# Practical S-m-n Theorem Examples: Step-by-Step Real Solutions

## How to Read and Solve ANY S-m-n Exercise

### STEP 1: Extract the Requirements from Exercise Text

Look for these patterns in the exercise:

- **"show there exists s:** ℕ → ℕ **such that..."** → This tells you what function to construct
- **"W_s(x) = ..."** → This is the DOMAIN (where function is defined)
- **"E_s(x) = ..."** → This is the CODOMAIN (what values function outputs)
- **"|W_s(x)| = ..."** → This is SIZE of domain
- **"|E_s(x)| = ..."** → This is SIZE of codomain

### STEP 2: Understand W_x and E_x Logically

- **W_x = domain of φ_x** = {inputs where φ_x converges}
- **E_x = codomain of φ_x** = {outputs that φ_x produces}
- **Key insight:** g(x,y) ↓ ⟺ y ∈ W_s(x), and g(x,y) = z ⟺ z ∈ E_s(x)

---

## OFFICIAL EXERCISE 3.4: Even Domain, Offset Codomain

**Exact Exercise Text:** "Prove that there is a total computable function k : ℕ → ℕ such that for each n ∈ ℕ it holds that W_k(n) = P = {x ∈ ℕ | x even} and E_k(n) = {x ∈ ℕ | x ≥ n}."

### Complete Step-by-Step Solution:

#### 1. Extract Requirements from Text:

- Look for "W_k(n) = ..." → W_k(n) = P = {0, 2, 4, 6, ...} (all even numbers)
- Look for "E_k(n) = ..." → E_k(n) = {n, n+1, n+2, ...} (all numbers ≥ n)

#### 2. Logical Analysis:

- **Domain pattern:** Function should be defined ONLY on even inputs (0, 2, 4, 6, ...)
- **Codomain pattern:** Function outputs should be ≥ n
- **Key insight:** When input x is even, we want output ≥ n

#### 3. Construct f(n,x):

```
f(n, x) = {
    x/2 + n,  if x is even
    ↑,        if x is odd
}
```

**4. Verify Why This Works:**

- **Domain check:** $f(n,x)$ converges $\iff$ x is even $\iff$ $x \in P$
  - So $W\_k(n) = \{x \mid f(n,x) \downarrow\} = \{x \mid x \text{ even}\} = P$ ✓
- **Codomain check:** When x is even: $f(n,x) = x/2 + n$
  - Since $x \in \{0, 2, 4, 6, \ldots\}$, we have $x/2 \in \{0, 1, 2, 3, \ldots\}$
  - So $f(n,x) \in \{n, n+1, n+2, n+3, \ldots\} = \{y \mid y \geq n\}$
  - Therefore $E\_k(n) = \{n, n+1, n+2, \ldots\}$ ✓

**5. Make It Computable:**

```
f(n, x) = qt(2, x) + n + μz.rm(2, x)
```

- qt(2, x) = x/2 (quotient)
- rm(2, x) = x mod 2 (remainder: 0 if even, 1 if odd)
- μz.rm(2, x) converges to 0 if x even, diverges if x odd

---

## OFFICIAL EXERCISE 3.5: Threshold Domain, Even Codomain

**Exact Exercise Text:** "Use it to prove it exists a total computable function $k : \mathbb{N} \to \mathbb{N}$ such that $W\_k(n) = \{x \in \mathbb{N} \mid x \geq n\}$ and $E\_k(n) = \{y \in \mathbb{N} \mid y \text{ even}\}$ for all $n \in \mathbb{N}$."

### Complete Step-by-Step Solution:

**1. Extract Requirements from Text:**

- $W\_k(n) = \{n, n+1, n+2, \ldots\}$ (all numbers $\geq$ n)
- $E\_k(n) = \{0, 2, 4, 6, \ldots\}$ (all even numbers)

**2. Logical Analysis:**

- **Domain pattern:** Function defined when $x \geq n$
- **Codomain pattern:** Function outputs should be even numbers
- **Key insight:** For inputs $x \geq n$, we want to produce even outputs

