NULLVECTOR
.monster

PHASE 01 — FOUNDATIONS

DAYS 1–21 · LEARN TO CODE · YOUR CHOICE OF TRACK · PYTHON FLUENCY

Before you can build AI, you need to speak its language. Phase 1 gets you there — at your own pace, on your own terms, using whichever learning style actually works for you.

By Day 21 you will be comfortable writing Python, reading error messages without panic, pushing code to GitHub every day, and thinking like someone who builds things with code. You do not need to be an expert. You need to be ready for Phase 2. These are very different things, and 21 days is enough.

IF YOU'RE SCARED, YOU'RE IN THE RIGHT PLACE.

→ **You don't need to understand everything on Day 1.**

Seriously. You will not understand everything on Day 1. That is expected and completely fine. Understanding comes from doing. Do the task. The understanding follows. Every professional programmer still Googles basic things every single day.

→ **Error messages are not failure. They are progress.**

Every error message means your code ran far enough to find a problem. Professional engineers see dozens of error messages every day. When something breaks, you are doing exactly what you are supposed to do. Read the error. Google it. Ask Claude to explain it. Fix it. Move on.

→ **You are allowed to use AI to learn — just not to skip.**

Claude or ChatGPT are extraordinary tutors. Ask them to explain something a different way. Ask them to use an analogy. Ask them why your code broke. The one thing not to do: ask them to write your code for you. The skills need to live in your hands, not the AI's.

→ **Slow is fine. The only failure is stopping.**

If a day takes 4 hours instead of 3, that's fine. If you reread something five times, that's fine. If you miss a day, come back the next day. The habit of returning matters more than the habit of perfection.

→ **Tell one person you're doing this.**

Post your Day 0 log publicly on GitHub. Text one friend. The moment you make it visible to even one other person, you shift from 'thinking about doing this' to 'being someone who is doing this.' That shift is more powerful than any curriculum.

BEFORE DAY 1 — DO THESE 5 THINGS NOW

1. Go to github.com and create a free account.

GitHub is where your code lives. Every day you will push your work here. It becomes your public record — proof that you showed up.

2. Create a public repository called nullvector-journey.

This is your home base for the entire 150-day challenge. Everything you build goes here.

3. Create a file in it called log.md.

This is your daily journal. Three sentences per day. What you built. What confused you. What you'll do tomorrow.

4. Write your first entry right now.

Copy this exactly: 'Day 0. I am starting the NULLVECTOR challenge. I have never coded before. I am a little scared. I am doing it anyway.'

5. Pick your track on the next page and start today.

Not tomorrow. Not after you finish setting things up. Today. Even 20 minutes of Day 1 is Day 1.

The curriculum starts on the next page. Take a breath. You've got this.

CHOOSE YOUR TRACK

All three tracks arrive at the same destination by Day 21. Pick the one that matches how you actually learn — not the one that sounds most impressive. There is no wrong choice. There is only the choice you will actually stick with.

TRACK A — THE WATCHER // LEARN BY WATCHING SOMEONE DO IT FIRST

You absorb things by watching someone explain and demonstrate. You pause videos and type along. You like seeing it done before you try it yourself. Videos feel natural to you.

Week 1 Days 1–7	**Mosh Hamedani — Python for Beginners** YouTube · Free · 6 hours youtube.com → search 'Programming with Mosh Python' Watch in sessions. Pause after every concept. Type every example before continuing. Do not just watch — your hands must be moving.
Week 2 Days 8–14	**Corey Schafer — Python Tutorial Series** YouTube · Free · Watch episodes 1–12 youtube.com → search 'Corey Schafer Python Tutorial' Goes deeper than Mosh — functions, OOP, error handling, file I/O. These are the Python patterns that appear in every AI codebase.
Week 3 Days 15–21	**Kaggle Learn — Python + Intro to Programming** kaggle.com/learn · Free · Browser-based Complete both free courses in order. Kaggle is the platform where real data scientists work. This week bridges general Python into data and AI context specifically.

