Last time:

$A_{TM} := \{\langle M, w\rangle : M \text{ a TM}, w \text{ a string}, \ \& \ M \text{ accepts } w\}$

Thm: A_{TM} is undecidable.

Def:

$HALT_{TM} := \{\langle M, w\rangle : M \text{ a TM}, w \text{ a string}, M \text{ halts on input } w\}$

Prop: $HALT_{TM}$ is undecidable.

Proof: Suppose $HALT_{TM}$ is decided by some TM D (a decider). Then we will use D as a subroutine to decide A_{TM} as follows:

"On input $\langle M, w\rangle$ where M is a TM & w is a string:

1. Modify M into a TM M' by replacing q_{rej} by a non-halting state that self-loops on all symbols (make rej. state of M' unreachable)

// If M accepts then M' accepts
// If M does not accept, then M' loops.

2. Run D on input $\langle M', w\rangle$
3. If D accepts in step 2 then accept; $\underbrace{\text{else reject.}}_{\text{ok b/c } D \text{ rejects}}$"

This TM is a decider that: accepts iff M accepts w and rejects otherwise, so it decides A_{TM} ↯

$\therefore$ $HALT_{TM}$ is undecidable. //

Def: An enumerator is a modified TM that

1. takes no input (all tapes start off blank)
2. Runs forever (does not halt)
3. Has one of its tapes be its printing tape [one-way write-only]
4. Has a designated printing state q_{print}

such that, whenever q_{print} is entered, the nonblank contents of the printing tape is "printed" (then erased & head reset).

what we mean by an enumerator printing a string.

Can have any # of other tapes (work tapes).

Def: Let E be an enumerator with printing alphabet Σ. Define $L(E)$ the language enumerated by E to be

$L(E) := \{w \in \Sigma^* : E \text{ prints } w \text{ at least once}\}$

Say that a language is enumerable (computably enumerable, recursively enumerable) if $L = L(E)$ for some enumerator E.

Thm: A language is T-rec iff it is enumerable.

Proof: ($\Leftarrow$) Suppose E is an enumerator. We define a TM M that recognizes $L(E)$ as follows:

$M :=$ "On input w:

1. Run E
2. If E (ever) prints w, then accept"

[If $w \notin L(E)$ then M loops on w.]

$w \in L(E) \Rightarrow E$ prints w at some point $\Rightarrow M$ accepts w

$w \notin L(E) \Rightarrow E$ never prints w $\Rightarrow M$ loops on w // ($\Leftarrow$)

($\Rightarrow$) Let M be TM

we design an enumerator E such that $L(E) = L(M)$ as follows:

$E :=$ "On no input:

1. Cycling through all strings of the form $\langle w, t\rangle$ where w is a string and $t \in \mathbb{N}$: ("time slicing")
   a) Run M on input w for t steps
   b) If M accepts w within t steps, then print w
2. Go on to the next pair $\langle \cdot, \cdot \rangle$"

[Correctness proof next time.]