■ Month 1 — Offline Plan (Cybersecurity + AI)

You said you won't have access to ChatGPT or AI for a while, so here's a fully offline month plan. It includes exact tasks, commands, and project deliverables. Everything is scoped to run locally on your machine or inside a VM/lab. Ethical note: Only test security tools on your own machines or isolated lab VMs.

Week 1 — Environment + Python Core

Day	Plan & Deliverables
Day 1	Install: Python 3.11, VS Code, Git, GCC/Clang, GDB. Verify: python --version, pip --version. Create venv and activate. Deliverable: screenshot of 'python --version' & venv activated.
Day 2	Python refresh: variables, loops, functions, files. Do: write_to_file.py that logs 'Hello, Security + AI' 5 times.
Day 3	Networking intro: sockets. Build: port_scanner.py (single host, top 100 ports). Deliverable: run against 127.0.0.1.
Day 4	CLI ergonomics: argparse, logging. Add flags to port_scanner.py: --host, --start-port, --end-port.
Day 5	Data basics: numpy + pandas. Load csv, compute summary stats. Deliverable: stats.md with min/mean/max.
Day 6	Mini-project: CSV log parser (regex). Extract IPs, timestamps, and counts from a sample log file.
Day 7	Review day: Document what you built. Clean code and add README in your project folder.

Week 2 — Python for Security (Local Lab)

Day	Plan & Deliverables
Day 8	Set up a lab: Install VirtualBox/VMware. Download a security practice VM (e.g., Kali Linux). Keep it isolated (Host-Only network).
Day 9	Packet basics + scapy. Build: packet_sniffer.py capturing localhost traffic. Deliverable: pcap summary (counts by protocol).
Day 10	Write: simple firewall-like filter (read pcap or live), print/drop based on port list. Dry-run only (no packet injection).
Day 11	File hashing utility: compute SHA-256 for files in a directory to detect changes (integrity check).
Day 12	Automation: write backup script to zip a project folder with timestamp. Deliverable: zip output in /backups.
Day 13	Documentation day: Create SECURITY_NOTES.md explaining your lab setup, ethics, and how to run tools.
Day 14	Checkpoint: run through all scripts fresh in a new venv. Fix any missing deps.

Week 3 — AI Foundations (All Local)

Day	Plan & Deliverables
Day 15	Install: Jupyter, scikit-learn, matplotlib. Verify import. Create notebooks/ai_basics.ipynb.
Day 16	Data prep: load sms_spam_sample.csv. Split train/test. Deliverable: shapes + class balance printed.
Day 17	Model 1: Logistic Regression spam classifier. Save model (joblib). Deliverable: accuracy + confusion matrix.
Day 18	Model 2: RandomForest for the same task. Compare with Model 1. Deliverable: brief write-up which is better and why.
Day 19	Tiny time-series task: moving-average on dummy prices CSV. Plot to PNG. Deliverable: prices_ma.png.
Day 20	Packaging: make a CLI script classify_sms.py that loads the saved model and predicts on a text input.
Day 21	Review day: Create AI_NOTES.md with commands to train, evaluate, and run the CLI.

Week 4 — C Basics + Performance Thinking

Day	Plan & Deliverables
Day 22	Install C toolchain (if not already). Hello world in C. Deliverable: hello
Day 23	Pointers + arrays: implement dot product (dotprod.c). Compare with Python/Numpy timing (basic).
Day 24	Structs + IO: read simple struct array from file and compute an aggregate.
Day 25	Debugging: introduce gdb (set breakpoints, inspect variables).
Day 26	Mini-perf: write a naive matrix multiply in C; print timing vs Python nested loops.
Day 27	Wrap-up: summarize C learnings in C_NOTES.md; list when to prefer C/C++ for AI/security tooling.
Day 28	Capstone for the month: Run your spam CLI, port scanner, sniffer, and C programs in one sitting; fix issues; tag a release.

■■ *Ethical/Legal Reminder: Only test security scripts on systems you own or explicitly control (e.g., your lab VMs). Keep your VirtualBox network Host-Only or NAT, and never scan networks you don't own.*