Practical Guide: Non-Computable Exercises - Diagonalization & χ_k Methods

The Two Universal Approaches

EXAM STRATEGY: Every non-computable function exercise uses exactly one of these methods:

1. **Diagonalization Argument**
2. **Characteristic Function of K (χ_k)**

Method 1: Diagonalization Argument

Core Principle

Build a function f such that **$f(x) \neq \varphi_x(x)$ for all x** $\in \mathbb{N}$

Standard Template

```javascript
f(x) = {
    something_involving_ϕₓ(x) + 1    if ϕₓ(x) ↓
    0 (or any constant)              if ϕₓ(x) ↑
}
```

How Diagonalization Works: Step-by-Step

The Core Idea: Build f so that $f(x) \neq \varphi_x(x)$ for ALL $x \in \mathbb{N}$

Visualization:

```
    x=0  x=1  x=2  x=3  ...
ϕ₀   ?    ?    ?    ?   ...
ϕ₁   ?    ?    ?    ?   ...
ϕ₂   ?    ?    ?    ?   ...
ϕ₃   ?    ?    ?    ?   ...
...  ...  ...  ...  ... ...
f   ≠ϕ₀(0) ≠ϕ₁(1) ≠ϕ₂(2) ≠ϕ₃(3) ...
```

Why It Works:

- **Total by construction**: Always defined (either $\varphi_x(x) + 1$ or constant)
- **Non-computable by design**: $f(x) \neq \varphi_x(x)$ always
  - Case 1: $\varphi_x(x) \downarrow \rightarrow f(x) = \varphi_x(x) + 1 \neq \varphi_x(x)$ ✓
  - Case 2: $\varphi_x(x) \uparrow \rightarrow f(x) = 0 \neq \uparrow$ ✓
- **Cannot be computable**: If $f = \varphi_e$ for some e, then $f(e) \neq \varphi_e(e) = f(e) \rightarrow$ contradiction!

Exam Examples from Your Materials

Example 1: Basic Diagonalization (Classic Pattern)

```javascript
f(x) = {
    ϕₓ(x) + 1    if ϕₓ(x) ↓
    0            if ϕₓ(x) ↑
}
```

Verification: $f(x) \neq \varphi_x(x)$ always ✓

Example 2: With Constraints (codomain ⊆ {0,1})

```javascript
f(x) = {
    sg(ϕₓ(x))    if ϕₓ(x) ↓  // sg = sign function
    0            if ϕₓ(x) ↑
}
```

Key: When $\varphi_x(x) \downarrow$, $sg(\varphi_x(x)) = 0$ if $\varphi_x(x) = 0$, $= 1$ if $\varphi_x(x) > 0$

Example 3: Image = $\{2^n \mid n \in \mathbb{N}\}$ (Exam Pattern)

```javascript
f(x) = {
    2             if ϕₓ(x) ↑
    2             if ϕₓ(x) ↓ ≠ 2
    4             if ϕₓ(x) ↓ = 2
}
```

Why works: $2^n \neq \varphi_n(x)$, $\varphi_x(x) = n \rightarrow 2^n = n$ impossible for most cases

Example 4: f(x) = x for infinitely many x

```javascript
f(x) = {
    ϕₓ(x) + 1    if x ∈ Wₓ
    x            if x ∉ Wₓ
}
```

Key: Empty function ∅ has infinitely many indices → $x \notin W_x$ infinitely often

Example 5: Image = ℕ{0} (From Feb 2022 Exam)

```javascript
f(x) = {
    ϕₓ(x) + 1    if x ∈ Wₓ
    1            if x ∉ Wₓ
}
```

Verification:

- If $\varphi_x(x)\downarrow$ then $f(x) = \varphi_x(x) + 1 \geq 1$
- If $\varphi_x(x)\uparrow$ then $f(x) = 1$
- For any $n \geq 1$, take index e of constant $n-1 \to f(e) = n$

