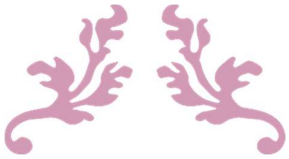


HARVARD UNIVERSITY

CS50 DEPARTMENT SINCE 1636 CE

HARVARD
UNIVERSITY

INTRODUCTION TO PROGRAMMING WITH PYTHON

DAVID J. MALAN, PROFESSOR HARVARD UNIVERSITY

"Indian Version"

Educators & Faculty

- *Yulia Zhukovets,*
- *Bernie Longboy*
- *Rongxin Lu & Brian Yu*

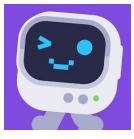
Documented by

Ravikiran R (CS50AP26031)
Meta Llama 3.2 GPT Model
Ollama x64 Offline GPT Model

"CS50 – Harvard University intellectual enterprises of Computer Science and The Art of Programming" ~ David J. Malan

CS50x India Production & Reference

Final week Reference



....more

Forward Note

Welcome to **CS50 Trimester 2025**, guided by Associative Professor, Ravi Kiran under educators' Quota. This 10-week self-paced course is designed to help you build a strong foundation in computer science. While there are no regular live sessions, weekend guidance is available upon request to support your learning when needed. A detailed syllabus will be provided, outlining all the necessary topics, resources, and practice material to help you stay on track.

At the end of the course, you'll take part in an **interview-based assessment** to evaluate your understanding. You can attempt the interview any time after completing the 10 weeks, with up to two months of extended time and two total attempts allowed. Stay consistent, make the most of the flexibility, and don't hesitate to reach out if you need support along the way. We're excited to see you grow through this journey.

Funny Facts

"Ada Lovelace" wrote the first computer program in the 1800s. She was coding before computers even existed. That's like writing Instagram posts before the internet. Analytical Engine was the first idea of a computer. It was like a steam-powered laptop… just missing the laptop part.

Languages like B, Fortran, Cobol came before C. These are the grandma and grandpa of programming. Old, slow, but still respected. C is not the first, but it's the strict parent. Make one mistake in C, and your computer might explode (okay, not really… but close).

Python and SQLite are built using C. C Language is like the parent. Python is the cool kid who made life easier. SQLite is the nerdy cousin storing everything in memory. C programmers don't show feelings. They don't cry; they just use printf("I'm sad\n");

All SQL dialects look the same but act different. Like how biryani is different in Hyderabad, Kozhikode, Chennai, and Kolkata… but still called biryani. MySQL, SQLite, Oracle, PostgreSQL – all are SQL. It's like having 5 messaging apps… all do the same thing, but everyone argues which is best.

SQL can be a file, a package, a module – whatever. SQL is like a potato. You can boil it, fry it, mash it… still the same potato. Python uses SQLAlchemy, sqlite3, mysql-connector. Python doesn't talk to databases directly - it sends a friend to do the job.

Java uses JDBC and Drive Manager, Java is like a super serious guy in a suit… even to connect to a simple database. JavaScript uses mysql, mssql, pgsql, firebase, JavaScript connects to databases like a kid trying to fix Wi-Fi — randomly pressing buttons and hoping it works.

Firebase is JavaScript's favourite, Because it's easy, flashy, and doesn't ask too many questions. SQLAlchemy (package) is like ordering food online. Writing SQL (actual command) is like cooking it yourself with a blindfold.

SQLite is everyone's backup friend. No server? No problem. SQLite runs quietly in the background like a shy but smart kid.

Index

Week –5 : Sets & Dictionaries

Sets, Introduction, Iterating over Sets, Growing Sets, Set Operations, Dictionaries, Introduction, Examples, Iterating over Dictionaries, Growing a Dictionary, Mutability, Text Processing, Number of Sentences, Number of Words, Number of Unique Words, Frequent Words, Pangrams and Dictionaries, Dictionary Methods, Dictionaries in Action: LMS, Assignment Model, Submission Model, Grader

Week –6 : Object-Oriented Programming (OOP'S)

Objects and Classes, OOP in Python: An Example, Classes and Objects, self, Class Attributes vs Object Attributes, Inheritance, Concrete Example, Parent-Child Relationship, Method Overriding, Vector: Mathematical Preliminaries, Vector: Specification, Vector: Definition, Collection of Vectors

Week –7 : Glossary & Important Functions

After successfully completing the six weeks of learning, the Glossary and Important Functions section serves as a quick-reference cheat sheet and revision guide. It summarizes key concepts, terms, and commonly used functions in one place, making it easy to recall essential topics and refresh your understanding.

Week –8 : Interview Questions & Logics

The Interview Questions and Logics section includes commonly asked Python questions and logical problems that test your understanding of core concepts. It helps strengthen your problem-solving skills by encouraging you to think critically and write efficient code.

