Learning Python

A readable, general-purpose language good for scripts, automation, and — as shown here — Discord bots.

Why Python

Python reads close to plain English and is forgiving for beginners, but it's also genuinely used for real software — the Discord bot project on this site's coding page is written in it. It's a strong second language once you've touched JavaScript, or a strong first one on its own.

Variables and types

```
name = "Leo"
age = 18
is_developer = True
```

No `let` or `const` — Python variables are just assigned directly. You can technically reassign one to a different type entirely, though it's best not to.

Functions

```
def greet(name):
    return f"Hello, {name}!"

print(greet("Leo"))  # Hello, Leo!
```

Indentation isn't a style choice in Python — it's how the language knows where a function or block starts and ends, instead of curly braces.

Conditionals and loops

```
if age >= 18:
    print("Adult")
else:
    print("Minor")

for i in range(5):
    print(i)
```

Same ideas as JavaScript, different punctuation — that's generally true across languages once you've actually learned one properly.

Lists and dictionaries

```
tags = ["HTML", "CSS", "JavaScript"]
project = {"name": "AXIOM", "tags": tags, "year": 2026}

print(project["name"])       # AXIOM
for tag in project["tags"]:
    print(tag)
```

Lists hold ordered collections; dictionaries hold labeled data, like a single database entry. Most real programs are built almost entirely out of these two structures, nested inside each other.

A small real example — a Discord command

```
import discord
from discord.ext import commands

bot = commands.Bot(command_prefix="!")

@bot.command()
async def hello(ctx):
    await ctx.send(f"Hey {ctx.author.name}!")

bot.run("YOUR_BOT_TOKEN")
```

That's a working Discord command — typing `!hello` in a server the bot is in gets a reply. Real bots grow from this same pattern: one decorated function per command, added one at a time.

Where to go from here

- Automate something small and mildly annoying first — renaming files, pulling data out of a spreadsheet. Python is genuinely good at this, and it's motivating.
- Don't skip virtual environments (`python -m venv`) once you start installing packages — it keeps projects from interfering with each other.
- Read error messages fully. Python's are usually specific about the exact line and the exact problem.
- The AI guide below applies just as well here — pasting a traceback and asking what it means and how to fix it is one of the fastest ways to learn.