Advanced Diagonalization: Multiple Constraints

Example: f(x) = f(x+1) infinitely often

```javascript
f(x) = {
    0                        if x ≢ 0 (mod 3)
    0                        if x = 3y ∧ ϕᵧ(x) ↑
    ϕᵧ(x) + 1                if x = 3y ∧ ϕᵧ(x) ↓
}
```

- **Why it works**: For $x = 3n+1$, both x and x+1 are not multiples of 3, so $f(x) = f(x+1) = 0$
- **Diagonalization**: $f(3y) \neq \varphi_y(3y)$ always

Method 2: Characteristic Function of K (χ_k)

When to Use χ_k

- Exercise asks for specific properties of total non-computable functions
- Need function with codomain {0,1}
- Question involves infinite sets or complementary behaviors

Key Properties of χ_k

```javascript
χₖ(x) = {
    1    if x ∈ K (i.e., ϕₓ(x) ↓)
    0    if x ∉ K (i.e., ϕₓ(x) ↑)
}
```

Why χ_k is Non-Computable

Proof by Contradiction: If χ_k were computable, then K would be recursive, contradicting known results.

χ_k Applications from Real Exams

Example 1: f(x) = f(x+1) infinitely often (Jan 2022 Exam)

- **Direct Answer**: $f = \chi_k$
- **Elegant Proof**: If $\{x \mid \chi_k(x) = \chi_k(x+1)\}$ were finite with max d, then for $x > d$: $\chi_k(x+1) = sg(\chi_k(x)) \rightarrow \chi_k$ becomes computable via primitive recursion → contradiction!

Example 2: Express via χ_k (Standard Pattern)

```javascript
// Given: f(x) = {2x+1 if φₓ(x) ↓; 2x-1 if φₓ(x) ↑}
// Prove non-computable by:
χₖ(x) = sg(|f(x) - 2x|)
// If f computable → χₖ computable → contradiction
```

Example 3: Almost Total Functions

```javascript
f(x) = {
    0           if x ∈ Wₓ
    ϕₓ(x) + 1   if x ∉ Wₓ
}
```

Proof: If f computable & almost total (undefined on finite D), then: $\chi_k(x) = \{f(x) \text{ if } x \in D; \text{finite\_function}(x) \text{ otherwise}\}$ computable → contradiction

Decision Framework for Exam

Step 1: Identify Exercise Type

```
Does the exercise ask for:
├─ Function with specific arithmetic property? → Try Diagonalization
├─ Function with domain/codomain constraints? → Try Diagonalization
├─ Function with {0,1} codomain or infinite behavior? → Try χₖ
└─ Proving another function non-computable? → Express via χₖ
```

Step 2: Apply Template

For Diagonalization:

1. Define $f(x) = \{\text{expression\_with\_}\varphi_x(x) \text{ if } \varphi_x(x) \downarrow; \text{ constant if } \varphi_x(x) \uparrow\}$
2. Verify totality (both cases covered)
3. Verify $f(x) \neq \varphi_x(x)$ always
4. Check additional constraints (codomain, etc.)

For χ_k Method:

1. Use f = χ_k directly OR
2. Show f can compute χ_k via f
3. Conclude by contradiction (χ_k not computable)

Real Exam Patterns and Instant Solutions

Pattern A: "Total non-computable with specific image"

Template: Use diagonalization with mapping

```javascript
f(x) = map_to_target({
    ϕₓ(x) + offset    if ϕₓ(x) ↓
    default_value     if ϕₓ(x) ↑
})
```

Examples:

- Image = ℕ{0} → f(x) = {$\varphi_x(x)+1$ if $x \in W_x$; 1 otherwise}
- Image = {2^n} → f(x) = {2 if $\varphi_x(x)\uparrow$; $2^{(\varphi_x(x)+1)}$ if $\varphi_x(x)\downarrow \neq 2$; 4 if $\varphi_x(x)=2$}
- Image = primes → f(x) = {next_prime($\varphi_x(x)$) if $\varphi_x(x)\downarrow$; 2 otherwise}

Pattern B: "f(x) = g(x) infinitely often"

