PYTHON

MODULES

13 Modules • Full Course

os • NumPy • pandas • TensorFlow



LONG VIDEO • 49 MIN

PYTHON 3.13 • FULL COURSE

Python Modules Full Course

os, sys, pathlib, urllib, requests, NumPy, pandas, Matplotlib, TensorFlow and more

FREE FULL COURSE • MyWebUniversity.com



Wahidullah Lutfy
Founder, MyWebUniversity • CS Teacher

INTRODUCTION

What Is a Module, and How Do We Use One?

import
import os • import numpy as np • from pathlib import Path

Standard library
Comes with Python: os, sys, pathlib, urllib, html.parser. No install needed.

Third-party packages
Installed with pip: requests, NumPy, pandas, Matplotlib, fpdf2, TensorFlow, PyInstaller.

Use a virtual environment
python -m venv .venv, activate it, then pip install what each project needs.

ROADMAP

The 13 Modules in This Course

System
os • sys • pathlib

The web
urllib • requests • html.parser • cgi

Data science
NumPy • pandas • Matplotlib

Documents
fpdf2 (FPDF, the "Free PDF" library)

Artificial intelligence
TensorFlow and Keras

Shipping your app
PyInstaller: turn a script into an .exe

Python Modules • Full Course

MODULE 1 · os

MODULE 1 · os

The os Module

What it does

Talks to the operating system: folders, files, environment variables, processes.

Portable

The same code works on Windows, macOS and Linux.

Built in

Part of the standard library: just import os.

Use it for

Creating folders, listing files, renaming, deleting, reading settings.

MODULE 1 · os

os in Action

os_demo.py — Python 3.13

```
>>> import os
>>> os.getcwd()
'C:\\Users\\wahid\\projects'
>>> os.listdir(".")
['notes.txt', 'photos', 'app.py']

>>> os.makedirs("reports/2026", exist_ok=True)
>>> os.rename("notes.txt", "reports/notes.txt")

>>> os.environ.get("USERNAME")
'wahid'
>>> os.cpu_count()
8
```

Where am I?

os.getcwd() is the current folder; os.listdir() lists it.

Safe folder creation

makedirs(..., exist_ok=True) makes nested folders and never fails if they exist.

Environment

os.environ holds settings like USERNAME and PATH. Great for secrets and config.

MODULE 1 · os

Walking a Folder Tree

tree.py — Python 3.13

```
import os
from collections import Counter
kinds = Counter()
for folder, subfolders, files in os.walk("photos"):
   for name in files:
       kinds[os.path.splitext(name)[1].lower()] += 1
print(kinds.most_common(3))
[('.jpg', 812), ('.png', 95), ('.heic', 40)]

os.remove("old_backup.zip")      # deletes immediately, no Recycle Bin
```

os.walk

Visits every folder and subfolder, giving you each folder name and its files.

Careful with deleting

os.remove and os.rmdir skip the Recycle Bin. Test with print() first.

Recap & Practice

Remember

getcwd, listdir, makedirs, rename, environ, walk.

Try it yourself

Count the file types in your Downloads folder with os.walk.

Watch out

os.remove skips the Recycle Bin. Print before you delete.

Python Modules • Full Course

MODULE 2 · sys

MODULE 2 · sys

The sys Module

greet.py — Python 3.13

```
>>> import sys
>>> sys.version
'3.13.1 (main, Dec  3 2024) [MSC v.1942 64 bit (AMD64)]'
>>> sys.platform
'win32'


import sys
if len(sys.argv) < 2:
   print("Usage: python greet.py NAME")
   sys.exit(1)
print(f"Hello, {sys.argv[1]}!")

$ python greet.py Wahid
Hello, Wahid!
```

sys vs os

os talks to the operating system; sys talks to Python itself.

sys.argv

The command-line arguments. argv[0] is the script name.

sys.exit(code)

Stops the program. 0 means success; any other number means an error.

More Useful sys Tools

sys.path
The list of folders Python searches when you import a module.

sys.executable
The exact python.exe running your code. Handy for checking your virtual environment.

sys.stdin / stdout / stderr
Standard input, output and error streams.

sys.getsizeof(obj)
How many bytes an object uses in memory.

Recap & Practice

Remember

version, platform, argv, exit, path, executable.

Try it yourself

Write add.py that adds two numbers given on the command line.

Watch out

argv items are text: convert with int() or float().

