignments 25%

and then using

Quizzes 25%
Midterm examination 30%
Final examination 30%
Written homework assignments 15%.

The student's final grade will be the higher of those two computations. (For most students, the first will be the higher; however, in the case of the rare student who does little homework but nonetheless does well on quizzes and tests, the second will be higher.)

Students are expected to take tests at the scheduled hour. Latecomers will not be permitted extra time. Make-ups for those who miss a test will be permitted only under special circumstances. If it is necessary for you to miss a test because of your own illness or that of a family member, then it is important that you send the instructor e-mail explaining your absence *in advance*. Also, students will not be permitted to leave the classroom during a test, and no cell phones may be used in any manner during tests (or quizzes).

As for the quizzes, there are in general no make-ups for missed quizzes unless this has been arranged in advance or in the case of documented illness. On the other hand, the lowest quiz grade will be dropped. (The author's grading program does this automatically.)

Course Handouts: There will be regular handouts prepared by the instructor, a few at the beginning but more later. These will appear on BlackBoard and should be printed by students as they appear. All handouts should be brought to every class. There will not be so many that this will pose a burden to students traveling by public transportation. Early on, **the instructor will be verifying that students possess course notebooks with some sort of pocket for storing these handouts**, since students unable to locate handouts will be at a disadvantage.

Attendance: Class attendance is mandatory. An attendance sheet will be circulated during each class unless there is a quiz or collected writing assignment, in which case that quiz or writing assignment functions as a record of attendance. Students are permitted six discretionary absences, which is more lenient that the policy of Baruch College (four discretionary absences). Thus any student who has missed seven classes will immediately receive a grade of WU for the course. Classes missed during the first week are recorded and count fully as discretionary absences. Consequently, if a student enrolls in the course during the second week of the semester, then he or she is immediately recorded as having two discretionary absences.

For whatever reason, some students log tremendous numbers of absences during the first few weeks of class. Such students are unlikely to succeed in the course subsequently. Consequently, there is an exception to the instructor's policy of six permissible discretionary absences stated above. *Namely, any student who misses five of the first ten class sessions (including those of the first week) will immediately be assigned a grade of WU*.