**3. Construct f(n,x):**

```
f(n, x) = {
    2 * (x - n),  if x ≥ n
    ↑,            if x < n
}
```

**4. Verify Why This Works:**

- **Domain check:** f(n,x) converges ⟺ x ≥ n
  - So W_k(n) = {x | f(n,x) ↓} = {x | x ≥ n} ✓
- **Codomain check:** When x ≥ n: f(n,x) = 2*(x-n)
  - Since x ≥ n, we have (x-n) ∈ {0, 1, 2, 3, ...}
  - So 2*(x-n) ∈ {0, 2, 4, 6, ...} = even numbers
  - Therefore E_k(n) = {y | y even} ✓

**5. Make It Computable:**

```
f(n, x) = 2 * (x - n) + μz.(n - x)
```

- (x - n) gives the offset from n
- μz.(n - x) converges to 0 if x ≥ n, diverges if x < n

---

## OFFICIAL EXERCISE 3.3: Equation Domain

**Exact Exercise Text:** "State the smn theorem and use it to prove that there exists a total computable function s : $\mathbb{N}^2 \rightarrow \mathbb{N}$ such that W_s(x,y) = {z : x * z = y}"

### Complete Step-by-Step Solution:

**1. Extract Requirements from Text:**

- Function s takes TWO parameters: x and y
- W_s(x,y) = {z | x * z = y} (solutions to equation x * z = y)
- No requirement for E_s(x,y) specified

**2. Logical Analysis:**

- **When does equation x * z = y have solutions?**
  - If x = 0 and y = 0: z can be anything → W_s(0,0) = ℕ
  - If x = 0 and y ≠ 0: no solutions → W_s(0,y) = ∅
  - If x ≠ 0 and y divisible by x: z = y/x → W_s(x,y) = {y/x}
  - If x ≠ 0 and y not divisible by x: no solutions → W_s(x,y) = ∅

**3. Construct g(x,y,z):**

```
g(x, y, z) = {
    0,  if x * z = y
    ↑,  otherwise
}
```

**4. Verify Why This Works:**

- g(x,y,z) converges ⟺ x * z = y ⟺ z is a solution to x * z = y
- So W_s(x,y) = {z | g(x,y,z) ↓} = {z | x * z = y} ✓

**5. Make It Computable:**

```
g(x, y, z) = μw.|(x * z) - y| + |y - (x * z)|
```

This converges to 0 iff x*z = y, diverges otherwise.

---

## OFFICIAL EXERCISE: Power Domain with Odd Codomain

**Exact Exercise Text:** "Use it for proving there exists k: ℕ → ℕ total and computable s.t. ∀n ∈ ℕ we have W_k(n) = {z^n | z ∈ ℕ} and E_k(n) is the set of odd numbers"

### Complete Step-by-Step Solution:

**1. Extract Requirements:**

- W_k(n) = {0^n, 1^n, 2^n, 3^n, ...} = {z^n | z ∈ ℕ}
- E_k(n) = {1, 3, 5, 7, ...} (all odd numbers)

**2. Logical Analysis:**

- **Domain:** Function defined when input y is a perfect n-th power
- **Codomain:** Function outputs should be odd numbers

**3. Construct g(n,y):**

```
g(n, y) = {
    2k + 1,  if ∃z such that y = z^n (and k is some index)
    ↑,       otherwise
}
```

**4. Better Construction:**

```
g(n, y) = {
    2 * (μz.|z^n - y|) + 1,  if ∃z such that y = z^n
    ↑,                       otherwise
}
```

**5. Make It Computable:**

```
g(n, y) = 2 * (μz.|z^n - y|) + 1
```

This finds z such that z^n = y, then outputs 2z + 1 (which is odd).

---

## PRACTICAL EXAMPLE 1: Basic Cardinality Control (From Tutorial) (From Tutorial) (From Tutorial)

**Exercise Text:** "Show there exists s: ℕ → ℕ such that |W_s(x)| = 2x and |E_s(x)| = x"

### Step-by-Step Solution:

#### 1. Extract Requirements:

- Need: function s such that |W_s(x)| = 2x and |E_s(x)| = x
- This means: W_s(x) has exactly 2x elements, E_s(x) has exactly x elements

#### 2. Logical Analysis:

- W_s(x) = {0, 1, 2, ..., 2x-1} (first 2x natural numbers)
- E_s(x) = {0, 1, 2, ..., x-1} (first x natural numbers)