TRACK B — THE BUILDER // LEARN BY DOING IMMEDIATELY ★ RECOMMENDED

You get bored watching. You want to type things and see them work — or break. You learn by doing, not observing. You prefer immediate feedback over explanation. This is the recommended track for most beginners because everything is free, browser-based, and interactive from minute one.

Week 1 Days 1–7	**freeCodeCamp — Scientific Computing with Python** freecodecamp.org · Free · No install needed Go to freecodecamp.org, find the Python certification, start from the beginning. Browser-based — you write and run code directly in the website. No downloading anything. No setup. Just open and start.
Week 2 Days 8–14	**Automate the Boring Stuff with Python — Al Sweigart** automatetheboringstuff.com · Free online · Read Chapters 1–8 The most practical Python book that exists. Written for people who want to solve real problems, not pass exams. Every chapter ends with a project you actually build.
Week 3 Days 15–21	**Kaggle Learn — Python + Intro to Programming** kaggle.com/learn · Free · Browser-based Complete both free courses in order. Kaggle is the platform where real data scientists work. This week bridges general Python into data and AI context specifically.

TRACK C — THE READER // LEARN FROM STRUCTURED WRITTEN MATERIAL

You learn from books. You like being able to re-read. You highlight things. Videos feel too fast or too slow. You want to understand something properly before moving on.

Week 1 Days 1–7	**Python Crash Course — Eric Matthes (book, ~$30)** Chapters 1–8: variables, lists, functions, loops, dictionaries, user input, files. The best beginner Python book available. Clear, project-based, well-paced. Buy the physical book or the ebook — either works. Read with your laptop open. Type every code example.
Week 2 Days 8–14	**Python Crash Course — Eric Matthes (Chapters 9–12)** Classes, files, exceptions, and the first project: a simple game. By end of Week 2 you will have built something that actually runs and does something. That matters. It shifts how you see yourself.
Week 3 Days 15–21	**Kaggle Learn — Python + Intro to Programming** kaggle.com/learn · Free · Browser-based Complete both free courses in order. Kaggle is the platform where real data scientists work. This week bridges general Python into data and AI context specifically.

TRACK D — THE ACADEMIC // FULL CREDENTIAL PATH VIA HARVARD CS50X

You want academic rigor and a recognized credential. You are motivated by structured problem sets and grading. You don't mind a harder on-ramp because you want the complete foundation. Honest warning: CS50x normally takes 12+ weeks. Completing it in 21 days requires 4–5 hours daily and is genuinely demanding. If you are scared of AI, start with Track B instead and come back to CS50x later.

Week 1 Days 1–7	**CS50x — Weeks 0, 1, 2, 3 (Harvard via edX)** cs50.harvard.edu/x · Free to audit · $149 for certificate Scratch → C → arrays → algorithms. Yes, you start with C, not Python. This is intentional — it teaches you how computers actually work, which makes everything else click faster later.
Week 2 Days 8–14	**CS50x — Weeks 4, 5, 6 (Memory, Data Structures, Python)** Memory management, pointers, hash tables, then Python. Week 6 is where Python appears and everything you learned in C suddenly becomes much cleaner and easier. This is the payoff.
Week 3 Days 15–21	**CS50x — Weeks 7–9 + Kaggle Bridge** SQL, HTML/CSS, Flask. Then Kaggle Learn Python to bridge into AI context. Skip the CS50x final project for now — you will build real projects in Phase 2. Focus on getting through the material and into the Kaggle bridge.

THE DAILY STRUCTURE

This is the same every day regardless of which track you chose. The structure is the habit. The habit is the whole thing.

■ 15 MIN	INDUSTRY	Watch or listen to something about the AI world. Not educational — current. A podcast, a YouTube clip, a news story. You are learning a language and you need to hear it spoken every day.
■ 90 MIN	STUDY + CODE	Your track resource for today. Always have a code editor open alongside whatever you're learning. Type every example. Change things. Break things. Fix things.
■ 60 MIN	BUILD	Write something original — not from a tutorial. Even something tiny. A calculator. A quiz. A word counter. Something that did not exist before today and now does because you made it.
■ 15 MIN	LOG	Three sentences in log.md on GitHub. Push to GitHub. The green square is proof you showed up. 21 green squares in a row by Day 21 is your Phase 1 milestone.

