1. Use diagonalisation to show that the membership problem is undecidable. That is, show that the following language is not recursive:

\[ \text{MP} \overset{\text{def}}{=} \{ M\#x \mid x \in L(M) \} \]

**Solution.** Suppose that there is a total TM \( K \) that accepts MP. If we were to draw the corresponding table, then the TM \( N \) that “negates the diagonal” would work as follows: \( N \) on input \( x \) constructs \( M_x \) and simulates the running of \( K \) on \( M_x\#x \). If the simulation ends with \( K \) accepting, then \( N \) rejects, otherwise \( N \) accepts.

Then \( N \) is a perfectly good TM, so it is \( M_n \) for some \( n \). In particular we have

\[ n \in L(M_n) \iff N \text{ accepts n } \iff K \text{ rejects } M_n\#n \iff n \notin L(M_n) \]

Which is a contradiction. What has gone wrong is that we assumed that \( N \) is in the table (that it is \( M_n \) for some \( n \)) but by its construction its behaviour is different from every row in the table at the diagonal entry. The conclusion is that no total \( K \) that treats all TMs can exist, so MP is undecidable.

2. Prove that \( \text{LP} \overset{\text{def}}{=} \{ M\#x \mid M \text{ loops on } x \} \) is not recursively enumerable.

**Solution.** We have proven that

\[ \text{HP} \overset{\text{def}}{=} \{ M\#x \mid M\#x \text{ are correct encodings of a TM and an input, and } M \text{ halts on } x \} \]

is recursively enumerable but not recursive. It is not difficult to see that

\[ \overline{\text{HP}} = \text{LP} \cup \{ M\#x \mid M\#x \text{ are correct encodings of a TM and an input.} \} \]

The second set in the union is recursive, so if \( LP \) were recursively enumerable so would \( \overline{\text{HP}} \), but we know that if a set and its complement are recursively enumerable then, in fact, it must be recursive. But \( \overline{\text{HP}} \) is not recursive, so \( LP \) cannot be recursively enumerable.

3. Construct a reduction from \( \text{LP} \overset{\text{def}}{=} \{ M\#x \mid M \text{ loops on } x \} \) to \( \{ M \mid L(M) = \emptyset \} \) to prove that the latter is not recursively enumerable.

**Solution.** We take \( M\#x \) to a TM \( M' \) that on input \( y \):

- (a) ignores its input \( y \);
- (b) simulates \( M \) on \( x \);
(c) if the simulation finishes then $M'$ accepts.

If $M$ loops on $x$ then $L(M') = \emptyset$ because step (a) never terminates. On the other hand if $M$ halts on $x$ then $L(M') = \Sigma^* \neq \emptyset$.

4. Show that the language $A \overset{\text{def}}{=} \{ M \mid L(M) = L(a^*) \}$ is not recursive by constructing a reduction from HP.

**Solution.** Recall that HP $\overset{\text{def}}{=} \{ M\#x \mid M \text{ halts on } x \}$. We need to describe how to get from a string $M\#x$ to a description of a TM $M'$ in a way that satisfies the property:

$$M \text{ halts on } x \iff L(M') = L(a^*).$$

This construction needs to be an algorithm – something for which we could construct a TM.

So, given $M\#x$ we can construct the TM $M'$ with behaviour as summed up below: $M'$ on input $y$:

(a) remembers $y$ (on the tape);
(b) simulates $M$ on $x$;
(c) checks whether $y \in L(a^*)$. Accepts if yes, rejects if no.

There is no magic going on. We know that we can do step (a) as TMs can simulate other TMs. Step (b) is simple: a TM can certainly do the job of a DFA!

Now if $M$ halts on $x$ then the simulation in the first phase of $M'$ will always stop and so $L(M') = L(a^*)$. If $M$ loops on $x$ then the simulation will not finish and so $L(M') = \emptyset \neq L(a^*)$. We have thus described a reduction from HP, proving that $A$ is not recursive.

5. Show that the language $\{ M \mid M \text{ loops on some input} \}$ is not r.e.

**Solution.** Reduction from LP.

We need to describe how to go from $M\#x$ to $M'$ such that ‘$M$ loops on $x$’ if and only if ‘$M'$ loops on some input’.

One construction that works is: $M'$ on input $y$,

(a) ignores its input $y$;
(b) simulates $M$ on $x$;
(c) if the simulation finishes then $M'$ accepts.

Now if $M$ loops on $x$ then the simulation in the first step does not terminate and so $M'$ loops on all inputs. On the other hand if $M'$ loops on some input then it must be that the simulation does not finish – i.e. $M$ loops on $x$. 

2
6. Show that the following problems are undecidable:

(i) given TMs $M$ and $N$, does $L(M) = L(N)$?  Hint: Use Question 4.

**Solution.** We know that it is easy to design a TM $M$ so that $L(M) = L(a^*)$. If the problem were decidable then $\{ N \mid L(N) = L(a^*) \}$ would be recursive (why?), but we know from Question 4 that it is not.

(ii) given TMs $M$ and $N$, does $L(M) \subseteq L(N)$?  Hint: Use Question 3.

**Solution.** If the above were decidable then the language $A \overset{\text{def}}{=} \{ M \mid L(M) \subseteq \emptyset \}$ would be recursive (why?). But we know from Question 3 that $A$ is not even recursively enumerable.

**Solution.** If it were then also (i) would be decidable since to prove that $L(M) = L(N)$ it is enough to check whether $L(M) \subseteq L(N)$ and $L(N) \subseteq L(M)$. Thus it is not decidable.

(iii) given TMs $M$ and $N$, is $L(M) \cap L(N) = \emptyset$?  Hint: Use Question 3.

**Solution.** Let $M$ be a total TM that accepts all inputs, ie $L(M) = \Sigma^*$. Then if the above were decidable then the language $B \overset{\text{def}}{=} \{ N \mid L(N) = \emptyset \}$ would be recursive (why?). But we know from Question 3 that $B$ is not even r.e.