Exercise [16.16]

ASSERTION:

A set $S \subseteq \mathrm{N}$ is *recursive* $\Leftrightarrow$ (I) *S* is *recursively enumerable*, and

(II) *N–S* is *recursively enumerable*.

PROOF:

First, I will introduce a symbolic element "#" that can be the output of a Turing machine if its input does NOT belong to a *recursively enumerable* set. To my mind, such a symbolic element is needed to include the empty set (and the whole set *N* of natural numbers) into the concept of recursively enumerable sets.

1. Part "$\Rightarrow$"

If $S \subseteq \mathrm{N}$ is recursive, there exists a Turing machine **T** with

$$T(n) = \begin{cases} 1 \;\; if \;\; n \in S \\ 0 \;\; if \;\; n \notin S \end{cases}$$

You may then define the following two Turing machines $\mathbf{T_E}$ and $\mathbf{T_{CE}}$:

$$T_E(n) = \begin{cases} n \;\; if \;\; T(n) = 1 \\ \# \;\; if \;\; T(n) = 0 \end{cases} \quad \text{and} \quad T_{CE}(n) = \begin{cases} \# \;\; if \;\; T(n) = 1 \\ n \;\; if \;\; T(n) = 0 \end{cases} \qquad (1)$$

$\mathbf{T_E}$ recursively enumerates *S*, while $\mathbf{T_{CE}}$ recursively enumerates N–*S*.[1]

2. Part "$\Leftarrow$"

Let $\mathbf{T_E}$ and $\mathbf{T_{CE}}$ be Turing machines that recursively enumerate *S* and *N–S*, respectively.

For any $n \in N$, you may then define the output of a new Turing machine **T** as follows:

[1] If S would be non-empty and S≠N, then one could use any default element x∈S and y∈S–N instead of "#" in the definition (1).

```
i=1;
BEGIN:
 IF         T_E(i) = n THEN T(n):= 1; STOP
 ELSE_IF    T_CE(i) = n THEN T(n):= 0; STOP
 ELSE i=i+1
GOTO BEGIN
```

(This algorithm feeds both $\mathbf{T_E}$ and $\mathbf{T_{CE}}$ successively with natural numbers i = 1; 2; 3; … until one of them produces the given number *n.* If *n* was produced by $\mathbf{T_E}$, then n belongs to the set *S*, and **T***(n)* is set to be "1"; if *n* was produced by $\mathbf{T_{CE}}$, then n belongs to the set *N–S*, and **T***(n)* is set to be "0").

The new Turing machine **T** shows that S is *recursive*.