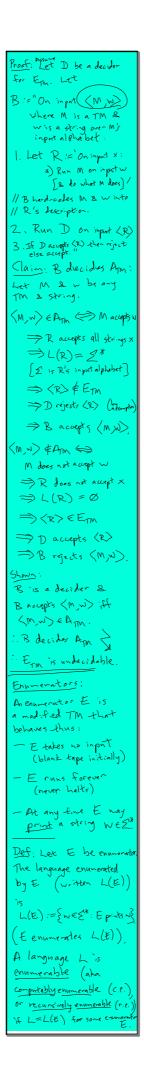
```
Theorem: Am is undecidable
  Recall:
  Arm := {< M, w> : M is atm
                                          in put w}
 Proof: Suppose Am 13
  decidable. Let In D
  be a decider for ATM.
 Consider the following TM
 B := "On input (M) where
                M is a TM
     1. Form the string
        (B) (B) M's rentombet
   2. Run D on input

\( \lambda \, \lambda \, \lambda \)
\( \lambda \, \lambda \, \lambda \)
\( \lambda \, \lambda \, \lambda \, \lambda \)
\( \lambda \, \lambda \, \lambda \, \lambda \, \lambda \)
\( \lambda \, \lambda \, \lambda \, \lambda \, \lambda \, \lambda \)
\( \lambda \, \
      then reject, else accept."

Brijects (8) Baccepts (2)
 Consider What B does
 on input (B)
 D accepts (B (B))
     (B) B accepts (B) (by assumpting)
  but > B rejects (B) (by B's o
Drejects (8)
                   B rujects (B) by assumpt
              B accepts (B) (by B's
    . Our assumption that D
    exists leads to a contractor
    so no such D can exist.
 Use: To show a lang. L
 undecidable, we
  "reduce" Arm to L:
 Any decider for L can be used to decide ATM &
  E_{TM} := \{ \langle M \rangle : M \text{ is a TM} \\ & L(M) = \emptyset \}
 Prop: ETM is undecidable.
Proof: Pesset D be a decider
 for Em. Let
 Bi="On input (M, W)
             Where M is a TM &
               wis a string over M's input alpha bet:
    1. Let R := 'On input x:
                       a) Run M on impact w
[& do what M does]/
 1/B hard-coles M & w into
1/ R's description.
2. Run D on input (R)
     [ & do what D does ]
Claim: B decides Am
```



```
Thm: A language is ensure able 15 it is
T-recognizable
Proof; Fix a language
L S Ex
(>)! Assume L is
enumerated by some enumerator E (L=L(E))
Let M be the following
TM;
 M:= "On input w:
 1, Run E.
 2. If E ever prints W, then accept
    [else loop]
Claim that M recognites L
 w arbitrary string.
NEL = E prints W (at south
      € M accepts N.
 . M recognizes L.
 [M either accepts or loops, so not a decider]
 Lis Torcognizable (by M)
(E) Assume L is Free,
Let N be a TM recognizing
 L. Let E be the
 following enumerator:
E:= "On no inpot;
 1. Cycle through all strings
of the form (w,t)
    where w is a strong
(over m's input alphabet)
& t is a natural number:
 2. For each such string (v,t)
   (a) Run M on input w
      for t steps,
within t steps
then print w "
Claim: Yw, we LE
Eprints w.
(E) obvious. E never prints
a stringfunless it sees that
Maccepts w, i.e., it weL
( Assume WEL. Thon M accepts Win, say, to
 steps for some to.
 Then when E gets to
 the pair (w, to), it
 notices that M accepts w,
 SO & prints W.
  WEL @ E prints W
: L = L(E), (7/1)
Thm: Let L be any
language. If L & [
are both T-rec, the L
is decidable,
Proof. Let E_1, E_2 be enumerators such that L=L(E_1)
T = L (to)

Decider For L:

D="On injust W:

J. Run E, & Ez in tandom.

2 If E, guide w, then accept

3 If Es prints w, then right"
```