Python Modules • Full Course

MODULE 3 · pathlib (paths)

MODULE 3 · pathlib

pathlib: Paths as Objects

paths.py — Python 3.13

```
>>> from pathlib import Path
>>> report = Path("data") / "2026" / "report.txt"
>>> report
WindowsPath('data/2026/report.txt')

>>> report.parent.mkdir(parents=True, exist_ok=True)
>>> report.write_text("Sales: 1200")
>>> report.read_text()
'Sales: 1200'

>>> report.name, report.stem, report.suffix
('report.txt', 'report', '.txt')
>>> [p.name for p in Path(".").glob("*.py")]
['app.py', 'paths.py']
```

Join with /

Path("data") / "2026" builds a correct path on every operating system.

Read and write

read_text() and write_text() handle opening and closing for you.

os.path vs pathlib

os.path.join() is the older style. pathlib is cleaner for new code.

Real Example: Organize Your Downloads

1 · Find files

downloads = Path.home() / "Downloads" • for f in downloads.iterdir()

2 · Choose a folder

Use f.suffix: .pdf → Documents, .jpg → Pictures, .zip → Archives.

3 · Move safely

target.mkdir(exist_ok=True) • f.rename(target / f.name)

4 · Run it weekly

Schedule it, and your Downloads folder stays clean forever.

Recap & Practice

Remember

Path(), the / operator, read_text, write_text, glob, suffix, stem.

Try it yourself

List every .py file in a folder and its subfolders with rglob("*.py").

Watch out

Relative paths depend on the current folder. Use Path(__file__).parent for files next to your script.

Python Modules • Full Course

MODULE 4 · urllib

MODULE 4 · urllib

The urllib Package

urllib.request
Open URLs and download web pages and files.

urllib.parse
Split URLs into parts, and build query strings safely.

urllib.error
HTTPError and URLError, for when things go wrong.

Why learn it?
Built in, no install. Works anywhere Python runs.

MODULE 4 · urllib

urllib in Action

web_basics.py — Python 3.13

```
>>> from urllib.request import urlopen
>>> with urlopen("https://www.python.org") as resp:
...     html = resp.read().decode("utf-8")
>>> resp.status, len(html)
(200, 50112)


>>> from urllib.parse import urlparse, urlencode
>>> urlparse("https://example.com/search?q=python").query
'q=python'
>>> urlencode({"q": "python modules", "page": 2})
'q=python+modules&page=2'
```

with urlopen(...)

Opens the page and closes the connection automatically.

Bytes vs text

read() returns bytes; decode("utf-8") turns them into a string.

urlencode

Builds a correct query string from a dictionary.

MODULE 4 · urllib

Handling Errors and Headers

safe_fetch.py — Python 3.13

```
from urllib.request import Request, urlopen
from urllib.error import HTTPError, URLError
req = Request("https://www.python.org/nope", headers={"User-Agent":
"MyWebUniversity-bot/1.0"})

try:
   with urlopen(req, timeout=10) as resp:
       print(resp.status)
except HTTPError as e:
   print("Server said:", e.code)
except URLError as e:
   print("Network problem:", e.reason)
Server said: 404
```

HTTPError vs URLError

HTTPError: the server answered with an error code. URLError: we could not reach it at all.

Headers and timeout

Identify your program politely, and never wait forever.

Recap & Practice

Remember

urlopen, Request with headers, urlparse, urlencode, HTTPError and URLError.

Try it yourself

Download python.org and count how many times the word "Python" appears.

Watch out

read() returns bytes. Decode with the right encoding, usually utf-8.

Python Modules • Full Course

MODULE 5 · requests

MODULE 5 · requests

requests: HTTP for Humans

api_demo.py — Python 3.13

```
$ pip install requests

>>> import requests
>>> r = requests.get("https://api.github.com/repos/python/cpython",
timeout=10)
>>> r.status_code
200
>>> r.json()["stargazers_count"]
67000

>>> r = requests.post("https://httpbin.org/post", json={"name": "Wahid"},
timeout=10)
>>> r.raise_for_status()
>>> r.json()["json"]
{'name': 'Wahid'}
```

Simple and readable

get, post, put, delete, with headers, params and JSON built in.

.json()

Turns the API response straight into Python dictionaries and lists.

Always set a timeout

And call raise_for_status() so errors do not pass silently.

requests: Professional Habits

params=

requests.get(url, params={"q": "python"}) builds the query string for you.

headers= and auth

Send API keys in headers, read from os.environ, never written in code.

Session()

Reuses the connection and settings for many requests. Faster and cleaner.

Respect the API

Read its rate limits and terms, and handle errors with try/except.