Week –8 : Revision & Et Cetera

Optionally extend this week learning File Handling with reference of IIT Madras, IIT Bombay & IIT Guwahati.

Introduction to Programming with Python

The Beginning – *From Kattappa's Sword to Keyboard Keys!*

"No betrayals or dhokha here, only clean code and clean logic"

Python is not just a programming language — it's that chill friend who never panics during exams and still scores 90+. While other programming languages shout at you with brackets {}, semicolons ;, and *"syntax errors"* like strict tuition teachers, Python just sips chai, pats your back and says, *"Aram se kar bhai, sab ho jaayega."*

Python is simple. So simple, it feels like you're explaining things to your computer in plain English — no shouting, no tantrums, no 3-hour debugging drama. It's like giving instructions to your cousin: *"Do this bhai. Then that. And haan, don't forget this."* And boom — it listens (well… mostly).

Want to build a website? Python. Want to automate boring tasks like renaming 200 files or calculating your relatives' shaadi budget? Python. Want to look smart in front of your crush by building an app? Python. Basically, Python is the *jugaadu dost* that helps you in every subject — without asking for notes in return.

And the best part? You don't need to be a computer science topper or IITian to learn Python. If you can think straight and search stuff on Google without spelling mistakes, *congratulations* — you're already a Python programmer in the making.

So, forget the tension, stop being scared of code, and say *"Hello World!"* to the most beginner-friendly, Indian-mom-approved programming language out there. "*Python*", easy to learn, impossible to hate, and best enjoyed with samosa and stack overflow, *"Chalo, Let's start!"*.

"Baahubali had Kattappa. You've got Python. No backstabbing, just debugging!"

"Python: The Only Language Born from Laughter, Not Logic!"

Python's history isn't like Indian history — no kings, no Mughal invasions, and thankfully, no exams. Just one chilled-out Dutch guy, *Guido van Rossum*, sitting back in 1991 with his laptop in one hand and a British comedy show (*Monty Python's Flying Circus - BBC world*) on the screen. And boom — instead of naming the language something boring like *"SuperCode++ Pro Max,"* he goes, *"Let's call it Python."*

Not because he loved snakes, but because the show made him laugh — imagine creating a whole programming language during a Netflix binge! Guido then became the *"benevolent dictator for life"*, which sounds like a villain from a South Indian film, but he was just the friendly boss of Python (no evil laugh, just clean syntax).

Fast forward 30 years, and Python's still rocking — like that one uncle at every wedding who refuses to age and always knows the DJ. It's built by regular folks like us — not kings, not warriors, just keyboard ninjas with strong Wi-Fi and weak chai. And trust us, it's the only thing in the world that forgives your mistakes with indentation instead of giving *"ek tight slap."*

Python Installation in 3 Easy Steps

"Aram Se, No Tension!"

STEP 1: Visit the Land of Snakes**,** "*No, Not Kerala this time"*

Go to the official Python website,
Link : ***https://www.python.org/downloads***

Click on the big **"*Download Python 3.x.x*"** button.
No research, no confusion — just download like you download memes.

STEP 2: Open File Like a Pro, Not Panic

Double-click the downloaded file.
Tick the box that says *"Add Python to PATH"* — it's tiny but powerful.

Hit *"Install Now"*.
Wait like you're buffering your favourite YouTube video.

STEP 3: Confirm Python's Not Napping

Open **Command Prompt/CMD**
Type: *C:/Users/your_name>* *python*

If you see >>> and Python version info, "*Congrats, Coding ustaad!*"
If not, don't panic. Blame Windows or call cousin tech support.

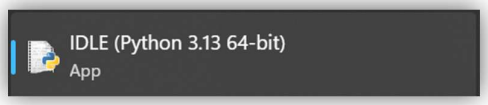
You're Done!
Welcome to the world of Python —
Where the only snake you like is the one that helps you code.

Now go ahead… start typing like a genius!

~~My~~ Your First Program

This is it — your first Python program! It's usually just a simple ***"Hello, World!"***, but don't laugh… it's basically the *namaste* of programming. It tells your computer, "Hey, I've arrived!" No complex logic, no drama — just good vibes and working code. From here, it only gets crazier (and more fun). So, type it like a pro and welcome yourself to the coding duniya!

Here is how you type your first program, *"Not Mine!"*
First find Application in your Laptop, search >>> *"IDLE",* that looks like this



Open IDLE, and you see >>> in the application, you can start typing your first program like,

```
>>> print("Hello, world!")
Hello, world!
```

Explanation:

So, what's happening here?

