# Setting Up & Running Your First Python Program

*AiCE Software Engineering Foundations — Python Programming*

Every journey in programming begins with a single question: what language should I learn? This lesson answers that question, explains why Python is the right starting point, and gets you set up and writing your first line of code.

## 1. What Is a Programming Language?

You communicate with other people using a spoken or written language — Yoruba to a friend in Ibadan, Swahili to a colleague in Nairobi, Twi to a neighbor in Kumasi. The language is the bridge that carries instructions and information from your mind to theirs.

A programming language works the same way — it is the bridge between your mind and a computer. You write instructions using that language, and the computer reads those instructions and carries them out. At the hardware level, a computer only understands electrical signals — ones and zeros. Programming languages exist so humans can give computers complex instructions without needing to think in ones and zeros.

## 2. What Is Python?

Python is a high-level, general-purpose programming language created by Guido van Rossum, a Dutch software engineer, first released in 1991. The name has nothing to do with the snake — it comes from the British comedy group Monty Python, which van Rossum was a fan of.

High-level means Python is far removed from the raw machine code computers actually execute — it handles low-level details like memory management automatically, so you can focus on solving problems. General-purpose means Python isn't built for just one kind of task; it's used across an enormous range of fields.

| Field | What Python Does There |
|---|---|
| Web Development | Powers backend systems and APIs (Django, Flask, FastAPI) |
| Data Science | Analyzes, cleans, and visualizes data (Pandas, NumPy, Matplotlib) |
| Artificial Intelligence | Builds machine learning and deep learning models (TensorFlow, PyTorch) |
| Automation | Automates repetitive tasks like file management and testing |
| Cybersecurity | Used in penetration testing and security tooling |
| Finance | Powers algorithmic trading and financial modeling |
| DevOps & Cloud | Automates cloud infrastructure and deployment pipelines |

Companies including Google, Netflix, Instagram, Spotify, and NASA rely heavily on Python. Across Africa, it's increasingly the backbone of fintech, healthtech, agritech, and edtech startups.

> **Why Python for Beginners?**
> *Python's syntax reads almost like English. print("Welcome to AiCE") is guessable even with zero programming background — it prints that text to the screen. That readability is intentional, and it's also not a sign of a 'toy' language: the same Python you'll use for your first print statement is used by engineers building AI systems and satellite software.*

## 3. Python Versions — Why We Use Python 3

Python has two major versions: Python 2, released in 2000, and Python 3, released in 2008. Python 3 introduced changes that were not backward-compatible with Python 2 — code written for one does not necessarily run in the other.

Python 2 reached its official end of life in January 2020 and no longer receives security updates or new features. The entire Python community has moved to Python 3, and this course uses Python 3 exclusively.

> **Always python3, Never python**
> *When running a Python file in the terminal, always type python3 filename.py — not just python. Some Ubuntu systems still have Python 2 installed alongside Python 3, and plain python may invoke the older version. python3 guarantees you're always running the correct, modern version.*

## 4. Checking if Python Is Installed

Open your Ubuntu terminal and check your Python version:

```
python3 --version
```

If installed, you'll see something like:

```
Python 3.12.3
```

If instead you see a 'command not found' error, install it with:

```
sudo apt update
sudo apt install python3
```

The first command updates your system's list of available packages; the second installs Python 3. Enter your password when prompted, then confirm success by running python3 --version again.

## 5. Setting Up Your Working Repository

Good developers organize their code from day one. All the Python work in this program lives inside a GitHub repository — giving you version control, a backup, and the beginning of a professional portfolio.

On GitHub, create a new repository with:

**Repository name:** aice-python-beginners

**Description:** Python for Beginners — AiCE Software Engineering Foundations

**Visibility:** Public, so your work can serve as a portfolio

Generate a Personal Access Token if you don't already have a valid one, then clone the repository into your working directory:

```
git clone
https://YOUR_TOKEN@github.com/yourusername/aice-python-beginne
rs.git
```

```
cd aice-python-beginners
```

Run ls to confirm the directory is empty — this folder will be your working directory for all the Python code ahead.

## 6. Your First Python File

Every Python file must end with the .py extension — this is how the operating system, and the python3 command, knows the file contains Python code.

Create your first file with Vim:

```
vim hello.py
```

Press i to enter Insert Mode, then type:

```
print("Hello, Africa!")
```

Press Escape, then :wq to save and quit. Confirm the file exists:

```
ls
```

## 7. Running Your First Python Program

To run a Python file, type python3 followed by the file name:

```
python3 hello.py
```

The terminal immediately displays:

```
Hello, Africa!
```

print() is one of Python's built-in functions — it displays whatever is inside its brackets. The text Hello, Africa! is wrapped in quotation marks, making it a string, a piece of text. Python read the instruction, followed it, and displayed the text.

> **print() Is Python's Voice**
> *Throughout this program, print() is your primary tool for seeing what your code is doing — checking a variable's value, confirming a calculation, or showing output to a user. Get very comfortable with it.*

## 8. Understanding What Python Actually Does

When you run python3 hello.py, here is what happens behind the scenes:

1. The operating system hands hello.py to the Python 3 interpreter.

2. The interpreter reads the file from the first line to the last, in order.

3. For each line, it translates your code into instructions the computer can execute.

4. It executes those instructions immediately, one line at a time.

5. Any output produced is displayed in the terminal.

This top-to-bottom, line-by-line execution is fundamental. Python does not skip around, and it does not run the last line first — it starts at line one and works downward. This is why the order in which you write your code matters.

Update hello.py to contain three print statements:

```
print("Hello, Africa!")
print("Python is the language of our generation.")
```

```
print("AiCE is building Africa's future, one developer at a time.")
```

Running python3 hello.py again produces the lines in exactly the order they were written — Python executed them one by one, top to bottom, exactly as expected.

## Key Takeaways

A programming language bridges human intent and computer execution, and Python's readable syntax makes it an ideal starting point without sacrificing real power. Always run files with python3, always save them with a .py extension, and remember that Python executes your code top to bottom, one line at a time — never out of order. With your environment set up and your first program running, the next lesson turns to what happens when that code doesn't run the way you expect.