Introduction to Python Programming

Welcome to the fascinating world of Python, one of the top 3 most popular programming languages projected for 2025. Created by Guido van Rossum and first released in 1991, Python has evolved into a versatile language powering web applications, data analysis, artificial intelligence, automation and countless other technological innovations.

N **by Naresh Chaurasia**

Why Learn Python?

Python stands out among programming languages for several compelling reasons:

- Its simple, clear and readable syntax makes it ideal for beginners
- Versatile support for multiple programming paradigms including procedural, object-oriented and functional programming
- Massive online community offering support and extensive libraries for nearly any task
- Growing industry demand with excellent career prospects and competitive salaries

Setting Up Python

Download Python

Visit python.org to download the latest version (3.13) for your operating system. Python works seamlessly across Windows, macOS and Linux platforms.

Choose an Editor

Select a code editor that suits your needs. Options include the built-in IDLE, Visual Studio Code with Python extensions, or PyCharm for a full IDE experience.

Write Your First Program

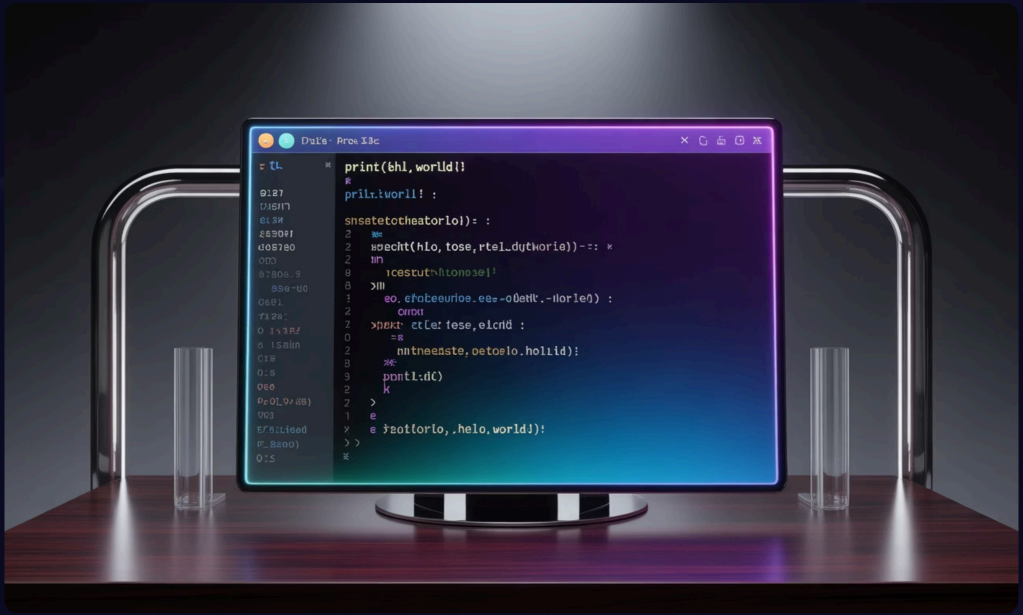
Create a file named hello.py and write: **print("Hello, World!")** - Then run it to see your first Python program in action!

Installing Python 3.12.2 – please wait...

