Variables & The Three Core Data Types

AiCE Software Engineering Foundations — Python Programming

A program that cannot remember anything is a program that cannot think. Variables are Python's memory — the mechanism by which your code stores, labels, and retrieves information. This lesson introduces variables and the three data types every beginner needs first: strings, integers, and floats.

1. What Is a Variable?

A variable is a named container that holds a value. Think about how a market stall is organized in Lagos, Kumasi, or Dar es Salaam — the trader has different boxes and bags, each holding a different product, each labeled so the trader knows exactly where to look when a customer asks for something.

A variable works the same way. You give it a name (the label) and store a value inside it (the product). Whenever you need that value later in your program, you refer to the name, and Python fetches the value from memory. Without variables, you'd have to retype every value in full, everywhere it's used — and update every occurrence by hand whenever it changed.

2. Creating Your First Variables

Create a new file:

```
vim variables.py
```

Enter Insert Mode and type:

```
name = "Adesina"
print(name)
```

Running this prints:

```
Adesina
```

name = "Adesina" creates a variable called name and stores the value "Adesina" inside it. The equals sign (=) is the assignment operator — it means 'store the value on the right into the container on the left,' not 'equal to' in the mathematical sense. print(name) then fetches whatever is inside the name container and displays it.

Variable Name vs. Variable Value — A Crucial Distinction

```
name = "Adesina"
print(name)
print("name")
```

This produces:

```
Adesina
name
```

print(name), with no quotes, fetches the value stored in the variable and displays Adesina. print("name"), with quotes, treats the word as plain text and displays the literal word name.

> **Quotes Mean Text — No Quotes Mean Variable**
> *Whenever you see something inside quotation marks, Python treats it as raw text. Whenever you see a word without quotation marks (that isn't a number or keyword), Python treats it as a variable name and looks up its value. Getting this right is fundamental.*

3. Storing Multiple Variables

```
name = "Adesina"
age = 26
country = "Nigeria"

print(name)
print(age)
print(country)
```

This produces three lines of output. Notice that name and country sit inside quotation marks, but age does not — this is our first hint at data types.

4. What Are Data Types?

A data type is the category of information stored in a variable. Python needs to know what kind of data it's dealing with because different types behave differently and support different operations — you cannot subtract one word from another, or do mathematics with a sentence.

Data Type	Python Name	Examples	Uses Quotes?
String	str	"Lagos", "Hello", "AiCE"	Yes
Integer	int	26, 100, 2026, 54	No
Float	float	3.14, 5.9, 36.6, 0.75	No

Strings — Text Data

A string is any piece of text — letters, words, sentences, or a single character. Strings are always written inside quotation marks, either double or single:

```
name = "Adesina"
country = 'Nigeria'
slogan = "Restoring Africa's Global Leadership Through
Technology"
greeting = "Welcome to AiCE"
```

Mixing Quote Styles Deliberately

The slogan variable uses double quotes on the outside and a single apostrophe inside the word Africa's. This is intentional — if the outer quotes and the inner apostrophe were both single quotes, Python would be confused about where the string actually ends.

Integers — Whole Number Data

An integer is any whole number — positive, negative, or zero — with no decimal point, and written without quotation marks:

```
age = 26
year_founded = 2025
number_of_countries_in_africa = 54
```

Floats — Decimal Number Data

A float is a number containing a decimal point. The name comes from 'floating-point number,' referring to how computers store decimal values in memory:

```
height_metres = 1.82
pi = 3.14159
body_temperature_celsius = 36.6
```

5. Combining All Three Data Types

Real programs mix data types constantly. Consider an AiCE platform profile:

```
# AiCE Platform Profile
platform_name = "AiCE Community"
year_founded = 2025
student_satisfaction_score = 4.8

print(platform_name)
print(year_founded)
print(student_satisfaction_score)
```

Comments

The line starting with # is a comment — text that Python ignores completely. Comments exist for the programmer, to explain or organize code. Any line starting with # is a note, not an instruction.

6. Checking Data Types with type()

Python's built-in type() function tells you the data type of any variable or value — extremely useful whenever you're unsure what you're working with:

```
country = "Kenya"
population_millions = 56
gdp_growth_rate = 5.4

print(type(country))
print(type(population_millions))
print(type(gdp_growth_rate))
```

This produces:

```
<class 'str'>
<class 'int'>
<class 'float'>
```

type() Output	Meaning	Full Name
<class 'str'>	String — text data	String
<class 'int'>	Integer — whole number	Integer
<class 'float'>	Float — decimal number	Floating-point number

Whenever you're unsure what type a variable is, wrap it in type() and print the result — this habit saves you from many confusing errors as your programs grow.

Key Takeaways

A variable is a labeled container; the value with no quotes fetches what's stored, while the same word in quotes is just literal text. Python's three foundational data types — str, int, and float — each behave differently, and type() lets you check which one you're dealing with at any point. The next lesson builds on this foundation with the rules and conventions for naming variables well, updating them over time, and assigning several at once.