#### 3. Construct g(x,y):

```
g(x, y) = {
    qt(2, y),  if y < 2x
    ↑,         otherwise
}
```

#### 4. Why this works:

- g(x,y) converges ⟺ y < 2x ⟺ y ∈ {0,1,2,...,2x-1}
- So W_s(x) = {0,1,2,...,2x-1} → |W_s(x)| = 2x ✓
- When y < 2x: g(x,y) = qt(2,y) = ⌊y/2⌋ ∈ {0,1,2,...,x-1}
- So E_s(x) = {0,1,2,...,x-1} → |E_s(x)| = x ✓

#### 5. Make it computable:

```
g(x, y) = qt(2, y) + μz.(y + 1 - 2x)
```

---

## PRACTICAL EXAMPLE 2: Set Construction with Conditions

**Exercise Text:** "Show there exists k: ℕ → ℕ such that W_k(n) = P (even numbers) and E_k(n) = {y | y ≥ n}"

### Step-by-Step Solution:

#### 1. Extract Requirements:

- W_k(n) = P = {0, 2, 4, 6, ...} (all even numbers)
- E_k(n) = {n, n+1, n+2, ...} (all numbers ≥ n)

#### 2. Logical Analysis:

- Function should be defined ONLY on even inputs
- For even input y, output should be something ≥ n

**3. Construct g(n,x):**

```
g(n, x) = {
    x/2 + n,  if x is even
    ↑,        if x is odd
}
```

**4. Why this works:**

- g(n,x) converges ⟺ x is even ⟺ x ∈ P
- So W_k(n) = P ✓
- When x is even: g(n,x) = x/2 + n ≥ n
- As x ranges over all evens {0,2,4,...}, x/2 ranges over {0,1,2,...}
- So g(n,x) ranges over {n, n+1, n+2, ...}
- Therefore E_k(n) = {y | y ≥ n} ✓

**5. Make it computable:**

```
g(n, x) = qt(2, x) + n + μz.rm(2, x)
```

---

## PRACTICAL EXAMPLE 3: Divisor Construction

**Exercise Text:** "Show there exists k: ℕ → ℕ such that φ_k(n) is total and E_k(n) is the set of integer divisors of n"

### Step-by-Step Solution:

**1. Extract Requirements:**

- φ_k(n) must be total (defined everywhere)
- E_k(n) = divisors of n = {d | d divides n}

**2. Logical Analysis:**

- Since φ_k(n) is total: W_k(n) = ℕ
- Need E_k(n) to contain exactly the divisors of n

**3. Construct g(n,x):**

```
g(n, x) = {
    x * n,  if x divides n
    1,      otherwise
}
```

**4. Why this works:**

- g(n,x) is always defined → W_k(n) = ℕ → φ_k(n) total ✓
- When x divides n: g(n,x) = x*n, but we want just x
- When x doesn't divide n: g(n,x) = 1

**Actually, better construction:**

```
g(n, x) = {
    x,  if x divides n
    1,  otherwise
}
```

- E_k(n) = {x | x divides n} ∪ {1} = divisors of n ✓

**5. Make it computable:**

```
g(n, x) = x * sg(rm(x, n)) + sg(rm(x, n))
```

where sg(0) = 1, sg(y) = 0 for y > 0.

---

## PRACTICAL EXAMPLE 4: Equation Solving

**Exercise Text:** "Show there exists s: $\mathbb{N}^2 \to \mathbb{N}$ such that W_s(x,y) = {z | x * z = y}"

## Step-by-Step Solution:

**1. Extract Requirements:**

- W_s(x,y) = {z | x * z = y} (solutions to equation x * z = y)

**2. Logical Analysis:**

- Function should be defined exactly when x * z = y
- No specific requirement for codomain, so we can output anything

**3. Construct g(x,y,z):**

```
g(x, y, z) = {
    0,  if x * z = y
    ↑,  otherwise
}
```

**4. Why this works:**

- g(x,y,z) converges ⟺ x * z = y ⟺ z ∈ {solutions to x*z = y}
- So W_s(x,y) = {z | x * z = y} ✓

**5. Make it computable:**

```
g(x, y, z) = μw.|(x * z) - y| + |y - (x * z)|
```

This converges to 0 iff x*z = y.

