

Hello World

A Quick Introduction to Computers, Programming, and Python

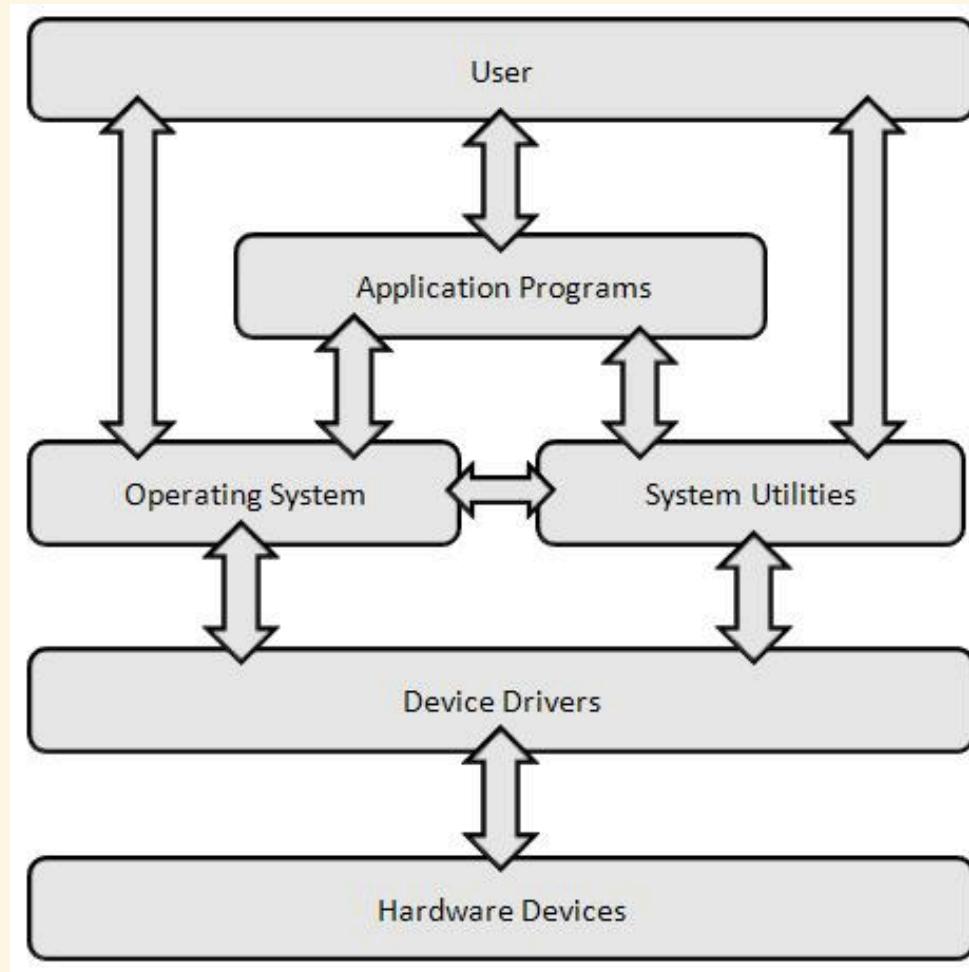
What Really is a Computer?

- Electronic machine that follows instructions
- Takes **input** → processes it → gives **output**
- Only understands **1s and 0s** (binary)
- Everything else is translation



