POCKET GUIDE · MINI

₹49 EDITION

Everything You Keep Googling. One Pocket Book.

Was it `.append()` or `.add()`? What does that `TypeError` actually mean? The core of Python, in visual spreads — no walls of prose.

WEKNOWWHATYOURELOOKINGFOR.COM

Pocket Insight Guide

WHAT'S INSIDE

The core language, in pointers

This mini pulls the highest-leverage pages from the full 50-page reference: the container comparison, the f-string cheat sheet, and the gotchas that bite every beginner. Fifteen minutes and you'll already look things up faster.

01 Python in 2026 — the honest ranking p.3

02 Variables, types & strings p.5

03 The four containers, compared p.7

04 Control flow & comprehensions p.9

05 Functions & classes p.10

06 Errors, files & modules p.11

07 Standard library greatest hits p.12

08 Idioms & gotchas — the chapter that pays for the book p.13

THE ONE IDEA TO HOLD THE WHOLE WAY

A Python variable is a name pointing at an object — never a box holding data. `b = a` copies the arrow, not the object. Hold that one picture and half of Chapter 8's gotchas disappear on their own.

CHAPTER 01

Python in 2026 — the honest ranking

- **#1 on TIOBE, July 2026:** 18.94% — an 8-point lead over #2 (C). Still dominant, still accelerating.
- **The honest asterisk:** TypeScript overtook Python in GitHub contributors in Aug 2025 — AI-assisted coding favors typed languages. Both facts are true at once.
- **The three-thirds map:** one third data & AI, one third web, one third automation-and-everything-else. The same core language powers all three.
- **Where it doesn't lead, honestly:** mobile apps, browser front-ends, game engines, hard real-time systems.
- **The five-minute setup:** install 3.13+/3.14 from python.org · meet the friendly new colored REPL · VS Code + the Python extension · run your first script.
- **One habit that prevents 90% of beginner pain:** a virtual environment (`venv`) per project — full workflow card in Chapter 6.

REALITY CHECK

~½ of all new AI repositories on GitHub are Python — 582K projects. Demand for the skill grew +153% YoY, more than any other language.

CHAPTER 01 · CONTINUED

The standing, in numbers

18.94%
TIOBE #1, Jul 2026 — 8pt lead over C

+7 pts
YoY usage jump — Python's biggest ever

+153%
YoY growth in Python skill demand

INDEX	RANK	THE DETAIL
TIOBE, Jul 2026	**#1 · 18.94%**	8+ pts ahead of #2; peaked near 27% in 2025
Stack Overflow 2025	**#4 usage · 57.9%**	Behind JS/HTML/SQL — but +7pt jump was its biggest ever
GitHub Octoverse 2025	**#2**	+850K contributors (+49% YoY); ~half of new AI repos

Typical US junior Python developer salary: **$70–95K**; mid-level ≈ **$125K**. Source: TIOBE, Stack Overflow Survey 2025, GitHub Octoverse 2025, ZipRecruiter/Glassdoor/Indeed 2026.

REMEMBER

Solving a real problem across three domains doesn't mean Python leads everywhere. Both the dominance and the asterisk are true at once.

CHAPTER 02

Variables, types & strings

- **Names, not boxes:** a variable is a label pointing at an object. `b = a` copies the arrow, not the object — two names, one object.
- **Ints never overflow:** Python integers have arbitrary precision, so `2 ** 1000` just works. Use underscores for readability: `1_000_000`.
- **The 0.1 + 0.2 problem:** `0.1 + 0.2 == 0.3` → `False`. Fix with `round()`, `math.isclose()`, or `decimal.Decimal` for money.
- **f-strings are the only formatting you need:** `f"{name} is {age}"` — any expression works inside `{}`.
- **The debug shortcut:** `f"{value=}"` prints `value=42` — name and value, free (3.8+).
- **Type checks:** `isinstance(x, int)` beats `type(x) == int` — it also accepts subclasses.

WATCH OUT

`//` floors toward negative infinity, not toward zero: `–7 // 2 == –4` (not –3). Coming from C or JavaScript, this bites.

CHAPTER 02 · CONTINUED

The format mini-language, on one page

CODE	RESULT
`f"{3.14159:.2f}"`	**'3.14'**
`f"{1234567:,}"`	**'1,234,567'**
`f"{0.25:.1%}"`	**'25.0%'**
`f"{'hi':>10}"`	**right-align w10**
`f"{'hi':^10}"`	**center**

- **Specs combine:** `f"{total:>12,.2f}"` → right-aligned, comma-grouped, two decimals. Perfect for aligned report columns.
- **Strings are immutable:** every method returns a new string. When you only need yes/no, prefer `"a" in "cat"` over `.find()`.
- **Slicing, decoded:** indices sit between characters, not on them. `s[1:4]` grabs everything between fence 1 and fence 4 — end exclusive.
- **The reverse trick:** `s[::−1]` reverses a string — the classic interview one-liner.

