I 10 (a)  

PCP \leq_m \text{ Halting prob.}

Typical instance, \(E=(x,y), \{E, y\} \) \((M, z)\).

Given \(E\), transform it to \((M, z)\) s.t. \(E\) has a solution if TM \(M\) halts on \(z\).

Fix \(z = \text{blank tape}\).

Description of \(M\):

(a) On any input other than blank tape, we don't care---say it halts.

(b) On blank tape: \(M\) checks whether

\[ z_1 = y_1; \text{then } z_2 = y_2; \ldots; \text{then } z_n = y_n; \text{ then} \]

\[ z_{n+1} = y_{n+1}; \text{then } z_{n+2} = y_{n+2}; \ldots; \text{then } z_{2n} = y_{2n}; \text{ then}\]

If one of them results in equality, \(M\) halts.

If \(E\) has a solution, \(M\) will detect it & halt.

If \(E\) has no solution, \(M\) will run forever.

Hence the transformation is correct. Note also that the transformation is a computable function.

\[
\begin{array}{c}
(\langle d \rangle) \rightarrow \text{ input } \rightarrow \text{ output } \rightarrow n = n+1
\end{array}
\]

We will give full credit for the above.

A TM enumerator is given below.
(b) \{ M \mid M \text{ halts on input } 101 \}

Generate \([M_1, M_2, \ldots]\) and develop their computations on input 101. Output all \([M_i]\) that halt.

More detail:

At step 1, generate \([M_i]\), perform one step of its simulation on input 101. Place the status in a queue.

At step 2, generate \([M_i]\), perform one more step of simulation for every process in the queue. Also perform one step of \([M_i]\) on input 101 and place its status in the queue.

If any \(M_i\) halts, output \([M_i]\), remove its process from the queue.

Prob 12: (c) If \(A\) and \(B\) are 1.e then \(A \not\equiv B\).
Start the enumerators of A & B and enumerate them in parallel:

\[ x, x_2, \ldots \] for A

\[ y, y_2, \ldots \] for B

a) When any \( x_i \) comes out of \( X \), check whether it is one of the \( y_i \)'s already enumerated. If so, enumerate \( x_i \).

b) When any \( y_i \) comes out of \( Y \), check whether it is one of the \( x_i \)'s already enumerated. If so, enumerate \( y_i \).

Note: \( A \cup B \) is r.e.

Proceed as above. Enumerate all the \( x_i \)'s that come out of \( X \) & all the \( y_i \)'s that come out of \( Y \).