Solution: χ_k (most elegant) OR diagonalization with modular arithmetic

Pattern C: "Non-computable with arithmetic property"

Template: Diagonalization ensuring property

```javascript
// Example: f(x) returns 0 when x even
f(x) = {
    0                       if x even
    ϕ_{(x-1)/2}(x) + 1    if x odd ∧ ϕ_{(x-1)/2}(x) ↓
    0                       if x odd ∧ ϕ_{(x-1)/2}(x) ↑
}
```

Pattern D: "Prove given function non-computable"

Template: Show it computes χ_k

```javascript
χₖ(x) = transformation_of_f(x)
```

Exam Strategy Flowchart

```
Read problem → What type?
├─ Build total non-computable f? → Diagonalization
│   ├─ With constraints? → Map ϕₓ(x) appropriately
│   └─ Standard case? → f(x) = {ϕₓ(x)+1 if ↓; 0 if ↑}
├─ Prove f non-computable? → Express χₖ via f
├─ f(x) = f(x+1) infinitely? → Use χₖ
└─ Special property + non-computable? → Modified diagonalization
```

Verification Checklist

For Diagonalization:

- [ ] Function is total (both $\varphi_x(x) \downarrow$ and $\varphi_x(x) \uparrow$ cases covered)
- [ ] $f(x) \neq \varphi_x(x)$ for all x (crucial!)
- [ ] Additional constraints satisfied (codomain, etc.)
- [ ] Clear explanation why f cannot be computable

For χ_k Method:

- [ ] Correctly identified that χ_k applies
- [ ] Clear explanation of χ_k properties used
- [ ] If indirect proof, correct expression relating f and χ_k
- [ ] Proper contradiction argument

Critical Exam Mistakes to Avoid

❌ Fatal Errors

1. **Forgetting $\varphi_x(x) \uparrow$ case** → Function not total → 0 points
2. **Not verifying $f(x) \neq \varphi_x(x)$** for both cases → No diagonalization → 0 points
3. **Wrong direction in χ_k reduction** → Logical error → Major point loss
4. **Incomplete totality proof** → Function might be partial → Point deduction

❌ Common Confusion

- Using diagonalization when χ_k is simpler (e.g., infinite behavior questions)
- Overcomplicating simple cases (basic $f(x) = \{\varphi_x(x)+1 \text{ if } \downarrow; 0 \text{ if } \uparrow\}$ works!)
- Missing image/codomain verification in constraint problems

10-Minute Exam Strategy

Minutes 0-2: Pattern Recognition

```
Quick scan → Which pattern?
- "Total non-computable" + constraints → Diagonalization
- "f(x) = f(x+k) infinitely" → χₖ
- "Prove f non-computable" → χₖ reduction
- "Specific arithmetic property" → Modified diagonalization
```

Minutes 2-8: Execution

For Diagonalization:

1. Write template: $f(x) = \{\varphi_x(x) + \text{something if } \downarrow; \text{constant if } \uparrow\}$
2. Adjust for constraints (image, codomain, etc.)
3. Verify $f(x) \neq \varphi_x(x)$ always
4. Check totality

For χ_k Method:

1. State $\chi_k(x) = \{1 \text{ if } \varphi_x(x)\downarrow; 0 \text{ if } \varphi_x(x)\uparrow\}$
2. Apply to problem (direct use OR reduction)
3. Conclude by contradiction

Minutes 8-10: Verification & Writing

- Double-check $f(x) \neq \varphi_x(x)$ for BOTH cases
- Verify all constraints satisfied
- Write clear "therefore f is non-computable" conclusion

Emergency Cheat Sheet

Default Diagonalization: $f(x) = \{\varphi_x(x) + 1 \text{ if } \varphi_x(x)\downarrow; 0 \text{ if } \varphi_x(x)\uparrow\}$

Default χ_k Reduction: $\chi_k(x) = sg(|f(x) - \text{expression}|)$

Verification Mantra: "$f(x) \neq \varphi_x(x)$ for all $x \in \mathbb{N}$"

If stuck: Start with basic diagonalization, then adapt for constraints!