Lecture 26: Variable Scope

Understanding Local, Global, and Enclosing scopes.
The LEGB rule.

Agenda

What is Scope?
Local Variables
Global Variables
The ‘global‘ Keyword
Enclosing Scope & ‘nonlocal‘
The LEGB Rule
Common Pitfalls

What is Scope?

Scope refers to the region of code where a variable is accessible.
Variables created inside a function are not accessible outside of it.
Python resolves variable names using the LEGB rule (Local, Enclosing, Global, Built-in).

Local Variables

Variables defined inside a function have local scope.
They are destroyed when the function returns.

Global Variables

Variables defined at the top level of a script/module.
Accessible from anywhere within the file.

Shadowing Global Variables

If you assign to a variable inside a function, Python assumes it's a local variable.
This can shadow a global variable with the same name.

The ‘global‘ Keyword

To modify a global variable inside a function, use the ‘global‘ keyword.

Enclosing Scope (Nested Functions)

Functions can be defined inside other functions.
The inner function can access variables in the outer function's scope (Enclosing scope).

The ‘nonlocal‘ Keyword

Used to modify a variable in an enclosing (non-global) scope.

The LEGB Rule Summary

When you reference a variable, Python searches in this order:

1. Local (inside current function)
2. Enclosing (inside enclosing functions)
3. Global (top level of module)
4. Built-in (Python's built-in namespace)

Summary

Scope protects variables from accidental modification.
Understand LEGB to avoid NameErrors and shadowing bugs.
Use ‘global‘ and ‘nonlocal‘ sparingly.

Next Lecture

Next, we look at the Python Standard Library and Built-in Functions.