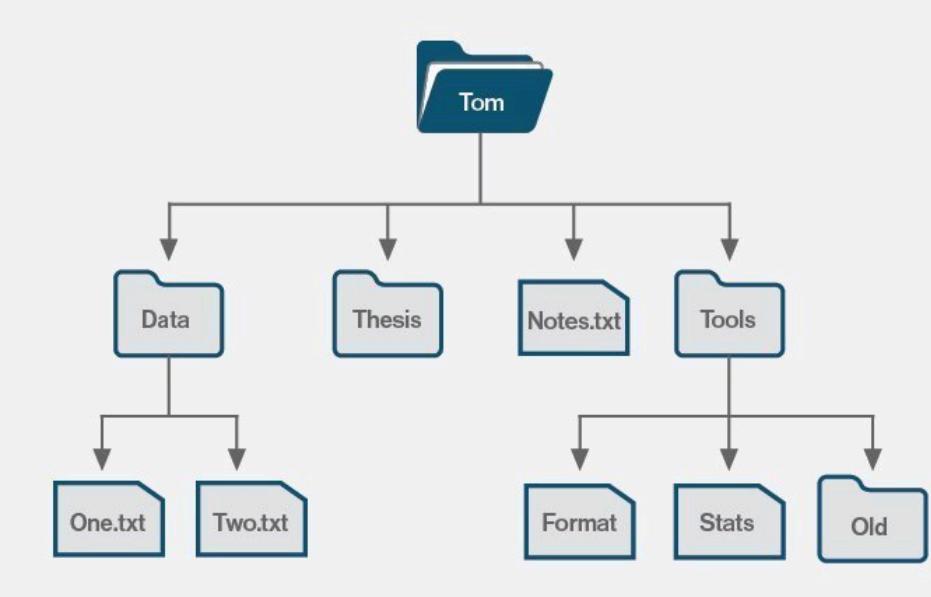
Modern Computers are Layers of Abstraction

- **Hardware:** Physical components (CPU, RAM, etc.)
- **Operating System:** Manages hardware (Windows, macOS, Linux)
- **Programming Languages:** Human-readable instructions (Python, Java, etc.)
- **Applications:** Software that performs tasks (web browsers, games, etc.)



What is a File?

- **Collection of data** stored on your computer in a **tree-like structure**
- Has a **path**, **name**, and **extension**
- File extensions indicate type:
 - `.txt` = text
 - `.py` = Python code
 - `.jpg` = image
- Files can be `created`, `read`, `written`, and `deleted`
- Some files can be `executed`, these are programs.



Source Code vs Machine Code

Source Code

- **Human-readable** instructions we write
- Looks like:

```
print("Hello World")
```

Machine Code

- **Computer-readable** instructions
- Looks like:

```
48 89 e5      mov    %rsp,%rbp  
48 83 ec 10   sub    $0x10,%rsp  
bf 01 00 00   mov    $0x1,%edi
```

- **Source code** is translated to **machine code** by a **compiler**.



C_{PYTHON} Interpreter

- What actually runs Python code

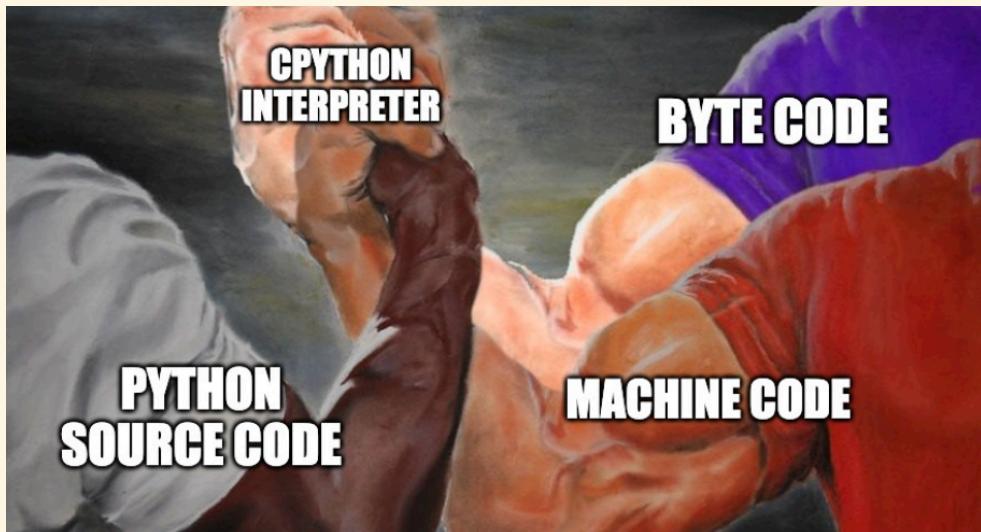
- **Two-step process:**

1. Python source → **bytecode**
2. Bytecode → **machine code**

- Looks like:

```
0 LOAD_GLOBAL          0 (print)
2 LOAD_CONST           1 ('Hello, World!')
4 CALL_FUNCTION        1
6 RETURN_VALUE
```

