1. **Department, number and title of course:** Electrical Engineering and Computer Sciences: CS 47A – Completion of Work in Computer Science 61A, Interpretation of Computer Programs

2. **Catalog Description:** (1 unit, self-paced, graded) Implementation of generic operations. Streams and iterators. Implementation techniques for supporting functional, object-oriented, and constraint-based programming in the Scheme programming language. Together with 9D, 47A constitutes an abbreviated version of 61A for students who have already taken a course equivalent to CS 61B.

3. **Prerequisites:** 61B or equivalent, plus CS 9D, plus consent of instructor. Students will not receive credit for CS 61A taken after CS 47A, or for CS 47A taken after CS 61A.

4. **Textbooks and/or other required material:**

5. **Course objectives:** This self-paced course continues from where CS 9D left off, providing an abbreviated version of CS 61A for students who have already taken a course equivalent to CS 61B. Course activities include programming assignments and quizzes; quizzes focus on low-level language details or programming techniques, while programming assignments are broader in scope. The list of programming assignments appears below. All assignments use the Scheme programming language.

6. **Topics Covered:**
   - Rule-based programming: Students experiment with a rule interpreter and associated pattern matcher, and then write a set of rules that does simple type inference on a Scheme program.
   - Generic operations: Students first implement the set-of-integers data type both as a list of integers and as a sorted list of intervals. Then they convert these implementations to use data-directed programming. Finally, they recode the implementations in message-passing style.
   - Iterators and lazy evaluation: Students define Scheme streams to represent infinite data types and to implement iterators through multi-dimensional structures.
   - Scheme interpreter: Students work with and extend a Scheme interpreter written in Scheme.
   - Efficiency improvements: Students explore the efficiency benefits of mutation and vectors compared to side-effect-free programming using lists.

7. **Class/laboratory schedule:** Self-paced.

8. **Contribution of course meeting the professional component:** This course contributes to engineering topics; it is approximately 60% science and 40% design.

9. **Relationship of course to program objectives:** Students gain the ability to analyze and solve computer science and engineering problems through application of fundamental knowledge of mathematics, science, and engineering. Students learn to apply modern skills, techniques and tools to create a system to meet desired needs.

10. **Prepared by:** Professors Michael Clancy and Brian Harvey, 23 February 2006.