LEGACY, RECOGNIZE DON'T WRITE

Old code uses `"{} is {}".format(...)` or `%s % (...)`. Recognize them in old codebases; never write them yourself.

CHAPTER 03 · THE BOOK'S CENTERFOLD

The four containers, finally compared

90% of Python data handling is picking the right one of these four and knowing its five key methods.

	LIST	TUPLE	SET	DICT
Literal	`[1, 2]`	`(1, 2)`	`{1, 2}`	`{"a":1}`
Empty	`[]`	`()`	`set()`	`{}`
Mutable	**yes**	**no**	**yes**	**yes**
Ordered	**yes**	**yes**	**no**	**yes***
Lookup	**O(n)**	**O(n)**	**O(1)**	**O(1) by key**

*dict keeps insertion order — guaranteed since 3.7.

WATCH OUT

`{}` is an empty dict, not a set. An empty set is spelled `set()` — there is no literal for it.

CHAPTER 03 · CONTINUED

Which container? Five seconds, four questions

- **Need key → value lookup?** Yes → `dict`. Just counting? Upgrade to `Counter`/`defaultdict`.
- **Only membership / uniqueness?** Yes → `set` — O(1) `in`, plus math ops `|` `&` `-` `^`.
- **Ordered sequence you'll edit?** Yes → `list` — append, sort, slice.
- **Fixed record / needs hashing?** Yes → `tuple`. Named fields? Reach for `@dataclass`.
- **Add/remove at both ends?** Yes → `collections.deque` — O(1) at both ends.

PRO INSIGHT

Converting a list to a set once, then testing membership in a loop, turns an $O(n^2)$ scan into $O(n)$ — the cheapest big-data speedup in Python.

THE DEDUPE IDIOM

`list(set(tags))` is unique but order is lost.
`list(dict.fromkeys(tags))` keeps first-seen order.

CHAPTER 04

Control flow & comprehensions

- **Nine falsy values, everything else is true:** `False` · `None` · `0` · `0.0` · `""` · `[]` · `{}` · `()` · `set()`. The idiom: `if items:` — never `if len(items) > 0:`.
- **Watch out:** `if x:` and `if x is not None:` differ when `0` or `""` are valid values — the first silently skips them.
- **for-else is real:** the `else` on a loop runs only if the loop finished without a `break` — classic interview bait.
- **match/case (3.10+):** structural pattern matching — it destructures shape, not just value. A bare name in a `case` is a capture, not a comparison.
- **Comprehensions, one line, three brackets:** `[x**2 for x in range(10)]` (list), `{k:v for...}` (dict), `(x**2 for x in huge)` (generator — lazy, memory-safe).
- **The one-readable-line rule:** if a comprehension no longer fits on one line you can read aloud, unroll it into a plain loop.

DEFINITION

A generator expression computes nothing up front — each value is produced on demand. A billion-item "sequence" costs almost no memory, as long as you consume it once.

CHAPTER 05

Functions & classes

- **No return means None — silently:** half of all `'NoneType' object has no attribute…` errors trace back to a function that forgot to return.
- **Defaults evaluate once, at def time:** a mutable default like `lst=[]` is shared state across every call — Python's most famous gotcha (full fix in Ch. 8).
- **lambda's one real job:** the `key=` argument — `people.sort(key=lambda p: p["age"])`. Never assign a lambda to a name.
- **A decorator, in one sentence:** a function that takes a function and returns a wrapped version. `@timer` above a `def` means `func = timer(func)`.
- **Generator vs list, the interview line:** a list holds every value in memory now; a generator computes each value on demand and forgets it.
- **@dataclass, the modern data-holder:** one decorator generates `__init__`, `__repr__`, `__eq__` for free — the default for records, configs, and API payloads.

THE HONEST GUIDANCE

Most automation never needs classes. Functions plus the built-in collections carry you remarkably far — reach for a class only when data and behavior travel together.

CHAPTER 06

Errors, files & modules

- **Catch specific exceptions:** a bare `except:` catches everything, including Ctrl-C and the bugs you needed to see. Widest acceptable net: `except Exception:`.
- **EAFP over LBYL:** Pythonic style tries the operation and catches the failure, instead of pre-checking every condition.
- **`with open(...)`, always:** auto-closes the file. Pass `encoding="utf-8"` on every text open — today's default is platform-dependent.
- **`if __name__ == "__main__":` decoded:** runs only when the file is executed directly, skipped when imported — makes one file both a library and a runnable tool.
- **The venv + pip workflow:** one venv per project ends both "pip doesn't work" and `ModuleNotFoundError` — the two biggest real-world beginner traps. Never `sudo pip install`.
- **uv, the 2026 sidebar:** ~75M monthly downloads, runs 10–100× faster than pip. Consensus: know pip, use uv for new projects.