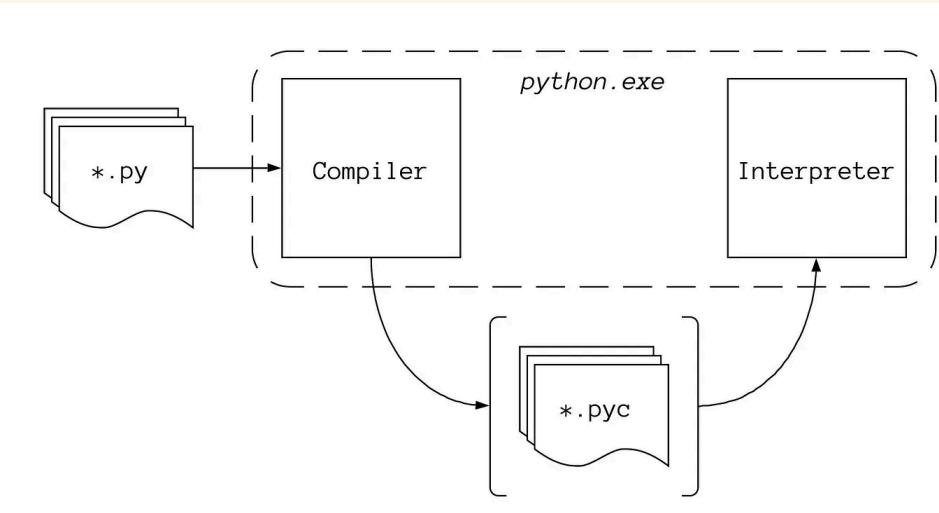
- Why two steps? **Portability** - same Python code runs everywhere.



How Python Gets Executed

1. You write Python code in a `.py` file
2. CPython interpreter reads your code
3. Converts it to bytecode (behind the scenes)
4. Computer executes the bytecode
5. You see the result

Python handles the translation for you!



Basic Input/Output

Input

- **Getting information** from the user
- Keyboard, mouse, files, internet

Output

- **Showing information** to the user
- Screen, speakers, files, internet

In Python Code

- `input()` - get text from user
- `print()` - show text to user

Output

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

Input and Output

```
name = input("Enter your name: ")  
print(f"Hello, {name}!")
```

A Byte of Python

- Written by **Swaroop Chitlur**
- All my slides for teaching Python are adapted from this book!
- Licensed under a **Creative Commons License**
- Creative Commons Attribution-Noncommercial- Share Alike 3.0 Unported License
- <https://python.swaroopch.com/>

The screenshot shows the homepage of the "A Byte of Python" website. On the left, a sidebar contains a search bar and a list of topics: Introduction, Dedication, Preface, About Python, Installation, First Steps, Basics, Operators and Expressions, Control flow, Functions, Modules, Data Structures, Problem Solving, Object Oriented Programming, Input and Output, Exceptions, Standard Library, More, What Next, Appendix: FLOSS, and Appendix: About. The main content area features the title "A Byte of Python" and a brief description: "'A Byte of Python' is a free book on programming using the Python language. It serves as a tutorial or guide to the Python language for a beginner audience. If all you know about computers is how to save text files, then this is the book for you." Below this is a section titled "For Python version 3" with a note: "This book will teach you to use Python version 3. There will also be guidance for you to adapt to the older and more common Python version 2 in the book." At the bottom, there are two sections of reviews: "Who reads A Byte of Python?" and "Here are what people are saying about the book".

A Byte of Python

"A Byte of Python" is a free book on programming using the Python language. It serves as a tutorial or guide to the Python language for a beginner audience. If all you know about computers is how to save text files, then this is the book for you.

For Python version 3

This book will teach you to use Python version 3. There will also be guidance for you to adapt to the older and more common Python version 2 in the book.

Who reads A Byte of Python?