WHAT YOU BUILD — SAME FOR ALL TRACKS

WEEK 1 · Days 1–7

Something that takes input and gives output.

A program that asks your name and says hello. A calculator. A temperature converter. A number guessing game. It does not matter what it is. It matters that you wrote it from scratch without following a tutorial line by line. Commit it to GitHub. Write three sentences about it in your log.

WEEK 2 · Days 8–14

Something that stores and uses data.

A simple contact list. A quiz that keeps score. A word frequency counter. A program that reads a file and does something with what's in it. This week's build uses lists, dictionaries, or files — the data structures that appear in every real program ever written.

WEEK 3 · Days 15–21

Something you actually want to exist.

Pick something from your own life that would be useful. A tool that tracks something you care about. A game you'd actually play. A script that automates something you do manually. This is the build you'll show someone when they ask what you've been doing. Make it something you're genuinely proud of.

INDUSTRY IMMERSION — 15 MIN DAILY

Every day before you open your learning resource, spend 15 minutes listening to the world you are entering. You don't need to understand everything you hear. You need to get used to the vocabulary, the names, the problems people are working on. This is how you learn any language — immersion.

YouTube — beginner friendly

→ 3Blue1Brown: How Large Language Models Work → youtube.com/watch?v=wjZofJX0v4M

→ Andrej Karpathy: Intro to Large Language Models → youtube.com/watch?v=zjkBMFhNj_g

→ AI Explained channel → search 'AI Explained' on YouTube

→ Fireship: Python in 100 Seconds → youtube.com/watch?v=x7X9w_GIm1s

Podcasts — listen while commuting or walking

→ Hard Fork (NYT) → nytimes.com/column/hard-fork

→ Lex Fridman Podcast → lexfridman.com/podcast (pick any AI researcher episode)

→ TWIML AI Podcast → twimlai.com/podcast

→ Latent Space → latent.space (more technical, good to grow into)

Short reads — 5 minutes or less

→ The Rundown AI newsletter → therundown.ai

→ Import AI by Jack Clark → jack-clark.net

→ MIT Technology Review AI section → technologyreview.com

WEEK 1 · DAYS 1–7 · FIRST CONTACT

The goal of Week 1 is simple: get comfortable. Comfortable with your tools. Comfortable with error messages. Comfortable with the feeling of not knowing something and figuring it out anyway. By Day 7 you will have written real Python programs from scratch.

DAY 001 · INITIALIZE

Your first program exists in the world today.

■ INDUSTRY (15 MIN)

Watch: How Large Language Models Work — 3Blue1Brown (12 min)

Don't try to understand it. Just get used to hearing this language.

→ youtube.com/watch?v=wjZofJX0v4M

■ STUDY (90 MIN) — Start your track resource

Track A: Open YouTube, search 'Programming with Mosh Python', start the video.

Track B: Go to freecodecamp.org, find the Python certification, begin.

Track C: Open Python Crash Course, read Chapter 1, type every example.

Track D: Go to cs50.harvard.edu/x, watch Week 0 lecture from the beginning.

Install Python: python.org/downloads (version 3.11 or higher)

Install VS Code: code.visualstudio.com/download

Install the Python extension inside VS Code (search 'Python' in Extensions)

■ BUILD (60 MIN) — Your first program

Open VS Code. Create a folder called 'nullvector' on your desktop.

Create a file called day001.py inside it.

Type this — do not copy paste:

```
name = input('What is your name? ')
print('Hello ' + name + '. You just wrote your first program.')
print('Day 1 of 150. The journey starts now.')
```

Press Run. See it work. Then change something. See what breaks.

Create a GitHub account: github.com

Create a repo called nullvector-journey. Push your day001.py.

■ LOG (15 MIN)

Open log.md in your GitHub repo. Write exactly three sentences:

What you did today. What confused you. What you will do tomorrow.

This is not optional. The log is the record. Push it to GitHub.