---

## PRACTICAL EXAMPLE 5: Square Set Construction

**Exercise Text:** "Show there exists s: ℕ → ℕ such that W_s(x) = $\{(k + 2)^2 \mid k \in \mathbb{N}\}$"

### Step-by-Step Solution:

**1. Extract Requirements:**

- W_s(x) = {4, 9, 16, 25, ...} = $\{(k+2)^2 \mid k = 0,1,2,...\}$
- Note: This set doesn't depend on x!

**2. Logical Analysis:**

- For any x, we want the same domain: perfect squares starting from 4
- We need to check if y is a perfect square of form $(k+2)^2$

**3. Construct g(x,y):**

```
g(x, y) = {
    k,  if ∃k such that y = (x + k)²
    ↑,  otherwise
}
```

Wait, the exercise says W_s(x) should be the same for all x. Let me re-read...

Actually, looking at the pattern, it should be:

```
g(x, y) = {
    k,  if ∃k such that y = (k + 2)²
    ↑,  otherwise
}
```

**4. Make it computable:**

```
g(x, y) = μk.|y - (k + 2)²|
```

This finds k such that $y = (k+2)^2$, or diverges if no such k exists.

---

## PRACTICAL EXAMPLE 6: Cardinality with Two Parameters

**Exercise Text:** "Show there exists s: $\mathbb{N}^2 \to \mathbb{N}$ such that |W_s(x,y)| = x * y"

### Step-by-Step Solution:

**1. Extract Requirements:**

- |W_s(x,y)| = x * y (domain has exactly x*y elements)

**2. Logical Analysis:**

- W_s(x,y) = {0, 1, 2, ..., x*y - 1} (first x*y natural numbers)

**3. Construct g(x,y,z):**

```
g(x, y, z) = {
    0,  if z < x * y
    ↑,  otherwise
}
```

**4. Why this works:**

- g(x,y,z) converges ⟺ z < x*y ⟺ z ∈ {0,1,2,...,x*y-1}
- So |W_s(x,y)| = x*y ✓

**5. Make it computable:**

```
g(x, y, z) = μw.(z + 1 - (x * y))
```

---

## PATTERN RECOGNITION FROM EXERCISE TEXT

### How to Read Exercise Requirements Like a Detective

**Pattern 1: Cardinality Control**

- **Text signals:** "|W_s(x)| = ...", "exactly n elements", "size of domain"
- **What to do:** Make g(x,y) converge for exactly that many y values
- **Example:** "|W_s(x)| = 2x" → g(x,y) defined for y ∈ {0,1,2,...,2x-1}

**Pattern 2: Specific Set Construction**

- **Text signals:** "W_s(x) = {specific description}", "even numbers", "divisors"
- **What to do:** Make g(x,y) converge exactly when y is in that set
- **Example:** "W_s(x) = even numbers" → g(x,y) defined when rm(2,y) = 0

**Pattern 3: Equation Solving**

- **Text signals:** "W_s(x,y) = {z | equation}", "solutions to", "such that"
- **What to do:** Make g(x,y,z) converge exactly when equation holds
- **Example:** "W_s(x,y) = {z | xz = y}" → g(x,y,z) defined when xz = y

**Pattern 4: Total Functions**

- **Text signals:** "φ_k(n) is total", "defined everywhere", "total computable"
- **What to do:** Make g(n,x) always converge (never use ↑)
- **Example:** "φ_k(n) total" → g(n,x) = some_always_defined_expression

**Pattern 5: Threshold Conditions**

- **Text signals:** "x ≥ n", "greater than", "at least"
- **What to do:** Use μz.(threshold - x) pattern
- **Example:** "x ≥ n" → add μz.(n - x) to make it diverge when x < n

## COMMON MISTAKES AND HOW TO AVOID THEM

### Mistake 1: Confusing Domain and Codomain

**Wrong thinking:** "W_s(x) is what the function outputs" **Correct thinking:** "W_s(x) is where function is DEFINED, E_s(x) is what it OUTPUTS" **Fix:** Always ask "When does g(x,y) converge?" for W_s(x), "What does g(x,y) equal?" for E_s(x)