Recap & Practice

Remember

get, post, params, headers, json(), timeout, raise_for_status, Session.

Try it yourself

Call a free public API, like api.github.com, and print three fields from the JSON.

Watch out

Never put API keys in your code. Read them from environment variables.

Python Modules • Full Course

MODULE 6 · html.parser

MODULE 6 · html.parser

Reading HTML with HTMLParser

links.py — Python 3.13

```
from html.parser import HTMLParser

class LinkFinder(HTMLParser):
   def __init__(self):
       super().__init__()
       self.links = []
   def handle_starttag(self, tag, attrs):
       if tag == "a":
           self.links += [v for k, v in attrs if k == "href"]


finder = LinkFinder()
finder.feed('<p>Visit <a href="https://mywebuniversity.com">us</a></p>')
print(finder.links)
['https://mywebuniversity.com']
```

Event-based
The parser calls your methods as it meets each start tag, end tag and piece of text.

handle_starttag / handle_data
Override only the events you care about.

Real projects
Built in and dependency-free. For heavy scraping, BeautifulSoup is friendlier.

MODULE 6 · html.parser

urllib + html.parser Together

titles.py — Python 3.13

```
from html.parser import HTMLParser
from urllib.request import urlopen

class TitleFinder(HTMLParser):
   in_title, title = False, ""
   def handle_starttag(self, tag, attrs): self.in_title = tag == "title"
   def handle_endtag(self, tag): self.in_title = False
   def handle_data(self, data):
       if self.in_title: self.title += data



p = TitleFinder()
p.feed(urlopen("https://www.python.org").read().decode())
print(p.title)
Welcome to Python.org
```

Keep state

A flag like in_title remembers where the parser is right now.

handle_data

Receives the text between tags; collect it only when the flag is on.

Recap & Practice

Remember

Subclass HTMLParser; handle_starttag, handle_endtag, handle_data; feed().

Try it yourself

Print every image address (the src of <img> tags) on a web page.

Watch out

Real HTML is messy. Test on several pages, and respect each site's terms.

Python Modules • Full Course

MODULE 7 · cgi

cgi: A Piece of Web History

What CGI was

Common Gateway Interface: the web server ran your script once for every request.

cgi.FieldStorage

Read form fields sent by the browser, e.g. form.getvalue("name").

Removed in Python 3.13

Deprecated in 3.11 and removed in 3.13 (PEP 594). It is slow and dated.

What to use now

urllib.parse.parse_qs for query strings; Flask or FastAPI for web apps.

The Modern Replacements

modern_web.py — Python 3.13

```
>>> from urllib.parse import parse_qs
>>> parse_qs("name=Wahid&course=python")
{'name': ['Wahid'], 'course': ['python']}

$ pip install flask
from flask import Flask, request
app = Flask(__name__)
@app.route("/hello")
def hello():
   return f"Hello, {request.args.get('name', 'student')}!"

$ flask --app modern_web run
 * Running on http://127.0.0.1:5000
```

parse_qs

Turns a query string into a dictionary. Values are lists, because a key can repeat.

Flask

A small web framework: one function per page, and it stays running between requests.

Old cgi scripts?

The community package legacy-cgi can keep old code running while you migrate.

Recap & Practice

Remember

cgi was removed in Python 3.13. Use parse_qs for query strings, and Flask or FastAPI for web apps.

Try it yourself

Extend the Flask app: add a /square page that returns a number squared.

Watch out

Check whether a module is deprecated before you depend on it.

Python Modules • Full Course

MODULE 8 · NumPy

MODULE 8 · NumPy

NumPy: Fast Numerical Arrays

numbers.py — Python 3.13

```
$ pip install numpy
>>> import numpy as np
>>> prices = np.array([10.0, 12.5, 9.0, 15.0])
>>> prices * 1.08
array([10.8 , 13.5 ,  9.72, 16.2 ])


>>> prices.mean(), prices.max(), prices.std().round(2)
(11.625, 15.0, 2.33)
>>> prices[prices > 10]
array([12.5, 15. ])

>>> grid = np.arange(12).reshape(3, 4)
>>> grid.shape, grid.sum(axis=0)
((3, 4), array([12, 15, 18, 21]))
```

Vectorized math

One operation on the whole array, no loop. Often 10 to 100 times faster than lists.

Boolean filtering

prices[prices > 10] keeps only the values that match.

Shapes and axes

Arrays can be 2D, 3D and more. axis=0 works down the columns.