✓ DONE WHEN: Python installed. VS Code open. First program runs. GitHub repo live. Day 1 log written.

DAY 002 · VARIABLES

Data has types. That matters in AI.

■ INDUSTRY (15 MIN)

Watch: Python in 100 Seconds — Fireship

→ youtube.com/watch?v=x7X9w_GIm1s

■ STUDY (90 MIN) — Continue your track

Focus today on: variables, data types (string, integer, float, boolean), print.

Every type matters. AI models are just math on numbers. Strings become numbers.

Type every example you see. Don't copy paste — your hands need to learn this.

■ BUILD (60 MIN) — Personal info calculator

From scratch — no tutorial:

Ask the user their name, age, and city.

Calculate what year they were born.

Print: 'Hello [name] from [city]. You were born in [year].'

Print: 'In 10 years you will be [age+10].'

When it works: add one more calculation of your own choosing.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Program runs, takes input, calculates and prints results. Day 2 log committed.

DAY 003 · LOGIC

Computers make decisions. Now yours can too.

■ INDUSTRY (15 MIN)

Search on YouTube: 'what is artificial intelligence explained simply'

Pick any result under 10 minutes. Watch it. Note words you don't know.

■ STUDY (90 MIN) — if / elif / else

Focus today on: if statements, elif, else, comparison operators, and/or/not.

Key insight: this is how every AI makes decisions.

A spam filter is just thousands of if statements learned from data.

■ BUILD (60 MIN) — Quiz game

Build a quiz about something you know — sports, movies, history, anything.

5 questions. Check each answer with if/else. Keep a score.

At the end: different message for 5/5, 3-4/5, and below 3/5.

This is your first program with real logic in it.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Quiz runs, scores correctly, gives different messages for different scores.

DAY 004 · LOOPS

Computers repeat without getting bored. Use that.

■ INDUSTRY (15 MIN)

Search: 'how does ChatGPT actually work simple explanation'

Note how much of it you understand compared to Day 1. It will be more.

■ STUDY (90 MIN) — for loops, while loops

Focus: for loops, while loops, range(), break, continue.

Key question to ask yourself at every loop: what happens on iteration 1? The last?

■ BUILD (60 MIN) — Number guessing game

The computer picks a random number between 1 and 100.

import random

number = random.randint(1, 100)

The user guesses. You say 'too high', 'too low', or 'correct'.

Keep looping until they get it. Count and display their guesses at the end.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Guessing game loops correctly until the right answer. Guess count displayed.

DAY 005 · FUNCTIONS

Stop repeating yourself. Wrap things that repeat.

■ INDUSTRY (15 MIN)

Search: 'what do AI engineers actually do all day'

Pick any video from a working AI engineer. This is where you're headed.

■ STUDY (90 MIN) — def, parameters, return

Focus: defining functions with def, parameters, return values, scope.

Key insight: a function is a reusable machine. Input goes in. Output comes out.

Everything in AI is functions calling other functions. This is fundamental.

■ BUILD (60 MIN) — Refactor your quiz using functions

Take your Day 3 quiz and rebuild it using functions:

def ask_question(question, correct_answer) — returns True or False

def run_quiz() — calls ask_question 5 times, tracks score

def show_result(score) — returns the appropriate message

def main() — runs everything

Same quiz. Cleaner code. This is how professionals write programs.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Quiz rebuilt with functions. Each function does one thing. Code is cleaner than Day 3.

DAY 006 · LISTS + DATA

Real programs handle collections of things.

■ INDUSTRY (15 MIN)

Hard Fork podcast — most recent episode, first 15 minutes.

nytimes.com/column/hard-fork — this is how mainstream media covers AI.

■ STUDY (90 MIN) — lists, dictionaries

Focus: lists, list methods (append, remove, sort, len), indexing, dictionaries.

Key insight: almost all data in AI is stored in lists and dictionaries.

A dataset is a list of dictionaries. A model's weights are a list of numbers.

■ BUILD (60 MIN) — Study tracker

Build a program that tracks your NULLVECTOR progress:

A list called completed_days that starts empty.

A function add_day(day_num, notes) that adds to the list.

