CSC236H

Introduction to the Theory of Computation

```
# Pre-condition: n is a natural number greater than 0.
def f(n):
    if n==1 or n==2:
        return 1
    return f(n-1)+f(n-2)
```

$$f(n) = \begin{cases} 1 & n = 1 \text{ or } n = 2 \\ f(n-1) + f(n-2) & n \geq 3 \end{cases}$$

Recursive Definition of a Set

1. Indicate the smallest or simplest objects. $\rightarrow$ Base Rule or the Basis
2. Indicate how larger or more complex objects can be built out of the smaller or simpler ones. $\rightarrow$ Recursive Rule or Inductive Rule

Example: A recursive definition for $\mathbb{N}$:

- $0 \in \mathbb{N}$;
- if $k \in \mathbb{N}$ then $k + 1 \in \mathbb{N}$;
- nothing else belongs to $\mathbb{N}$.

Recursive Definition of a Set

1. Indicate the smallest or simplest objects.
2. Indicate how larger or more complex objects can be built out of the smaller or simpler ones.

Example: A recursive definition for **non-empty binary trees**:

- a single node is a non-empty binary tree;
- if T_1, T_2 are two disjoint non-empty binary trees, then the tree with a new root r connected to the roots of T_1 and T_2 is a non-empty binary tree;
- if T_1 is a non-empty binary tree, then the tree with a new root r connected to the root of T_1 is a non-empty binary tree;
- nothing else is a binary tree.

Recursive Definition of a Set:

1. Indicate the smallest or simplest objects.
2. Indicate how larger or more complex objects can be built out of the smaller or simpler ones.

Structural Induction: Prove $\mathbf{P}$ holds for all elements of a recursively defined set $\mathbf{X}$:

1. **Base Case:** Prove that every smallest or simplest element of $\mathbf{X}$ satisfies $\mathbf{P}$.
2. **Induction Step:** Assume that $\mathbf{P}$ holds for smaller or simpler elements. **[IH]**

   Prove that every element that is constructed following each of the (finitely many) ways of constructing larger or more complex elements out of the smaller or simpler ones satisfies $\mathbf{P}$.

Structural Induction: Prove $\mathbf{P}$ holds for all elements of a recursively defined set $\mathbf{X}$:

1. **Base Case:** Prove that every smallest or simplest element of $\mathbf{X}$ satisfies $\mathbf{P}$.
2. **Induction Step:** Assume that $\mathbf{P}$ holds for smaller or simpler elements. **[IH]**

   Prove that every element that is constructed following each of the (finitely many) ways of constructing larger or more complex elements out of the smaller or simpler ones satisfies $\mathbf{P}$.

Example: A recursive definition for $\mathbb{N}$:

- $0 \in \mathbb{N}$;
- if $k \in \mathbb{N}$ then $k + 1 \in \mathbb{N}$;
- nothing else belongs to $\mathbb{N}$.

Structural Induction: Prove $\mathbf{P}$ holds for all elements of a recursively defined set $\mathbf{X}$:

1. **Base Case:** Prove that every smallest or simplest element of $\mathbf{X}$satisfies $\mathbf{P}$.
2. **Induction Step:** Assume that $\mathbf{P}$ holds for smaller or simpler elements. **[IH]**

   Prove that every element that is constructed following each of the (finitely many) ways of constructing larger or more complex elements out of the smaller or simpler ones satisfies $\mathbf{P}$.

A recursive definition for **non-empty binary trees**:

- a single node is a non-empty binary tree;
- if T_1, T_2 are two disjoint non-empty binary trees, then the tree with a new root r connected to the roots of T_1 and T_2 is a non-empty binary tree;
- if T_1 is a non-empty binary tree, then the tree with a new root r connected to the root of T_1 is a non-empty binary tree;
- nothing else is a binary tree.

Prove that every non-empty binary tree has one more node than edge.

Solution: Recall the recursive definition for the set of **non-empty binary trees**:

- a single node is a non-empty binary tree;
- if T_1, T_2 are two disjoint non-empty binary trees, then the tree with a new root r connected to the roots of T_1 and T_2 is a non-empty binary tree;
- if T_1 is a non-empty binary tree, then the tree with a new root r connected to the root of T_1 is a non-empty binary tree;
- nothing else is a binary tree.

Consider the following recursively defined set $S \subseteq \mathbb{N}^2$:

- $(0, 0) \in S$;
- if $(a, b) \in S$, then so are $(a + 1, b + 1)$ and $(a + 3, b)$;
- nothing else belongs to S.

Consider the following recursively defined set $S \subseteq \mathbb{N}^2$:

- $(0, 0) \in S$;
- if $(a, b) \in S$, then so are $(a + 1, b + 1)$ and $(a + 3, b)$;
- nothing else belongs to S.

Show that for all $(x, y) \in S$, $x \geq y$ and 3 divides $x - y$.

Consider the following recursively defined set E

- $x, y, z \in E$;
- if $e_1, e_2 \in E$ then the following four expressions are also in E:

$$(e_1 + e_2), (e_1 - e_2), (e_1 \times e_2), (e_1 \div e_2)$$

- nothing else belongs to E.

Structural Induction – Example

Consider the following recursively defined set E

- $x, y, z \in E$;
- if $e_1, e_2 \in E$ then the following four expressions are also in E:

$$(e_1 + e_2), (e_1 - e_2), (e_1 \times e_2), (e_1 \div e_2)$$

- nothing else belongs to E.

If e is a string, $vr(e)$ denotes the number of variable (i.e., x, y, z) occurrences, and $op(e)$ denotes the number of operator (i.e., $+, -, \times, \div$) occurrences in e.

Consider the following recursively defined set E

- $x, y, z \in E$;
- if $e_1, e_2 \in E$ then the following four expressions are also in E:

$$(e_1 + e_2), (e_1 - e_2), (e_1 \times e_2), (e_1 \div e_2)$$

- nothing else belongs to E.

If e is a string, $vr(e)$ denotes the number of variable (i.e., x, y, z) occurrences, and $op(e)$ denotes the number of operator (i.e., $+, -, \times, \div$) occurrences in e.

Prove that for any $e \in E$, $vr(e) = op(e) + 1$.