MODULE 8 · NumPy

Why NumPy Is So Fast

speed.py — Python 3.13

```
>>> import numpy as np, time
>>> nums = list(range(10_000_000))
>>> t = time.time(); total = sum(x * x for x in nums); time.time() - t
0.61



>>> arr = np.arange(10_000_000)
>>> t = time.time(); total = (arr * arr).sum(); time.time() - t
0.02


>>> np.random.default_rng(42).normal(70, 10, 5).round(1)
array([73. , 59.6, 77.5, 79.4, 50.5])
```

About 30 times faster

The same calculation: a Python loop vs one NumPy expression. Your timings will vary.

Random numbers

np.random.default_rng(seed) makes repeatable random data for testing.

Recap & Practice

Remember

np.array, vectorized math, mean/max/std, boolean filtering, reshape, axis.

Try it yourself

Create 1,000 random test scores and count how many are above 90.

Watch out

Arrays hold one type. Mixing text and numbers turns everything into text.

Python Modules • Full Course

MODULE 9 · pandas

MODULE 9 · pandas

pandas: Tables in Python

sales.py — Python 3.13

```
$ pip install pandas openpyxl
>>> import pandas as pd
>>> df = pd.read_csv("sales.csv")
>>> df.head(3)
    city  month  sales
0   Kabul    Jan   1200
1  Fresno    Jan    950

>>> df[df["sales"] > 1000]
>>> df.groupby("city")["sales"].sum().sort_values(ascending=False)

>>> df.describe()
>>> df.to_excel("summary.xlsx", index=False)
```

DataFrame

A table with named columns, like an Excel sheet inside Python.

Filter and group

The same boolean filtering as NumPy, plus groupby for totals by category.

Read and write anything

CSV, Excel, JSON, SQL databases and more.

MODULE 9 · pandas

Cleaning Real-World Data

Find missing values
df.isna().sum() counts the empty cells in every column.

Fix or drop them
df["sales"].fillna(0) • df.dropna(subset=["city"])

Fix types
pd.to_datetime(df["date"]) • df["sales"].astype(float)

Remove duplicates
df.drop_duplicates() keeps your totals honest.

Recap & Practice

Remember

read_csv, head, describe, filtering, groupby, sort_values, to_excel, data cleaning.

Try it yourself

Load any CSV (try a public dataset) and find the top 5 rows by one column.

Watch out

Look at your data with head() and isna() before trusting any total.

Python Modules • Full Course

MODULE 10 · Matplotlib (pyplot)

MODULE 10 · Matplotlib

Charts with matplotlib.pyplot

chart.py — Python 3.13

```
$ pip install matplotlib
import matplotlib.pyplot as plt
months = ["Jan", "Feb", "Mar", "Apr"]
sales  = [950, 1200, 1100, 1500]

fig, ax = plt.subplots(figsize=(8, 4))
ax.plot(months, sales, marker="o", label="Sales")
ax.set_title("Monthly Sales")
ax.set_ylabel("USD")
ax.legend()

fig.savefig("sales.png", dpi=150)
plt.show()
```

Figure and axes

fig is the whole picture; ax is one chart inside it.

Many chart types

plot (lines), bar, scatter, hist, pie, and more.

savefig

Save as PNG or PDF for reports and websites; show() opens a window.

More Chart Types, Side by Side

dashboard.py — Python 3.13

```
import matplotlib.pyplot as plt, numpy as np
fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(12, 4))

ax1.bar(["Kabul", "Fresno", "Paris"], [1200, 950, 1100])
ax2.scatter(np.arange(20), np.random.rand(20) * 100)
ax3.hist(np.random.default_rng(1).normal(70, 10, 500), bins=20)

ax1.set_title("Sales by city"); ax2.set_title("Scatter");
ax3.set_title("Scores")
fig.tight_layout(); fig.savefig("dashboard.png")
```

Pick the right chart

Bar: compare categories. Line: change over time. Scatter: relationships. Histogram: distribution.

tight_layout()

Stops titles and labels from overlapping.

Recap & Practice

Remember

fig, ax = plt.subplots(); plot, bar, scatter, hist; titles, labels, legend; savefig.

Try it yourself

Chart your pandas results: df.plot(kind="bar") uses Matplotlib behind the scenes.

Watch out

Always label your axes and units. A chart without labels can mislead.