A function show_progress() that prints all logged days and % complete out of 150.

Run it with at least 6 fake entries. See your own progress printed back at you.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Study tracker runs, logs days, calculates and displays completion percentage.

DAY 007 · WEEK 1 COMPLETE

Review, consolidate, and build something real.

■ INDUSTRY (15 MIN)

Watch: Andrej Karpathy — Intro to Large Language Models (first 15 min)

youtube.com/watch?v=zjkBMFhNj_g — by the person who built Tesla Autopilot's AI.

■ SELF TEST (45 MIN)

Without looking at any code: write the structure of a Python function on paper.

Write a for loop. Write an if/elif/else. Write a dictionary.

If you can do it without looking, you understand it.

If you cannot, spend 30 minutes on whichever is weakest before building.

■ BUILD (2 HRS) — Week 1 Project: Your Own Idea

Build something you actually want to exist. No tutorial. No example to follow.

Ideas if you're stuck:

A movie/book recommendation tool that asks your mood and suggests something

A simple budget tracker that logs income and expenses and shows balance

A workout logger that tracks what you did and shows weekly totals

A word game — anagrams, rhymes, scrambles

It must use: variables, at least one loop, at least one function,

at least one list or dictionary, and at least one if statement.

This is your Week 1 capstone. It goes in your GitHub portfolio.

■ LOG (15 MIN)

Write a slightly longer log today — 5 sentences:

What you built this week. What surprised you. What still feels unclear.

What you're most proud of. What you'll focus on in Week 2.

✓ DONE WHEN: Original project built and committed. Uses variables, loops, functions, and data structures. Week 1 log complete.

WEEK 2 · DAYS 8–14 · GOING DEEPER

Week 2 is where coding becomes real. The concepts get harder. The gap between 'I understand this' and 'I can build this' becomes visible. That gap is normal and it closes with practice. Push through it.

DAY 008 · OBJECTS

Everything in AI is an object. Models, layers, datasets.

■ INDUSTRY (15 MIN)

Search: 'how PyTorch works simple explanation'

PyTorch — the AI framework you'll use from Phase 3 — is built on objects.

■ STUDY (90 MIN) — Classes and OOP

Focus: classes, __init__, methods, self, instances.

Key insight: a neural network layer is a Python class.

A dataset is a Python class. A trained model is a Python class.

Understanding OOP is understanding how AI code is actually organized.

→ youtube.com — search 'Corey Schafer Python OOP Tutorial'

■ BUILD (60 MIN) — Student class

Build a class that models a NULLVECTOR student:

class NullVectorStudent:

def __init__(self, name, track, start_date)

def log_day(self, day_num, hours, notes)

def streak(self) — returns consecutive days logged

def completion_pct(self) — returns % of 150 days done

def report(self) — prints a full status summary

Create 3 different students. Log different days for each. Print their reports.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: NullVectorStudent class works. Three instances created with different data. Reports print correctly.

DAY 009 · LIBRARIES

You don't build everything from scratch. You stand on others' work.

■ INDUSTRY (15 MIN)

Search: 'how AI companies actually make money 2024 2025'

Understanding the industry you're entering matters as much as the technical skills.

■ STUDY (90 MIN) — NumPy, Pandas, Matplotlib

NumPy (20 min): numpy.org/doc/stable/user/quickstart.html

Arrays, array math, shape — the foundation of all ML math in Python.

Pandas (20 min): pandas.pydata.org/docs/getting_started/10min.html

DataFrames, reading CSVs, filtering — how you handle data in AI projects.

Matplotlib (20 min): matplotlib.org/stable/tutorials/introductory/quick_start.html

Line charts, bar charts, scatter plots — how you see your data.

Install all three: pip install numpy pandas matplotlib

■ BUILD (60 MIN) — Data analyzer

Create a CSV file called students.csv with columns:

name, days_completed, hours_per_day, track

Add 10 made-up students. Then build a program that:

1. Loads it with Pandas

2. Calculates average days_completed

3. Filters students who are more than 50% complete

4. Creates a bar chart with Matplotlib — days per student

5. Saves the chart as progress_chart.png

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: CSV loaded with Pandas. Stats calculated. Bar chart generated and saved as PNG.

