1. Rice’s Theorem

2. The Other Rice’s Theorem
The purpose of Rice’s Theorem is to provide a tool that allows us to prove that certain general sets are not recursive.

**Definition**

Let $\Gamma$ be a collection of partially computable functions of one variable. The index set associated with $\Gamma$ is the set

$$R_\Gamma = \{ t \in \mathbb{N} \mid \Phi_t \in \Gamma \}.$$ 

An index set $R_\Gamma$ contains program codes that display a certain input-output behavior as specified by $\Gamma$. 
By Church’s Thesis, $R_{\Gamma}$ is a recursive set when there is an algorithm that accepts program codes as inputs and returns the value TRUE or FALSE depending on whether or not the function $\psi_{P}^{(1)}$ does or does not belong to $\Gamma$. 
Examples of sets of functions $\Gamma$:

- the set of computable functions;
- the set of primitive recursive functions;
- the set of partially computable functions that are defined for all but a finite numbers of values of $x$. 
It would be pleasing to have algorithms that accept a program as input and return as output some useful property of the partial function computed by the program. Unfortunately, such algorithms do not exist.

**Theorem**

**Rice’s Theorem:** Let $\Gamma$ be a collection of partially computable functions of one variable. Let there be partially computable functions $f(x)$ and $g(x)$ such that $f(x)$ belongs to $\Gamma$ and $g(x)$ does not belong to $\Gamma$. Then $R_\Gamma$ is not recursive.
Proof.

Let $h(x)$ be the function such that $h(x) \uparrow$ for all $x$, that is the empty function. There are two cases to discuss:

- $h(x) \not\in \Gamma$, and
- $h(x) \in \Gamma$.

Suppose initially that $h(x) \not\in \Gamma$

Let $q$ be the number of the program

\[
\begin{align*}
Z & \leftarrow \Phi(X_2, X_2) \\
Y & \leftarrow f(X_1)
\end{align*}
\]

Let $\Phi(x_1, x_2, q)$ be the function computed by this program. By the smn Theorem we have $\Phi(x_1, x_2, q) = \Phi(x_1, S^1_1(x_2, q))$. \qed
Proof cont’d

Proof.

Thus, $S^1_1(i, q)$ is the number of the program

\[
\begin{align*}
X_2 &\leftarrow i \\
Z &\leftarrow \Phi(X_2, X_2) \\
Y &\leftarrow f(X_1)
\end{align*}
\]

that computes the function $f$. 

\qed
Proof cont’d

Proof.

Note that

\[ i \in K \implies \Phi(i, i) \downarrow \]

\[ \implies \Phi_{S_1(i,q)}(x) = f(x) \text{ for all } x \]

\[ \implies \Phi_{S_1(i,q)} \in \Gamma \]

\[ \implies S_1(i, q) \in R_\Gamma. \]
Proof cont’d

Also,

Proof.

\[ i \notin K \implies \Phi(i, i) \uparrow \]
\[ \implies \Phi_{S^1_1(i,q)}(x) \uparrow \text{ for all } x \]
\[ \implies \Phi_{S^1_1(i,q)} = h \]
\[ \implies \Phi_{S^1_1(i,q)} \notin \Gamma \]
\[ \implies S^1_1(i, q) \notin R^\Gamma, \]

so \( K \leq_m R^\Gamma \). Therefore, \( R^\Gamma \) is not recursive.
Proof cont’d

Proof.

If \( h(x) \) does not belong to \( \Gamma \), the same argument with \( \Gamma \) and \( f(x) \) replaced by \( \overline{\Gamma} \) and \( g(x) \), respectively, shows that \( R_{\overline{\Gamma}} \) is not recursive.

Since \( R_{\overline{\Gamma}} = \overline{R_{\Gamma}} \), \( R_{\Gamma} \) is not recursive in this case either. \( \square \)
Corollary

There are no algorithms for testing a given program $P$ of the language $S$ to determine whether $\psi_P^{(1)}(x)$ belongs to any of the classes:

- the set of primitive recursive functions;
- the set of partially computable functions that are defined for all but a finite numbers of values of $x$.

Proof.

In each case we need to find the required functions $f$ and $g$ to show that $R_\Gamma$ is not recursive. For example, the functions $f(x) = x$ and $g(x) = 1 - x$ (so that $g$ is defined only for $x = 0$ or $x = 1$) work.
The Other Rice’s Theorem offers a technique for proving that certain sets are nor r.e.