- `print` – This is Python's way of shouting. Not like your mom when you don't do homework, but politely telling the computer, *"Say this out loud."*
- `"Hello, world!"` – This is your computer's first friendly greeting to the world. It doesn't know much yet, so it just waves and says Hi. No swearing, no bugs — just pure innocence.
- `The brackets ()` – Like tiffin boxes, they carry what's inside. Without them, Python gets confused like us without tea.
- `The quotes ""` – These tell Python, *"Boss, this is a string, not a number or logic."* No quotes, no talk. You can also use single quotes ' ' instead of " ".

Basically, this line means,

> *"Oye Python, please display this line on screen like a good boy!"*
> …and Python happily obeys — no attitude, no debugging drama.

Prompts & Outputs

The Prompt, The Print, and The Python

When you open Python, you'll see something like this on the screen, >>>
Now don't panic — that's not an error, not a warning, not your marks — it's called a prompt.
Just like your blinking cursor in MS Word, it's politely saying,
"Bol na bhai, what should I write?"

```
>>> print("Hello, world!")
```

Then we get,

```
Hello, world!
```

Aree Wah! Your first code is working; Bas one simple line and Python speaks.

What is this "print" drama?

So, in Python, print is not like your school printer. It's a built-in function — which basically means Python already knows it, like how we already know pani puri is awesome.

Whenever you use print(), you're telling Python:
"Bhai, display this on screen."

```
>>> print("Telangana's Capital is Hyderabad")
Telangana's Capital is Hyderabad
```

See? It even handles apostrophes like a gentleman.

Quotes ka confusion? Don't worry!

You can use single quotes *'Hello'* or double quotes *"Hello"* — both work the same.
Just don't mix them up like you mix tea and coffee together.

```
>>> print('Hello, world!')
Hello, world!
>>> print("Hello, world!")
Hello, world!
```

But if your sentence has an apostrophe like *Telangana's*, then use double quotes. Why?
Because Python is smart, but not your English teacher — it'll get confused otherwise.

Printing Numbers? Easy Peasy!

```
>>> print(1)
1
>>> print(2.0)
2.0
```

Python handles numbers like a pro — no unit test needed.

Want to print multiple things together?

```
>>> print('Harvard', 'University', 'CS50')
Harvard University CS50
```

Notice the space between the words? Python puts it automatically. Like how aunties leave space in line... okay, maybe not that automatically.

Want silence? Print nothing!

```
>>> print()

>>>
```

Result: Blank line! (Shhh... peace!)

No Brackets? Python will fight back!

If you try to be over-smart and skip the brackets:

```
>>> print 'Hello, world!'
SyntaxError: Missing parentheses in call to 'print'. Did you mean print(...)?
```

Python will get angry and say:

***SyntaxError*:** `Missing parentheses in call to 'print'. Did you mean print('…')?`

See? Even Python is giving *"Did you mean...?"* like Google search. Grammar mistake, boss! In programming, this is called a syntax error — basically, your code's grammar is wrong.

Summary:

- >>> → This is Python asking you, *"Bhai, likho kuch!"*
- `print()` → This is you replying *"Suno na Python, display this line!"*.
- Use `''` or `" "`, both are fine — don't be that confused friend in group project.
- Print numbers, words, even blank lines. Python won't complain.
- But always, ALWAYS use brackets — warna Python ka mood kharab ho jaayega!

Literals & Variables

Literals & Variables – Fixed Values and Chalu Boxes!

In Python, when you write things like *'Hello World!', 1, or 2.0*, they are called *Literals*.

No, not "literally" your best friends — they're just fixed values. They don't change; they sit quietly like a topper on the front bench.

Now comes the fun part — variables. Think of variables as containers or boxes where you can keep values. Like your tiffin box — today poha, tomorrow maggi… same box, new item!

In Python, you can create a variable like this:

```
>>> x = 1
>>> print(x)
1
```

Easy na? Here, **x** is your container, and **1** is the thing inside.

Want to store a string?

```
>>> y = 'a string'
>>> print(y)
a string
```

Want to be fancy?

```
>>> foo_bar = 123.456
>>> print(foo_bar)
123.456
```

(Yes, variable names can look like a Wi-Fi password. Just don't start with numbers.)

Now what is that = sign doing? That's not *"equal to"* like in maths — in Python, it's called the *assignment operator*. It means *"Put this thing into that box."*

You can use it in two ways:

1. To create a new box (variable)
   x = 1 → means: *"Put 1 into x."*

2. To update an existing box
   x = x + 1 → means: *"Add 1 to whatever is in x, and put it back in x."*

```
>>> x = 1
>>> x = x + 1
>>> print(x)
2
```

And remember: Python reads this like we read masala dosa — right to left. First it calculates what's on the right side, then puts the result into the left side.