Python Modules • Full Course

MODULE 11 · fpdf2 (FPDF, Free PDF)

MODULE 11 · fpdf2

Create PDF Reports with fpdf2

report.py — Python 3.13

```
$ pip install fpdf2
from fpdf import FPDF
pdf = FPDF()
pdf.add_page()
pdf.set_font("Helvetica", "B", 20)
pdf.cell(0, 12, "Monthly Sales Report", new_x="LMARGIN", new_y="NEXT")



pdf.set_font("Helvetica", size=12)
pdf.multi_cell(0, 8, "Sales grew 58% from January to April.")
pdf.image("sales.png", w=170)

pdf.output("report.pdf")
```

FPDF = Free PDF

Install fpdf2, the maintained version; import it with from fpdf import FPDF.

Pages, fonts, cells

add_page, set_font, cell for one line, multi_cell for paragraphs.

Automate reports

Combine pandas + Matplotlib + fpdf2 to build a full report automatically.

fpdf2: Tips for Great Reports

Tables
with pdf.table() as table: add rows from your pandas data.

Headers and footers
Subclass FPDF and override header() and footer(), e.g. page numbers.

Unicode fonts
pdf.add_font("DejaVu", fname="DejaVuSans.ttf") for Dari, Arabic and other scripts.

Links and metadata
pdf.cell(..., link="https://mywebuniversity.com") • pdf.set_title(...)

Recap & Practice

Remember

pip install fpdf2, from fpdf import FPDF; add_page, set_font, cell, multi_cell, image, output.

Try it yourself

Build a one-page report: title, a paragraph, your Matplotlib chart and a table.

Watch out

Built-in fonts are Latin only. Add a Unicode font for other languages.

Python Modules • Full Course

MODULE 12 · TensorFlow

What Is TensorFlow?

A machine learning library
Created by Google. Builds and trains neural networks.

Tensors
Multi-dimensional arrays, like NumPy arrays, that can run on GPUs.

Keras
TensorFlow's friendly high-level API: build a model layer by layer.

Install
pip install tensorflow. Check the supported Python versions; for GPU on Windows, use WSL2.

MODULE 12 · TensorFlow

Your First Neural Network

first_model.py — Python 3.13

```
import numpy as np, tensorflow as tf
xs = np.array([-1, 0, 1, 2, 3, 4], dtype=float)
ys = np.array([-3, -1, 1, 3, 5, 7], dtype=float)   # y = 2x - 1

model = tf.keras.Sequential([tf.keras.Input(shape=(1,)),
tf.keras.layers.Dense(1)])
model.compile(optimizer="sgd", loss="mean_squared_error")
model.fit(xs, ys, epochs=500, verbose=0)

print(model.predict(np.array([[10.0]])))
[[18.98]]
```

Learn from examples

We never write the rule y = 2x - 1. The model discovers it from the data.

compile and fit

The optimizer adjusts the weights; the loss measures how wrong the guesses are.

Close, not exact

18.98, not 19: models learn probabilities and approximations.

Recap & Practice

Remember

Tensors, Keras Sequential, Dense layers, compile, fit, predict.

Try it yourself

Change the data to y = 3x + 2 and retrain. Does the model find the new rule?

Watch out

More epochs is not always better. Watch the loss, and test on data the model has not seen.

Python Modules • Full Course

MODULE 13 · PyInstaller

MODULE 13 · PyInstaller

Turn a Script into an App

Windows PowerShell

```
PS C:\Users\wahid> pip install pyinstaller
PS C:\Users\wahid> pyinstaller --onefile --name SalesReport report.py
... Building EXE from EXE-00.toc completed successfully.


PS C:\Users\wahid> dir dist
SalesReport.exe

PS C:\Users\wahid> pyinstaller --onefile --windowed --icon app.ico --add-data "sales.csv;." app.py
```

Why PyInstaller?

Your users do not need Python: Python and every module are bundled inside.

Key options

--onefile, --windowed (no console), --name, --icon, --add-data for extra files.

Build per OS

Build on Windows for .exe, on macOS for Mac, on Linux for Linux.

Recap & Practice

Remember

--onefile, --windowed, --name, --icon, --add-data; output in the dist folder.

Try it yourself

Package your fpdf2 report script into one .exe and run it on another PC.

Watch out

Build inside a clean virtual environment, so only the modules you need are bundled.

WRAP-UP

Course Recap & Next Steps

System
os, sys and pathlib: control files, folders and your program.

Web
urllib, requests and html.parser; Flask instead of cgi.

Data
NumPy, pandas and Matplotlib: analyze and visualize.

Reports
fpdf2 turns your results into PDF documents.

AI
TensorFlow: models that learn from examples.

Share
PyInstaller: one .exe anyone can run.