■ 36-Month Computer Science Self-Study Syllabus

Year 1 – Programming & Problem Solving

Month 1

- Learn: Python modules, OOP basics (classes, objects, inheritance).
- Tasks: Read 'Automate the Boring Stuff' (ch. 10–12).
- Write a class-based text adventure game.

Month 2

- Learn: JavaScript async/await, event loop.
- Tasks: Watch a guide on the JS event loop.
- Build a weather app that fetches API data.

Month 3

- Learn: Error handling & testing (pytest).
- Tasks: Write unit tests for your projects.
- Automate a small real-life task (e.g., rename files).

Month 4

- Learn: Arrays, linked lists. Big-O notation basics.
- Tasks: Implement linked list in Python.
- Solve 10 array problems on LeetCode.

Month 5

- Learn: Stacks, queues, hash tables.
- Tasks: Implement stack/queue/hash map.
- Solve 15 problems involving them.

Month 6

- Learn: Trees (binary search trees).
- Tasks: Implement a BST (insert/search/delete).
- Solve 10 tree problems.

Month 7

- Learn: Graphs & recursion.

- Tasks: Implement DFS and BFS.
- Solve 10 graph problems.

Month 8

- Learn: Sorting algorithms (merge/quick sort).
- Tasks: Implement 3 sorting algorithms.
- Time them on large datasets.

Month 9

- Learn: Git branching/merging, Linux basics.
- Tasks: Do MIT 'Missing Semester' Git lessons.
- Set up GitHub portfolio.

Month 10

- Learn: Bash scripting.
- Tasks: Write 3 scripts (backup, search logs, auto-update).

Month 11

- Learn: Testing practices & refactoring.
- Tasks: Refactor an old project with tests.
- Write blog post explaining changes.

Month 12

- Capstone Project: Build a small CLI app (to-do list manager, file organizer, or notes app).

Year 2 – Systems Foundations

Month 13

- Learn: CPU basics, machine code, registers.
- Tasks: Read ch. 1–2 of 'Computer Organization and Design'.
- Run simple assembly on a simulator.

Month 14

- Learn: Memory hierarchy (RAM, cache, storage).
- Tasks: Diagram CPU cache vs RAM.
- Write Python benchmark to compare file I/O vs memory speed.

Month 15

- Learn: Instruction sets, compilers overview.
- Tasks: Hand-trace assembly for a small program.
- Build a 'tiny calculator' in assembly or C.

Month 16

- Learn: OS processes, scheduling.
- Tasks: Read 'OSTEP' ch. on processes.
- Write a program that spawns child processes.

Month 17

- Learn: Threads & concurrency.
- Tasks: Implement producer/consumer in Python threading.
- Write blog post on race conditions.

Month 18

- Learn: Memory management & filesystems.
- Tasks: Simulate malloc/free in Python.
- Write a program that lists files recursively (like 'tree').

Month 19

- Learn: Networking layers, OSI model.
- Tasks: Read 'Computer Networking: Top-Down' ch. 1–2.
- Packet sniff with tcpdump/Wireshark.

Month 20

- Learn: TCP/UDP, sockets.
- Tasks: Write a TCP chat server.
- Write a UDP file transfer tool.

Month 21

- Learn: HTTP, DNS.
- Tasks: Build minimal HTTP server in Python.
- Write a DNS lookup tool.

Month 22

- Learn: Databases, SQL basics.
- Tasks: Build database schema for blog.
- Learn SELECT/INSERT/UPDATE/DELETE queries.

Month 23

- Learn: Transactions, indexes, joins.
- Tasks: Optimize queries with indexes.
- Build analytics queries on sample dataset.

Month 24

- Capstone Project: Build a blog/note-taking app with SQL database + web server.

Year 3 – Theory & Specialization

Month 25

- Learn: Lexing & parsing.
- Tasks: Read 'Crafting Interpreters' part I.
- Write a calculator parser.

Month 26

- Learn: Interpreters.
- Tasks: Extend calculator into mini language (variables + conditionals).

Month 27

- Learn: Compilers basics.
- Tasks: Write a transpiler from your toy language → Python/JS.

Month 28

- Learn: Discrete math – sets, logic, proofs.
- Tasks: Work through MIT 'Math for CS' week 1–3.
- Solve proof exercises.

Month 29

- Learn: Graph theory & combinatorics.
- Tasks: Implement Dijkstra’s algorithm.
- Solve graph problems on HackerRank.

Month 30

- Learn: Automata & Turing machines.
- Tasks: Implement NFA → DFA converter.
- Write about decidability/halting problem.

Month 31

- Learn: Complexity theory.
- Tasks: Study P vs NP.
- Try solving classic NP problems (SAT, Knapsack).

Month 32

- Learn: Advanced Linux (system calls, signals).
- Tasks: Write program that installs a custom signal handler.
- Explore /proc filesystem.

Month 33

- Learn: Security basics (crypto, exploits).
- Tasks: Implement Caesar, RSA in Python.
- Read about buffer overflows.

Month 34

- Learn: Advanced Git & software engineering.
- Tasks: Contribute to open-source project.
- Practice code reviews.

Month 35

- Specialization Month (pick 1):
- AI/ML → Learn linear algebra, scikit-learn.
- Systems → Write kernel module.
- Security → Capture The Flag challenges.
- Web → Build scalable web app.
- Game Dev → Build small game engine.

Month 36

- Final Capstone Project: Large project combining everything learned.
- Examples: interpreter + web interface, multiplayer game, ML research project, or tiny OS kernel.
- Document thoroughly on GitHub.