CS154: Introduction to Automata and Complexity Theory **Fall 1997**

Assignment #6

Due: Wednesday, November 26th, 1997.

Problem 1 (5 points) Since some of you were wondering about the utility of proving problems undecidable, let's show that many compiler optimization questions are undecidable. Determine which of the following problems are decidable and which aren't:

1. Dead Code Elimination: Given a program, does it have a line that is never reached?
2. Security Breach Detection: Given a program, does it ever read the `/etc/passwd` file?
3. Time Sink: Given a program, will it run for more than a million steps on any input?

For each of the above, either *outline* how to decide the problem or *outline* how you would reduce a known undecidable problem to it. The problems above are deliberately informal. It is up to you to define the corresponding language and argue whether it is decidable or not. You may assume the programs are given as C programs.

Problem 2 (5 point) Which of the following statements are true? Briefly justify your answer.

1. $n^n = O(2^n)$.
2. $\log(n^5) = O(\log n)$.
3. $\log^5 n = \Omega(\log n)$.
4. $2^n + n^{1000} = \theta(2^n)$.
5. if $f(n) = o(\log n)$ then $2^{f(n)} = O(n)$.