DAY 010 · FILES + ERRORS

Real programs read and write files. Real programs handle mistakes.

■ INDUSTRY (15 MIN)

Search: 'AI is changing jobs 2025' — any recent article or video.

Context for why you are doing this matters. Keep it visible.

■ STUDY (90 MIN) — File I/O and Exception Handling

File I/O: open(), read(), write(), with statement.

Exception handling: try, except, finally — how programs survive unexpected input.

Key insight: every AI system reads files and handles errors constantly.

A model that crashes on bad input is useless in production.

■ BUILD (60 MIN) — Journal reader

Build a program that reads your log.md file and:

Counts how many days you've logged

Finds and prints the longest entry

Counts total words written across all entries

Handles the case where the file doesn't exist (use try/except)

Saves a summary to log_summary.txt

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Journal reader works, handles missing file gracefully, outputs summary file.

DAY 011 · ALGORITHMS

How you solve a problem matters as much as whether you solve it.

■ INDUSTRY (15 MIN)

Watch: Big O Notation explained — CS Dojo (9 min)

This connects directly to how AI models are evaluated for speed and efficiency.

→ youtube.com/watch?v=v4cd1O4zkGw

■ STUDY (90 MIN) — Search and Sort

Linear search vs binary search — understand why binary is faster.

Bubble sort vs built-in sort — understand what sorting costs.

Big O notation: O(n), O(log n), O(n²) — the language of algorithm efficiency.

You don't need to memorize proofs. You need intuition for why it matters.

■ BUILD (60 MIN) — Search and sort from scratch

Implement in Python — no built-in sort or search functions:

linear_search(list, value) — returns index or -1

binary_search(sorted_list, value) — returns index or -1

bubble_sort(list) — returns sorted list

Test all three on a list of 20 random numbers.

Time them with Python's time module. Compare results.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: All three functions work correctly. Timing comparison printed. Binary search is faster.

DAY 012 · PROBLEM SOLVING

The skill under all the other skills.

■ INDUSTRY (15 MIN)

Search: 'what makes a great software engineer' — any podcast or interview.

Spoiler: it's not memorizing syntax. It's how they approach problems.

■ STUDY (90 MIN) — Practice problems

Go to Codewars. Create a free account. Filter by Python, 8 kyu (easiest).

Do 5 problems today. Read each problem carefully before writing any code.

If you're stuck for more than 15 minutes: look at the hints, not the solution.

After solving: read other people's solutions. Notice what's different.

→ codewars.com — free, browser-based, thousands of Python challenges

■ BUILD (60 MIN) — Mini project sprint

Build 3 small things in 60 minutes — 20 minutes each:

1. FizzBuzz — classic, simple, everyone knows it, build it anyway

2. Palindrome checker — is 'racecar' the same forwards and backwards?

3. Caesar cipher — shift every letter by N positions to encode a message

These are interview problems. You will see them again.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: 5 Codewars problems solved. FizzBuzz, palindrome checker, and Caesar cipher all working.

DAY 013 · PUTTING IT TOGETHER

A slightly bigger project.

■ INDUSTRY (15 MIN)

Search: 'Mustafa Suleyman AI automation jobs 2025'

The CEO of Microsoft AI. What he's saying directly affects why you're here.

■ STUDY (60 MIN) — Review your weakest area

Look back at Days 8–12. What concept still feels unclear?

Classes? File I/O? Algorithms? Spend 60 minutes specifically on that gap.

Use Claude to explain it differently if it's not clicking.

Ask: 'explain [concept] like I'm new to programming and give me a simple example'

■ BUILD (90 MIN) — NULLVECTOR Progress Dashboard

Build a command-line dashboard a student could actually use:

Load their log.md and count entries

Show a simple progress bar: ■■■■■■■■■ 37%

Calculate their current streak

Show days until next phase milestone

List their GitHub repos (hardcode the names — real API comes later)

Use: classes, file I/O, lists, functions, string formatting.