**Definition**

Let \( f, g \) be two partial functions. We write \( f \subseteq g \) if \( x \in \text{Dom}(f) \) implies \( x \in \text{Dom}(g) \) and \( g(x) = f(x) \).
The Other Rice’s Theorem: Let $\Gamma$ be a set of computable functions. If there exist $m, m'$ such that $\Phi_m \in \Gamma$, $\Phi_{m'} \notin \Gamma$ and $\Phi_m \subseteq \Phi_{m'}$, then $R_\Gamma$ is not r.e.
Proof

Consider the flowchart shown on the next slide that can be readily transformed into a $S$ program $\mathcal{P}$, where $\#\mathcal{P} = p$. Note that the execution of this program depends on the input value $x_2$. 
$z \leftarrow 0$

$\text{STP}(x_2, x_2, z) = 1$

$\begin{cases} \text{yes} & y \leftarrow \Phi(x_1, m') \\ \text{no} & \text{STP}(x_1, m, z) = 1 \leftarrow \Phi(x_1, m) \leftarrow \Phi(x_1, m) \end{cases}$

$z \leftarrow z + 1$
The equivalent program $\mathcal{P}$ with $\#(\mathcal{P}) = p$ can be written as:

\[
\begin{align*}
Z &\leftarrow 0 \\
[C] &\text{ IF STP}(X_2, X_2, Z) \text{ GOTO } A \\
&\text{ IF STP}(X_1, m, Z) \text{ GOTO } B \\
&Z \leftarrow Z + 1 \\
&\text{ GOTO } C \\
[A] &Y \leftarrow \Phi(X_1, m) \\
&\text{ GOTO } E \\
[B] &Y \leftarrow \Phi(X_1, m')
\end{align*}
\]
If $x_2 \in K$, we have $\Phi(x_1, x_2, p) = \Phi_{m'}(x_1)$; if $x_2 \notin K$, then $\Phi(x_1, x_2, p) = \Phi_m(x_1)$.

Thus, we have:

$$\Phi(x_1, x_2, p) = \begin{cases} 
\Phi_m(x_1) & \text{if } x_2 \notin K \\
\Phi_{m'}(x_1) & \text{if } x_2 \in K.
\end{cases}$$

By the smn Theorem we have:

$$\Phi_{S_1^{1}(x_2, p)}(x_1) = \begin{cases} 
\Phi_m(x_1) & \text{if } x_2 \notin K \text{ (that is, } x_2 \in \overline{K}) \\
\Phi_{m'}(x_1) & \text{if } x_2 \in K.
\end{cases}$$
Define $f$ as $f(x_2) = S^1_1(x_2, p)$. We have:

- $x \in \overline{K}$ if and only if $\Phi_{f(x_2)} = \Phi_m$, that is, $f(x_2) \in R_{\Gamma}$;
- $x \in K$ if and only if $\Phi_{f(x_2)} = \Phi_{m'}$, that is, $f(x_2) \not\in R_{\Gamma}$.

Thus, $x \in \overline{K}$ if and only if $f(x) \in \Gamma$, so $\overline{K} \leq R_{\Gamma}$, which implies that $R_{\Gamma}$ is not r.e.
**Example**

The following sets can be shown not to be r.e. using the Other Rice’s Theorem:

- \( \text{EMPTY} = \{ x \mid \text{Dom}(\Phi_x) = \emptyset \} \);
- \( \{ x \mid \text{range}(\Phi_x) = \emptyset \} \);
- \( \text{FIN} = \{ x \mid \text{Dom}(\Phi_x) \text{ is finite} \} \);
- \( \text{NOTTOT} = \{ x \mid \Phi_x \text{ is not total} \} \), etc.
For instance, to prove that EMPTY is not r.e. we need to show that there is $m \in \text{EMPTY}$, $m' \notin \text{EMPTY}$ such that $\Phi_m \subseteq \Phi_{m'}$. Choose $\Phi_m$ to be the empty function and $\Phi_{m'}$ to be any function with domain $\{0\}$. Both functions are computable and $\Phi_m \subseteq \Phi_{m'}$. Therefore, EMPTY is not r.e.