# Universal Function & Basic Definitions

## Universal Function ($\Psi^k_u$)

```
Ψᵏᵤ: Nᵏ⁺¹ → N
Ψᵤ(e,x̄) = ϕₑᵏ(x̄)
```

This is computable and represents a universal interpreter that can simulate any k-arity program.

# Important Predicates & Functions

## H Predicate (Halting)

```
H(e,x,t) = "Program Pe(x) halts in t or fewer steps"
```

- Decidable
- Used to check program termination within bounded steps
- Characteristic function: $\chi H(e,x,t)$

## S Predicate (Step)

```
S(e,x,y,t) = "Program Pe(x) halts with output y in t or fewer steps"
```

- Decidable
- Used to check specific program output within bounded steps
- Characteristic function: $\chi S(e,x,y,t)$

## Key Properties

- Both H and S are decidable because they only consider bounded computation
- Used extensively in proofs and reductions
- Form basis for many computability results

# Effective Operations on Functions

## Inverse Function

For a computable injective function f:

```
f⁻¹(y) = {
    x  if ∃x. f(x) = y
    ↑  otherwise
}
```

### Proof of Computability

```
f⁻¹(y) = (μw.|χS(e,(w)₁,y,(w)₂) - 1|)₁
```

where e is an index for f

## Union of Domains

There exists total computable s such that:

```
Ws(x,y) = Wx ∪ Wy
```

## Union of Ranges

There exists total computable s such that:

```
Es(x,y) = Ex ∪ Ey
```

# Program Properties

## Computing Indices

- Every computable function has infinitely many indices
- Can effectively transform programs (indices) while preserving function behavior

## Program Compositions

Can effectively compute indices for:

- Function composition
- Function products
- Function sums
- Inverse functions (when they exist)

# Important Sets & Functions

## K (Halting Set)

```
K = {x | x ∈ Wx} = {x | ϕx(x)↓}
```

- R.e. but not recursive
- Often used in reductions

## Tot (Totality Set)

```
Tot = {x | ϕx is total}
```

- Not r.e.
- Not co-r.e.

# Working with Indices

## Example: Computing $f^{-1}$

1. Get index e for f
2. Search systematically for x where f(x) = y
3. Use S predicate to verify computation
4. Return first matching x

## Common Pattern for Proofs

1. Define a computable function g
2. Use s-m-n theorem to get index transformation
3. Show resulting function has desired property