1. Rice's Theorem

1.1 Statement

If A is saturated, $A \neq \emptyset$, and $A \neq \mathbb{N}$, then A is not recursive.

1.2 Analysis Template

```
To prove A is not recursive using Rice's theorem:

1. Prove A is saturated:
   - Show A describes a property of computed functions
   - Verify ϕ_x = ϕ_y ⟹ (x ∈ A ⇔ y ∈ A)

2. Prove A ≠ ∅:
   - Find a computable function with property
   - Give index of such function

3. Prove A ≠ ℕ:
   - Find computable function without property
   - Often use undefined function or simple constant

4. Apply Rice's theorem
```

1.3 Common Applications

1. Properties of domains:

```
A = {x | W_x infinite}
A = {x | W_x = ℕ}
A = {x | W_x recursive}
```

2. Properties of functions:

```
A = {x | ϕ_x total}
A = {x | ϕ_x constant}
A = {x | ϕ_x primitive recursive}
```

2. Rice-Shapiro Theorem

2.1 Statement

If $A \subseteq C$ is a set of computable functions and $A = \{x | \phi_x \in A\}$ is r.e., then:

```
∀f (f ∈ A ⇔ ∃θ finite function, θ ⊆ f ∧ θ ∈ A)
```

2.2 Analysis Template

```
To prove A is not r.e. using Rice-Shapiro:

1. Show one of:
   a) ∃f ∉ A with finite θ ⊆ f, θ ∈ A
   b) ∃f ∈ A where no finite θ ⊆ f is in A

2. Verify finite functions involved:
   - Define specific finite functions
   - Show containment in A or not
   - Verify extensions exist

3. Apply Rice-Shapiro theorem
```

2.3 Common Application Patterns

1. Pattern for non-r.e.:

```
- Find total function f ∉ A
- Find finite θ ⊆ f with θ ∈ A
- Conclude A not r.e.
```

2. Pattern for non-co-r.e.:

```
- Find f ∈ A
- Show no finite θ ⊆ f is in A
- Conclude Ā not r.e.
```

3. Combined Applications

3.1 Full Analysis Template

```
For set A:
```

```
1. Check if saturated:
   - Property of computed function?
   - Independent of specific indices?

2. If saturated:
   - Apply Rice's theorem for recursiveness
   - Apply Rice-Shapiro for r.e.

3. If not saturated:
   - Use SRT to construct counterexample
   - Find explicit indices with ϕ_x = ϕ_y but
     different membership in A
```

3.2 Example Problems

1. Total functions:

```
A = {x | ϕ_x total}

- Saturated: yes (property of function)
- Non-empty: yes (constant functions)
- Not ℕ: yes (undefined function)
⟹ Not recursive by Rice

- No finite function in A
⟹ Not r.e. by Rice-Shapiro
```

2. Finite domain:

```
A = {x | W_x finite}

- Saturated: yes (property of function)
- Non-empty: yes (constant functions)
- Not ℕ: yes (identity function)
⟹ Not recursive by Rice

- Every finite function has extension in A
- Identity function not in A but has finite
  subsets in A
⟹ Neither A nor Ā r.e. by Rice-Shapiro
```

4. Common Pitfalls

1. Not verifying saturation:

```
Always check if property depends on:
- Computed function (saturated)
- Specific program details (not saturated)
```

2. Wrong finite functions:

```
Ensure finite functions:
- Are properly defined
- Have correct domain/range
- Actually exist
```

3. Missing cases:

```
Check all conditions:
- Non-empty
- Not all of ℕ
- Finite function existence
```