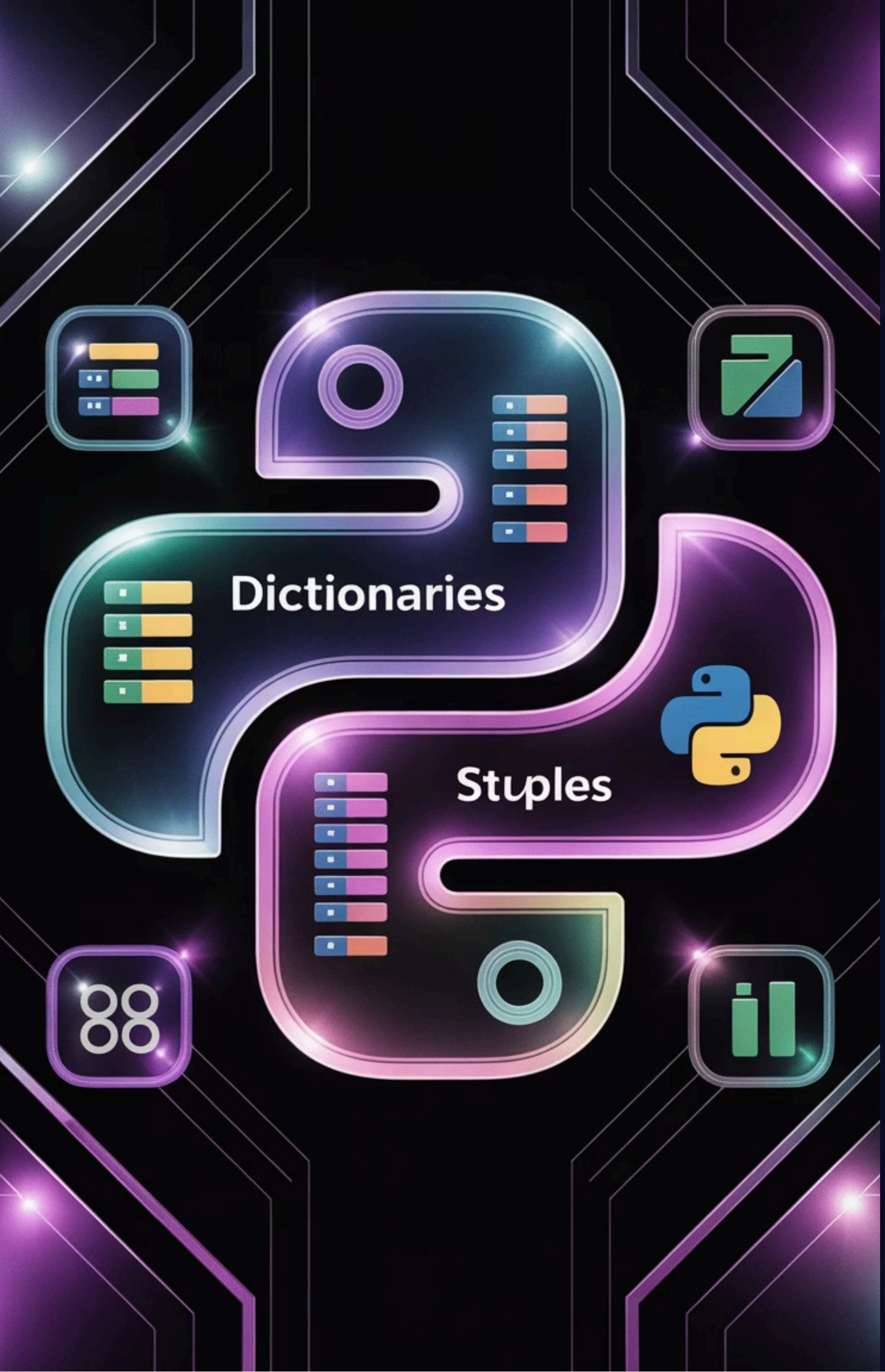
Python Syntax Basics

Python's syntax is what makes it stand out from other programming languages:

- Code blocks are defined by indentation (typically 4 spaces), not braces or keywords
- Statements end with a newline rather than semicolons
- Comments begin with the # symbol and extend to the end of the line
- Variable declaration doesn't require specifying data types



```
# This is a comment
name = "Alice"  # Variable assignment
if name == "Alice":
    print("Hello, Alice!")
    print("Welcome to Python")
else:
    print("Hello, stranger!")
```



Variables and Data Types

Numeric Types

int: Whole numbers like 42 or -7

float: Decimal numbers like 3.14 or -0.001

complex: Complex numbers like 3+4j

Text Type

str: Text strings like "Hello" or 'Python'

Supports single or double quotes

Multi-line strings use triple quotes

Collection Types

list: Ordered, mutable [1, 2, 3]

tuple: Ordered, immutable (1, 2, 3)

dict: Key-value pairs {"name": "Alice"}

set: Unordered, unique {1, 2, 3}

Control Flow: Conditionals and Loops

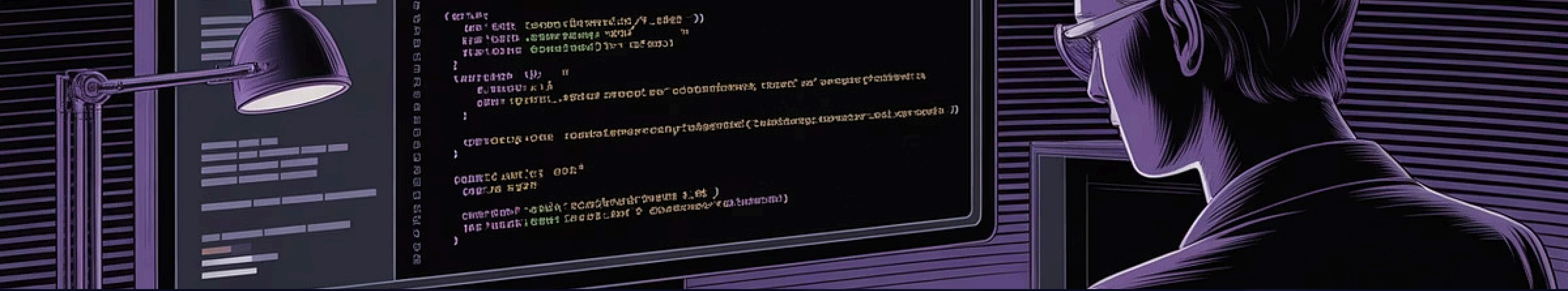
Conditional Statements

```
# Decision making with if-elif-else
x = 10
if x > 15:
    print("x is greater than 15")
elif x > 5:
    print("x is greater than 5")
else:
    print("x is 5 or less")
```

Loop Structures

```
# For loop example
for i in range(1, 6):
    print(i)

# While loop example
count = 1
while count <= 5:
    print(count)
    count += 1
```



Functions and Modular Code

Define Functions

Use the **def** keyword to create reusable blocks of code that accept parameters and return values.

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

Import Modules

Extend Python's capabilities by importing built-in or third-party modules.

```
import math
import random
from datetime import datetime
```

Organise Your Code

Create your own modules and packages to structure larger projects for better maintainability.

Summary and Next Steps

What We've Covered

- Python's popularity and versatility
- Setting up your development environment
- Basic syntax and programming constructs
- Variables, data types, and control structures
- Functions and code organisation

Continue Your Python Journey

- Practice with small personal projects
- Explore online tutorials and documentation
- Dive deeper into data structures
- Learn object-oriented programming principles
- Explore specialised libraries for your interests