Predicates can be used to define sets.

**Definition**

If $P(x_1, \ldots, x_n)$ is a predicate, the set $B$ defined by $P$ is

$$B = \{(x_1, \ldots, x_n) \mid P(x_1, \ldots, x_n) = \text{TRUE}\}.$$ 

$P$ is the characteristic predicate of the set $B$.

The set $B$ is defined as computable or recursive if its characteristic predicate is computable.

$B$ is primitive recursive if $P$ is a primitive recursive predicate.
In other words, $B$ is recursive if we can give a yes/no answer to the question "$x \in B$". This follows from the fact that $P$ is computable.
Example

The set

\[ B = \{ (x, y) \mid \text{the program } P \text{ with } \#(P) = y \text{ halts on } x \} \]

has \( \text{HALT}(X, Y) \) as its characteristic predicate. Since \( \text{HALT} \) is not computable, the set \( B \) is not recursive.
Definition

A set $B$ belongs to a class of functions if its characteristic predicate belongs to that set.
Theorem

Let $\mathcal{C}$ be a PRC class. If $B, C$ belong to $\mathcal{C}$, then so do the sets $B \cup C, B \cap C$ and $\overline{B}$. 

Proof.

If $P_B, P_C$ are the characteristic predicates of $B$ and $C$, respectively, and $P_B, P_C \in \mathcal{C}$, then the characteristic predicates of $B \cup C, B \cap C$ and $\overline{B}$ are $P_B \lor P_C$, $P_B \land P_C$, and $\sim P_B$, respectively, and we saw that they belong to $\mathcal{C}$. 

Theorem

Let \( C \) be a PRC class, and let \( B \subseteq \mathbb{N}^m \), where \( m \geq 1 \). Then \( B \in C \) if and only if the set of numbers

\[
B' = \{ [x_1, \ldots, x_m] \mid (x_1, \ldots, x_m) \in B \}
\]

belongs to \( C \).

Proof.

If \( P_B(x_1, \ldots, x_m) \) is the characteristic function of \( B \), then

\[
P_{B'}(x) \Leftrightarrow P_B((x)_1, \ldots, (x)_m) \& \text{Lt}(x) = m,
\]

and \( P_{B'} \) clearly belongs to \( C \) if \( P_B \in C \).

On the other hand, \( P_B(x_1, \ldots, x_m) \Leftrightarrow P_{B'}([x_1, \ldots, x_n]) \), hence \( P_{B'} \in C \) implies \( P_B \in C \).
Definition

The set $B \subseteq \mathbb{N}$ is **recursively enumerable** if there is a partially computable function $g(x)$ such that

$$B = \{x \in \mathbb{N} \mid g(x) \downarrow\}.$$

The term recursively enumerable is abbreviated as r.e.

A set is recursively enumerable when it the domain of a partially computable function. Equivalently, $B$ is r.e. if it is just the set of inputs on which some program $P$ halts.
If $P$ is an algorithm for testing the membership in $B$, $P$ will provide an *yes* answer for any $x$ in $B$.

If $x \notin B$ the algorithm $P$ will never terminate. This is why $P$ is also called a *semidecision procedure* for $B$. 
Recursive Enumerable Sets

\[ \begin{cases} 
1 & \text{if } x \in B \\
? & \text{otherwise}
\end{cases} \]

Recursive Sets

Recursive Enumerable Sets

\[ \begin{cases} 
1 & \text{if } x \in B \\
0 & \text{if } x \notin B
\end{cases} \]
Theorem

If $B$ is a recursive set, then $B$ is r.e.

Proof.

Since $B$ is recursive, the predicate $x \in B$ is computable, so we can write the program $P$:

$$[A] \quad \text{IF } (X \in B) \text{ GOTO A}$$

If $h(x)$ is computed by this program then

$B = \{x \in \mathbb{N} \mid h(x) \downarrow\}$. 

□
The set $B$ is recursive if and only if both $B$ and $\overline{B}$ are both r.e.

**Proof.**

If $B$ is recursive, then so is $\overline{B}$, hence both $B$ and $\overline{B}$ are r.e. Conversely, suppose that $B$ and $\overline{B}$ are both r.e., that is

$$B = \{ x \in \mathbb{N} \mid g(x) \downarrow \},$$
$$\overline{B} = \{ x \in \mathbb{N} \mid h(x) \downarrow \},$$

where $g$ and $h$ are both partially computable.
Proof.

Let $g$ be the function computed by program $P$ and $h$ be the function computed by program $Q$, where $\#(P) = p$ and $\#(Q) = q$. The next program computes the characteristic function of $B$:

```plaintext
[A]  IF STP(1)(X, p, T) GOTO C
     IF STP(1)(X, q, T) GOTO E
     T ← T + 1
     GOTO A

[C]  Y ← 1
```
The technique used in the previous proof is known as dovetailing. It combines the algorithms for computing $g$ and $h$ by running the two algorithms for longer and longer times until one of them terminates.
Theorem

If \( B \) and \( C \) are r.e. sets, then so are \( B \cup C \) and \( B \cap C \).

Proof.

Let

\[
B = \{ x \in \mathbb{N} \mid g(x) \downarrow \} \quad \text{and} \quad C = \{ x \in \mathbb{N} \mid h(x) \downarrow \},
\]

where \( g \) and \( h \) are partially computable. Let \( f \) be computed by

\[
Y \leftarrow g(X) \\
Y \leftarrow h(X)
\]

Note that \( f(x) \downarrow \) if and only if \( g(x) \downarrow \) and \( h(x) \downarrow \). Hence

\[
B \cap C = \{ x \in \mathbb{N} \mid f(x) \downarrow \}, \quad \text{so} \quad B \cap C \text{ is r.e.} \]
Proof cont’d

Proof.

For $B \cup C$ we use dovetailing again. Let $g$ be the function computed by program $P$ and $h$ be the function computed by program $Q$, where $(P) = p$ and $(Q) = q$. Let $k(x)$ be computed by

$$[A] \quad \text{IF STP}^{(1)}(X, p, T) \text{ GOTO E}$$
$$\quad \text{IF STP}^{(1)}(X, q, T) \text{ GOTO E}$$
$$\quad T \leftarrow T + 1$$
$$\quad \text{GOTO A}$$

Thus, $k(x) \downarrow$ just when either $g(x) \downarrow$ or $h(x) \downarrow$, that is $B \cup C = \{ x \in \mathbb{N} \mid k(x) \downarrow \}$. 
The definition domain of $\Phi_n(x)$ is the set denoted as $W_n$. Equivalently,

$$W_n = \{ x \in \mathbb{N} \mid \Phi(x, n) \downarrow \}.$$
Theorem

Enumeration Theorem: A set $B$ is r.e. if and only if there is an $n$ for which $B = W_n$.

Proof.

This follows immediately from the definition of $\Phi(x, n)$.

The theorem gets its name from the fact that

$$W_0, W_1, \ldots, W_n, \ldots$$

is an enumeration of all r.e. sets.