Resume

Date: 13 October, 1992

1. While-programs.
   (a) Syntax.
   (b) Macros, appropriate and inappropriate use of macros.
   (c) Variable relabeling.
   (d) Equivalence with other "program forms".
2. Relationship between while-programs and functions.
   (a) Partial functions, domain of definition and extension ordering.
   (b) Input/output conventions for arity k functions.
   (c) Non-halting of a program versus undefinedness of a function.
   (d) Church's thesis, the definition of a computable function.
   (e) Are all functions computable?
3. Enumeration of computable functions.
   (a) Definition of countability: in bijection with the naturals.
   (b) Counting while-programs, encodings.
   (c) Introduction to uncountability.
      i. The uncountability of the reals and of partial functions.
      ii. Cantor diagonalization.
   (d) Cardinality argument for the existence of uncomputable functions.
   (e) The halting function is uncomputable.
4. Universal Functions.
   (a) Program rewriting: the string operations head, tail and concatenation as number-theoretic functions.
   (b) Syntax checking.
   (c) Simulation of unbounded memory.

i. Pairing functions: a bijection from pairs of naturals to the naturals.
ii. Extensions of pairing functions to n-tuples and ∞-tuples of finite support.

(d) The universal function is computable.

- **Second Half of the Course.**

5. Recursion
   (a) Recursive programs are while-program computable.
      i. Posets of functions by extension ordering and least-upper-bounds.
      ii. Recursive programs are least-upper-bounds.
      iii. Effective least-upper-bounds are computable.
   (b) The Recursion Theorem, Section 6.1.
6. Acceptable Programming Systems.
   (a) Roger's Isomorphism Theorem, Section 6.2.
   (b) Turing Machines.
   (c) Recursive Functions.
      i. Primitive recursive functions and loop-programs.
      ii. Unbounded minimization and partial recursive functions.
   (d) Undecidability of certain word problems.
7. The Theory of NP-Completeness
   (a) Deterministic and Non-deterministic Turing machines.
   (b) Polynomial-time transformations. Classes P and NP.
   (c) SAT is NP-Complete.
   (d) Other NP-Complete problems.