So, bas! That's how variables work. They're flexible, reusable, and unlike relatives, they don't ask too many questions.

More details on variables coming soon in next week— tab tak, keep practicing and *don't forget your = ka asli kaam!*

Basic Data types

Basic Data Types in Python – Meet the Four Main Characters!

In Python, there are 4 basic data types you'll meet again and again. Think of them as the ***main cast*** in your coding movie:

1. Integer – The Whole Number Hero

An **integer** is just a normal number — no decimal, no drama. Like 1, 99, -47, or *how many samosas you ate?* (don't lie).

Want to check if something's an integer?

```
>>> print(1)
1
>>> type(1)
<class 'int'>
```

See? *int = integer = asli number*.

2. Float – The Number with a Decimal Point

A **float** is a number with a dot. Not floating in water, but floating-point — that's coder lingo for decimals.

```
>>> print(1.0)
1.0
>>> type(1.0)
<class 'float'>
```

Even this is valid:

```
>>> print(1.)
1.0
```

So, *1. and 1.0* are same-same — like chai in glass or pot cup, still chai.

3. String – The Chatterbox

A **string** is a bunch of letters, words, or anything inside quotes. Basically, Python's way of letting you gossip.

```
>>> print("string")
string
>>> type('string')
<class 'str'>
```

Single quotes or double quotes — *dono chalega! But no mixing, samjhe*?

4. Boolean – The Yes or No Type

A **boolean** is like your mom during exams: either **True** or **False**, no in-between. Python uses,

```
>>> print(True)
True
>>> type(False)
<class 'bool'>
```

Warning: **Don't write true or false in small letters** — Python will throw tantrums. It only understands **True** and **False** (capital T, capital F — like VIP treatment).

Quick Recap,

- `int` = whole numbers (like how many times you hit snooze)
- `float` = numbers with dots (like *petrol price: 100.23*)
- `str` = words in quotes (like *"I love Python"*)
- `bool` = True or False (like *Did you eat the last laddu?*)

Comments

Comments – The Gossip Python Ignores!

In Python, a comment is like that one friend who keeps talking during class — but the teacher (Python) completely ignores them.
You make a comment by starting the line with a # symbol.

```
>>> # This is a comment
>>> # print(1)
>>>
```

See that second line? It has code, but since it starts with #, Python says *"Nope, I didn't see anything."* So, nothing happens. Peace!

You can even put a comment after your code:

```
>>> print(1) # This line printting the values 1
1
>>>
```

Python happily prints 1, and totally ignores the gossip after #. Full focus, no distractions!

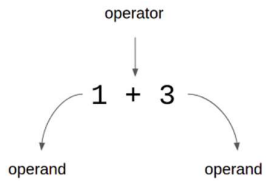
Why use comments?

- To explain your code to yourself (especially when you forget what you wrote 3 days ago)
- To make it readable for others
- To disable code without deleting it (a.k.a. *"temporary break-up"*)

So, remember: #, whispering in Python's ear. It hears nothing, but you remember everything.

Operators

The anatomy of an operation is given below:



Arithmetic Operators – Python's Maths Masala!

Python knows math, and just like us during exams, it also uses operators to do the calculations. These are the special symbols that help Python solve problems — without needing a calculator or tuition!

Here's the math gang in Python,

Operator	What It Does?
+	Addition
-	Subtraction
*	Multiplication
/	Division
//	Floor division
%	Modulus
**	Exponentiation

Let's See Them in Action Like a Maths Movie:

```
>>> 10 + 5 # Addition
15

>>> 10 * 5 # Multiplication
50

>>> 10 % 5 # Modulus
0

>>> 10 // 5 # Floor Division
2
```

```
>>> 10 - 5 # Subraction
5

>>> 10 / 5 # Division
2.0

>>> 10 ** 5 # Exponentation
100000
```

Wait, Wait! What's This New Stuff?

// → *Floor Division*

Only gives the whole number part of division.

```
>>> 8 // 3
2
```

% → Modulus

Gives the remainder, like how much money is left after buying samosas.

```
>>> 10 % 3
1
```

** → Exponentiation

Like Maths teacher shouting: *"2 to the power of 3!"*

```
>>> 2 ** 3
8
```

What's the Difference Between / and //?

```
>>> 5 / 2
2.5
```

```
>>> 5 // 2
2
```

One gives full result (like your honest friend), the other gives only the integer part (like a friend who keeps the change).

Now Let's Mix in Variables!

```
>>> x = 1
>>> y = x * 5
>>> print(x, y)
1 5
```

Bas! That's how Python does math. Clean, smart, and no calculator required.

So now, maths in Python is no more-scary. It's just another masala dosa of operators — tasty, logical, and fully understandable!