Here are what people are saying about the book:

Somewhere around 2004 - 05 when I was convinced I wasn't smart enough to be a programmer , I came across the original A Byte of #Python, and that changed my entire perspective on computing and life . I owe a lot to that book @swaroopch had written. -- [Rahul](#) on Jul 30, 2020

This is the book that got me into programming almost a decade ago. Thank you @swaroopch. You changed my life. -- [Stefan Froelich](#) on Aug 2, 2019

I am writing this email to thank you for the great help your book has done for me! It was a really good book that I enjoyed thoroughly. As a 15 year old who has never done programming before, trying to learn Python online was difficult and I couldn't understand anything. But I felt like your book gave was much easier to understand and eased me into the whole new world of programming.

Comments

- Use `#` for comments
- They can be above:

```
# Note that print is a function
print('hello world')
```

- Or at the end

```
print('hello world') # Note that print is a function
```

Well commented code is easier to understand for future readers (including yourself).

Literal Constants

- Numbers: 5 , 1.23
- Strings: 'This is a string' , "It's a string!"
- Boolean: True , False
- Nothing!: None

Numbers

Numbers are mainly of two types - **integers** and **floats**.

Integers

- An example of an integer is `2` which is just a whole number.

Floats

- Examples of floating point numbers ("floats") are `3.23` and `52.3e-4` .
- The `e` notation indicates powers of 10. `52.3e-4` => `52.3 * 10^-4` .

Strings

- Strings are used to represent text.
- You can use single quotes, double quotes, or triple quotes for multi-line strings.

```
'Hello'  
"Hello"  
...  
  
This is a  
multi-line string  
...  
...
```

- Strings are **immutable**

An error occurred on this slide. Check the terminal for more information.

An error occurred on this slide. Check the terminal for more information.

An error occurred on this slide. Check the terminal for more information.

Variables Rules

- The first character of the identifier must be a letter of the alphabet or an underscore (_).
- The rest of the identifier name can consist of letters (uppercase ASCII or lowercase ASCII or Unicode character), underscores (_) or digits (0-9).
- Identifier names are case-sensitive. For example, `myname` and `myName` are not the same.

Examples

- `age` , `name` , `foo` , `bar`
- `age2` , `name_2` , `foo_bar`
- `_age` , `_name` - starting with `_` denotes "private"
- `AGE` , `NAME` - ALL_CAPS denotes "constant"
- `i` , `j` , `k` - common for iterators in loops (more on this later)
- `df` , `x` , `y` , `X` , `Y` - bad variables but common in data science

An error occurred on this slide. Check the terminal for more information.

Logical Operators

```
True and False
```

```
## False
```

```
True or False
```

```
## True
```

```
not True
```

```
## False
```

An error occurred on this slide. Check the terminal for more information.

REPL - Read-Eval-Print Loop

- **Interactive** way to run code
- Read your code → Evaluate it → Print the result → Loop back
- Type code, hit Enter, see result immediately
- Perfect for **experimenting** and learning

```
Welcome to the Pyodide 0.28.0 terminal emulator iQuery Terminal
Python 3.13.2 (main, Jul  4 2025 13:41:45)
on WebAssembly/Emscripten
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Jupyter Notebook

- **Web-based** coding environment
- Combines **code**, **text**, and **output** in one document
- Code goes in **cells** - run one cell at a time
- Like a **digital notebook** for programming
- Perfect for **data science** and **learning**

Think: Google Docs + Python code

The screenshot shows the Jupyter Notebook interface. At the top, there's a menu bar with File, Edit, View, Run, Kernel, Tabs, Settings, and Help. Below the menu is a toolbar with icons for creating a new file, opening, saving, and running cells. To the left is a file browser showing a directory structure with 'data', 'notebooks', and 'README.md'. The main area displays a table of files with columns for Name and Modified. On the right, there's a 'Launcher' panel with a search bar and a list of kernel options: 'Notebook' (selected), 'Python (Pyodide)', 'C++17', and 'C++20'. A toggle switch at the bottom indicates the 'Simple' mode.

Name	Modified
data	21d ago
notebooks	21d ago
README.md	21d ago

Simple

Exercise: Variables

bigd103.link/variables