York University
CSE 4111 Fall 2009
Instructor: Jeff Edmonds
Steps Reductions-Halting

1. Let $P = \{\langle\text{"}M\text{"}, I\rangle \mid \text{TM } M$ prints "Hi" on input I at some point in its computation $\}$,
   i.e. computational problem P says "yes" on inputs $\langle\text{"}M\text{"}, I\rangle$ in this set and says "no" on inputs not in this set.
   We will prove that P is undecidable by proving: $Halting \leq_{compute} P$.
   Suppose I have an oracle that decides P.
   Here is an algorithm that will decide the Halting Problem.
   Given an input $\langle\text{"}M\text{"}, I\rangle$,
   I construct another TM M_M.
   This TM has M hard wired into it.
   M_M on input I does the following.
   Run M on I suppressing any out.
   Print ("Hi")
   I give $\langle\text{"}M\text{"}, I\rangle$ to my oracle and if the oracle says "yes", then I say "yes" and if it says "no", then I say "no".
   I prove that my algorithm works as follows.
   Suppose M halts on I.
   Then M_M then completes its simulation of M on I
   and goes on to print "Hi".
   Then the oracle says "yes".
   Then I say "yes".
   Hence I gave the correct answer.
   Suppose M runs forever on I.
   Then M_M runs forever in its simulation of M on I and as such never prints "Hi".
   Then the oracle says "no".
   Then I say "no".
   Again I gave the correct answer.
   This proves that if P is decidable, then the Halting Problem is decidable.
   However, the Halting Problem is not decidable and hence P is not decidable.
   Note that you cannot directly use Rice's Theorem to prove this, because membership of "M" in P does not depend on the language M accepts but on whether M outputs "Hi".

2. Let $P = \{\text{"}M\text{"} \mid$ The language $L(M)$ accepted by M is regular $\}$,
   i.e. There exists a FSA A such that for all inputs $M(I) = A(I)$
   and there is a regular expression like 0^*1^* that accepts this language $L(M)$.
   We will prove that P is undecidable by proving: $Halting \leq_{compute} P$.
   Suppose I have an oracle that decides P.
   Here is an algorithm that will decide the Halting Problem.
   Given an input $\langle\text{"}M\text{"}, I\rangle$,
   I construct another TM $M_{\langle M,I\rangle}$.
   This TM has M and I hard wired into it.
   $M_{\langle M,I\rangle}$ on input I' does the following.
   If I' has the form 0^n1^n, then $M_{\langle M,I\rangle}$ halts and accepts.
   else $M_{\langle M,I\rangle}$ simulates M on I and halts and accept if M halts.
   I give "$M_{\langle M,I\rangle}$" to my oracle and if the oracle says "yes", then I say "yes" and if it says "no", then I say "no".
   I prove that my algorithm works as follows.
   Suppose M halts on I.
   Then $M_{\langle M,I\rangle}$ halts and accepts every input I'.
   Then the language $L(M)$ accepted by M contains every string.

This language is regular as demonstrated by the regular expression $\{0,1\}^*$.
Then the oracle says "yes".
Then I say "yes".
Hence I gave the correct answer.
Suppose M runs forever on I.
Then $M_{\langle M,I\rangle}$ halts and accepts strings I' of form 0^n1^n, but runs forever on all other strings.
This language $L(M) = 0^n1^n$ is known not to be regular.
Hence, the oracle says "no".
Then I say "no".
Again I gave the correct answer.
This proves that if P is decidable, then the Halting Problem is decidable.
However, the Halting Problem is not decidable and hence P is not decidable.
Alternatively, you could directly use Rice's Theorem to prove.
For every pair of TMs, if $L(M_1) = L(M_2)$ then $P(M_1) = P(M_2)$.
There are TM for which $P(M)$ is yes and there are TM for which $P(M)$ is no.