This is portfolio-quality work. Treat it that way.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Progress dashboard loads log file, calculates streak, shows progress bar. Committed to GitHub.

DAY 014 · WEEK 2 COMPLETE

Consolidate and prepare for the bridge.

■ INDUSTRY (15 MIN)

Search: 'state of AI 2025 summary' — any comprehensive overview.

You have been coding for 2 weeks. See how much more context you have now.

■ STUDY (45 MIN) — Self test

Without looking at code: write a Python class with an __init__ and two methods.

Write a try/except block. Write a list comprehension.

Open a CSV file with Pandas and filter rows where a column is above a value.

If you can do all four: you are ready for Week 3.

If not: spend the remaining study time on whichever you couldn't do.

■ BUILD (90 MIN) — Week 2 Project

Build something meaningful using everything from Week 2:

A personal finance tracker — log income/expenses, show totals by category

A habit tracker — log daily habits, show streaks, export to CSV

A simple contact manager — add/search/delete contacts saved to a JSON file

Must use: at least one class, file I/O, error handling, and a library.

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Week 2 project complete and committed. Self-test passed on all four concepts.

WEEK 3 · DAYS 15–21 · THE BRIDGE TO AI

Week 3 is where general Python becomes AI-specific Python. The Kaggle courses this week are the bridge. By Day 21 you will understand the data structures and tools that all AI work is built on — and you will have proven it by completing them.

DAY 015 · KAGGLE INTRO

The platform where real data scientists work.

■ INDUSTRY (15 MIN)

Search: 'what is Kaggle and why do data scientists use it'

Understanding the ecosystem you're entering matters.

■ STUDY + BUILD (3 HRS COMBINED) — Kaggle Learn: Intro to Programming

Create a free Kaggle account at kaggle.com

Go to kaggle.com/learn and find 'Intro to Programming'

Work through all lessons. Do every exercise before moving to the next.

Kaggle's interface runs code in your browser — nothing to install.

By end of today: Intro to Programming course complete.

→ kaggle.com/learn/intro-to-programming

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Kaggle account created. Intro to Programming course complete. Certificate earned.

DAY 016 · KAGGLE PYTHON

Python the way data scientists use it.

■ INDUSTRY (15 MIN)

Search: 'Kaggle grandmaster how I got started' — any interview.

The people at the top of Kaggle all started exactly where you are.

■ STUDY + BUILD (3 HRS COMBINED) — Kaggle Learn: Python

Go to kaggle.com/learn and find the Python course.

Work through all 7 lessons: syntax, functions, booleans, lists, loops,

strings, and dictionaries.

This covers everything from Weeks 1–2 but in the context of data science.

It will feel familiar and also show you things you missed.

Complete the full course today.

→ kaggle.com/learn/python

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Kaggle Python course complete. All 7 lessons done. Certificate earned.

DAY 017 · PANDAS + DATA

The tool that handles every dataset in AI.

■ INDUSTRY (15 MIN)

Watch: '3Blue1Brown — But what is a neural network?' (first 10 min)

youtube.com/watch?v=aircAruvnKk — preview of what Phase 3 looks like.

■ STUDY + BUILD (3 HRS COMBINED) — Kaggle Learn: Pandas

Go to kaggle.com/learn and find the Pandas course.

Work through all 6 lessons: creating/reading/writing DataFrames,

indexing/selecting, summary functions, grouping, data types, renaming.

Pandas is how every AI practitioner handles data. This is not optional.

Complete the full course today.

→ kaggle.com/learn/pandas

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Kaggle Pandas course complete. All 6 lessons done. Certificate earned.

DAY 018 · DATA VISUALIZATION

Seeing your data is how you understand it.

■ INDUSTRY (15 MIN)

Search: 'how does a self driving car see the world'

Computer vision — what you'll build in Phase 4 — starts with visualizing data.

■ STUDY + BUILD (3 HRS COMBINED) — Kaggle Learn: Data Visualization

Go to kaggle.com/learn and find Data Visualization.

Work through all lessons: line charts, bar charts, scatter plots, distributions.