### Mistake 2: Not Making g(x,y) Computable

**Wrong:** g(x,y) = "check if y is prime" (not primitive recursive) **Correct:** Use only qt, rm, sg, μ-operator, +, *, and compositions **Fix:** Break down complex conditions into primitive recursive parts

### Mistake 3: Wrong μ-operator Usage

**Wrong:** μz.(condition) where condition can be false **Correct:** μz.|expression1 - expression2| where difference is 0 iff condition holds **Example:** μz.|xz - y| finds z such that xz = y

### Mistake 4: Ignoring Edge Cases

**Wrong:** Assuming $x \neq 0$ in equation $x*z = y$ **Correct:** Handle $x = 0$ case separately or use robust construction **Fix:** Test your construction with small values: x=0, x=1, y=0, etc.

### Mistake 5: Circular Reasoning in Verification

**Wrong:** "W_s(x) contains even numbers because g(x,y) works for even y" **Correct:** "g(x,y) converges exactly when y is even, therefore W_s(x) = even numbers" **Fix:** Always verify: W_s(x) = {y | g(x,y) ↓} matches requirement

## MECHANICAL READING ALGORITHM


## MECHANICAL READING ALGORITHM


### For ANY S-m-n Exercise, Follow These Steps:

**Step 1: SCAN the exercise text**

- Find: "show there exists..." → This tells you what to construct
- Count parameters: s: $\mathbb{N} \to \mathbb{N}$ (one param) vs s: $\mathbb{N}^2 \to \mathbb{N}$ (two params)
- Extract ALL requirements about W_s and E_s

**Step 2: CLASSIFY the exercise type**

- Cardinality? Look for |W_s(x)| = ...
- Specific sets? Look for W_s(x) = {description}
- Equations? Look for W_s(x,y) = {z | equation}
- Total functions? Look for "φ_k(n) is total"

**Step 3: TRANSLATE requirements to g(x,y) logic**

- W_s(x) requirement → "When should g(x,y) converge?"
- E_s(x) requirement → "What should g(x,y) equal when it converges?"

**Step 4: CONSTRUCT g(x,y) using templates**

- Use pattern templates from examples above
- Ensure it uses only primitive recursive functions

**Step 5: VERIFY your construction**

- Check: W_s(x) = {y | g(x,y) ↓} matches requirement
- Check: E_s(x) = {g(x,y) | y ∈ W_s(x)} matches requirement
- Test with small values: x=0, x=1, x=2

**Step 6: WRITE the formal solution**

- State S-m-n theorem
- Define g(x,y) explicitly
- Apply S-m-n theorem to get s
- Verify both W_s and E_s properties

## UNIVERSAL PATTERN RECOGNITION

### Quick Reference Table:

| Exercise Text Contains | Pattern Type | g(x,y) Template |
|---|---|---|
| "\|W_s(x)\| = f(x)" | Cardinality | g(x,y) defined for y < f(x) |
| "W_s(x) = even numbers" | Even/Odd | g(x,y) + μz.rm(2,y) |
| "W_s(x) = {y \| y ≥ x}" | Threshold | g(x,y) + μz.(x - y) |
| "W_s(x,y) = {z \| equation}" | Equation | g(x,y,z) defined when equation holds |
| "φ_k(n) is total" | Total Function | g(n,x) always defined |
| "divisors of n" | Divisibility | Use rm(x,n) = 0 test |
| "E_s(x) = odd numbers" | Odd Codomain | g(x,y) = 2*something + 1 |

### Meta-Pattern: The Universal Logic

**Every S-m-n exercise follows this logic:**

1. You need to create a function φ_s(x) with specific domain/codomain properties
2. φ_s(x)(y) = g(x,y) by S-m-n theorem
3. Design g(x,y) so convergence creates desired domain, outputs create desired codomain
4. W_s(x) = {y | g(x,y) ↓}, E_s(x) = {g(x,y) | y ∈ W_s(x)}

**This is why the methodology works for EVERY exercise:**

- Extract what W_s(x) and E_s(x) should be
- Design g(x,y) to create those convergence and output patterns
- Verify the construction matches requirements
- Apply S-m-n theorem mechanically

You now have a complete system to solve any S-m-n theorem exercise by pattern recognition and mechanical application!