READ A TRACEBACK BOTTOM-UP

The last line says **what** went wrong; the arrows above say **where** — ending at the line that actually failed. Since 3.13, tracebacks are colored and point at the exact token.

CHAPTER 07

Standard library greatest hits

- **datetime, in four objects:** `date`, `datetime`, `timedelta`, plus `strftime` (format out) / `strptime` (parse in). `utcnow()` is deprecated since 3.12 — use `now(timezone.utc)`.
- **pathlib over os.path:** join paths with a literal `/`: `Path("data") / "file.csv"`. Memorize `mkdir(parents=True, exist_ok=True)`.
- **json's s = string:** `loads/dumps` work on strings; `load/dump` work on open files. CSV files always open with `newline=""`.
- **Counter & defaultdict, interview gold:** `Counter(words).most_common(3)` is the expected answer to "find the top-k frequent elements."
- **Type hints, modern style:** write `list[int]` (3.9+) and `str | None` (3.10+) — checkers like mypy or Pylance enforce them, Python itself doesn't.

AFTER THE STDLIB, PICK ONE LANE

Data: pandas → matplotlib → scikit-learn. **Web:** FastAPI/Flask → SQLite → Django. **Automation:** pathlib + openpyxl → APIs with requests. **AI:** LLM SDKs → agent tooling → PyTorch. One lane, one project at a time — 6–9 months to job-ready.

CHAPTER 08 · THE CHAPTER THAT PAYS FOR THE BOOK

Idioms & gotchas — decoded

- **#1 — the mutable default argument:** `def add(item, lst=[]):` reuses the SAME list forever. Fix: default to `None`, create the list inside the function.
- **#2 — is vs ==:** == compares values; `is` asks "same object?" The rule: use `is` only for `None`, `True`, `False`.
- **#3 — assignment aliases, it never copies:** `b = a` is a second name for the same list. Need independence? `.copy()` (shallow) or `copy.deepcopy()` (full).
- **#4 — .sort() returns None:** `lst = lst.sort()` destroys your list. `sorted(lst)` returns a new list instead.

THE IDIOMS TABLE, IN ONE LINE EACH

`enumerate(xs)` not `range(len(xs))` · `zip(xs, ys)` not index-parallel loops · `a, b = b, a` not a temp variable · `d.get(k, 0)` not `if k in d`.

Errors decoded — the top real-world trap

ERROR	THE FIX
`NameError`	Typo, or a missing import — Python now suggests the fix
`TypeError: can only concatenate str`	Use an f-string instead of +
`KeyError`	`d.get(k, default)` to read safely
`IndexError`	Off-by-one — or just use `lst[-1]`
`AttributeError`	Print `type(x)`, walk back to the assignment

BONUS — THE TOP REAL-WORLD TRAP

`ModuleNotFoundError: No module named 'requests'` is not a bug in your code — the package isn't installed in the Python you're running. Fix: activate your venv, then `pip install`. The single most common beginner support question.

That's the 20% of Python you'll use 95% of the time. The full edition below covers all ten gotchas, both pages of errors decoded, and the complete appendix.

YOU'VE GOT THE MAP. NOW GET THE REFERENCE.

This was the highlight reel. The full guide is 50 visual pages.

You now look things up faster than most people who've used Python for years. The complete **Python in 50 Visual Pages** takes every concept above and gives you the full diagrams, decision trees, and code tables — verified against Python 3.13/3.14.

- **The full 8 chapters** — every concept above, illustrated and expanded to 50 pages.
- **All 4 containers, method by method** — the centerfold comparison, on facing pages.
- **All 10 gotchas, fully diagrammed** — mutable defaults, aliasing, the `[[0]*3]*3` grid trap.
- **The top 10 error messages decoded** — cause, meaning & fix, in plain English.
- **The one-page cheat card + 25 interview rapid-fire answers** plus the full glossary and version table.

UPGRADE TO THE FULL EDITION

~~₹199~~ **₹99**

50-page illustrated reference · instant download

SCAN / TAP
to upgrade

GO FURTHER, PAY LESS

Two ways to keep going

THE AI & TECH STACK BUNDLE

Python + AI Engineer Roadmap + Prompt Engineering + No-Code AI

Everything you need to go from zero to building with code and AI, in one download.

~~₹396~~ **₹199**

THE COMPLETE LIBRARY

All 30 Skill Series books

Spirituality, money, career, health, tech — the entire collection, one price.

₹499

GET EVERY NEW BOOK FREE

Follow **@weknowwhatyourelookingfor** and join our WhatsApp list — buyers get first access and reader-only pricing on everything we publish next.

weknowwhatyourelookingfor.com

NOTICE

This guide is for **general educational purposes only**. Individual results vary with effort and circumstances, and no specific outcome is warranted.

Published by weknowwhatyourelookingfor.com · For queries, corrections, or licensing, visit weknowwhatyourelookingfor.com.