Complete the full course today.

Then: download a dataset from Kaggle (any that interests you)

and build one original chart from it — something Kaggle's tutorials didn't show you.

→ kaggle.com/learn/data-visualization

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Data Visualization course complete. Original chart built from a real Kaggle dataset.

DAY 019 · INTRO TO ML

Your first look at what Phase 2 teaches.

■ INDUSTRY (15 MIN)

Search: 'machine learning explained in 5 minutes' — any clear video.

After today's session, watch the same video again. Notice the difference.

■ STUDY + BUILD (3 HRS COMBINED) — Kaggle Learn: Intro to Machine Learning

Go to kaggle.com/learn and find Intro to Machine Learning.

Work through all 7 lessons: how models work, basic data exploration,

your first ML model, model validation, underfitting/overfitting, random forests.

This is a preview of Phase 2. It will make Phase 2 feel familiar.

Complete the full course today.

→ kaggle.com/learn/intro-to-machine-learning

■ LOG (15 MIN)

Three sentences. Push to GitHub.

✓ DONE WHEN: Intro to ML course complete. First machine learning model trained. Certificate earned.

DAY 020 · FINAL PROJECT DAY

Build something that uses everything.

■ INDUSTRY (15 MIN)

Listen: Lex Fridman Podcast — any episode, first 15 minutes.

lexfridman.com/podcast — pick any AI researcher. You'll recognize more terms now.

■ REVIEW (45 MIN)

Go through your GitHub repo. Read every file you've pushed.

How does Day 20 code compare to Day 1 code?

Write down 5 specific things you know now that you didn't know 19 days ago.

Keep that list — you'll need it on hard days in future phases.

■ BUILD (2 HRS) — Phase 1 Capstone

Build a data analysis project using real data:

Download any dataset from kaggle.com/datasets that interests you

Load it with Pandas

Clean it (handle missing values, wrong types)

Calculate 5 meaningful statistics

Create 3 visualizations

Write a 100-word summary of what you found

Save everything to a folder called 'phase1_capstone' in your GitHub repo

This is portfolio work. It shows you can work with real data independently.

■ LOG (15 MIN)

Five sentences today. This is an important one.

✓ DONE WHEN: Phase 1 capstone complete. Real dataset analyzed. 3 charts created. Summary written. All committed.

DAY 021 · PHASE 1 COMPLETE

21 days. The foundation is built.

■ INDUSTRY (15 MIN)

Watch: 'The future belongs to builders' — search for any recent AI founder talk.

You have been building for 21 days. This is no longer theoretical for you.

■ FINAL REVIEW (60 MIN)

Open your GitHub profile. Count the green squares. Count the commits.

Read your first log entry. Read your most recent one.

That distance — from Day 0 to Day 21 — is real. You created it.

■ GITHUB PROFILE UPDATE (60 MIN)

Update your nullvector-journey README to include:

Which track you completed

Kaggle courses completed with certificate links

3 projects you built with one-sentence descriptions of each

A one-paragraph description of what you can do now that you couldn't do 21 days ago

This README is the first thing Phase 2 asks you to update.

Start Phase 2 with it already done.

■ FINAL LOG (30 MIN)

Write your longest log entry yet. Answer these questions:

What were you most afraid of on Day 1?

What surprised you most about learning to code?

What are you most looking forward to in Phase 2?

What do you want to build when this is all over?

✓ DONE WHEN: GitHub shows 21 consecutive commits. README updated. Kaggle certificates earned. Phase 1 done.

DAY 21 MILESTONE · PHASE 1 COMPLETE — PYTHON FOUNDATIONS

The Phase 1 Milestone Is Not a Certification. It Is Three Things.

First: 21 consecutive days of GitHub commits — visible to anyone who looks at your profile. Second: four Kaggle course certificates — Intro to Programming, Python, Pandas, Data Visualization, and Intro to Machine Learning. Third: a GitHub README that shows what you built. You are ready for Phase 2. Phase 2 is CS50 AI — Harvard's course on artificial intelligence. You now have the Python foundation it requires. Everything gets more interesting from here.
