Multiple Environments

Life Cycle of a User-Defined Function

What happens?

A new function is created!
Name bound to that function in the current frame

Operator & operands evaluated
Function (value of operator) called on arguments
(values of operands)

A new frame is created!
Parameters bound to arguments
Body is executed in that new environment

Names Have No Meaning Without Environments

Every expression is evaluated in the context of an environment.
A name evaluates to the value bound to that name
in the earliest frame of the current environment
in which that name is found.

An environment is a sequence of frames.
• The global frame alone
• A local, then the global frame
Names Have Different Meanings in Different Environments

A call expression and the body of the function being called are evaluated in different environments.

Every expression is evaluated in the context of an environment.

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

Conditional Statements

Compound statements:

- `<header>`:
- `<statement>`
- `<statement>`
- `<statement>`
- `<statement>`
- `<separating header>`:
- `<statement>`
- `<statement>`
- `<statement>`
- `<statement>`
- `<statement>`

Execution Rule for a sequence of statements:

- Execute the first statement
- Unless directed otherwise, execute the rest

Conditional Statements

Definition of a conditional statement:

- If a condition is true, execute the suite. If not, execute the next clause.

Execution Rule for Conditional Statements:

1. Always starts with "if" clause.
2. Zero or more "elif" clauses.
3. Zero or one "else" clause, always at the end.

Boolean Contexts

- False values in Python: False, 0, '', None
- True values in Python: Anything else (True)

George Boole

Reading:

http://composingprograms.com/pages/15-control.html#conditional-statements
While Statements

Execution Rule for While Statements:
1. Evaluate the header’s expression.
2. If it is a true value, execute the (while) suite, then return to step 1.