CS ELECTIVE
Malaluan, Nervin C.
BSCS-3C

Understanding Variables in Python:

What is a Variable?

In Python, a variable is essentially a named storage container that holds a value. You can think of it like a label attached to a box where you can store different things depending on the label. The variable name acts as the label, and the value is what's stored inside the box.

Fundamentals of Python Programming:

Python is a dynamically typed language, meaning you don't explicitly declare the data type of a variable when you create it. The type is automatically determined by the value assigned to the variable. This makes Python more flexible and concise, but it's important to understand the different data types:

- Numbers: Integers (whole numbers) and floats (decimal numbers).
- Strings: Text enclosed in quotes (single or double).
- bBooleans: True or False values.
- Lists: Ordered collections of items enclosed in square brackets.
- Tuples: Immutable ordered collections of items enclosed in parentheses.
- Dictionaries: Unordered collections of key-value pairs enclosed in curly braces.

Rules for Declaring a Variable:

- Variable names can consist of letters, numbers, and underscores, but cannot start with a number.
- Names cannot be Python keywords (reserved words with specific meanings).
- Names are case-sensitive (e.g., age is different from Age).
- Use meaningful names to improve code readability.

Keywords in Python:

Python has numerous keywords for various functionalities. Some relevant to variables include:

- def: Defines a function.
- class: Defines a class.
- if, elif, else: Conditional statements.
- for, while: Loops.
- return: Returns a value from a function.

Rules for Local and Global Variables:

Local Variables:

Scope:

Variables defined within a function are local to that function and cannot be accessed

Lifetime:

The lifetime of a local variable is limited to the duration of the function or block where it is defined. Once the function or block execution is complete, the local variable is destroyed.

Shadowing:

A local variable can have the same name as a global variable, but it will "shadow" the global variable within the local scope. The local variable takes precedence over the global variable within the function or block.

Global Variables:

Scope:

Variables defined outside any function are global and accessible throughout the program. Use the global keyword cautiously to modify globals inside functions.

Lifetime:

The lifetime of a global variable extends to the entire duration of the program's execution. It persists until the program terminates or until the variable is explicitly modified or deleted.

Accessing Global Variables from a Function:

To access a global variable from within a function, you need to use the global keyword to indicate that the variable is a global one. Without using global, a function will treat the variable as local and create a new local variable with the same name.

Operators:

Arithmetic: +, -, *, /, etc. for calculations.
Comparison: ==, !=, <, >, etc. for comparisons.
Logical: and, or, not for combining conditions.
Assignment: =, +=, -=, etc. for assigning and modifying values.