**📘 Google Apprenticeship Prep Roadmap (2025)**

**1️⃣ Understand the Hiring Phases**

* **Screen** → Online coding + resume/application review
* **Interview** → Coding, problem solving, light system design, Googliness
* **Decide** → Overall evaluation

👉 Goal: You need **strong DSA + clear communication + basic system design + teamwork mindset**.

**2️⃣ Core Coding Prep (Daily Practice)**

**🔹 Data Structures & Algorithms (must-know)**

* Arrays & Strings → sliding window, two pointers, prefix sums
* Linked Lists → reverse, detect cycle, merge
* Hash Tables / Sets → frequency maps, lookups
* Trees & Graphs → BFS, DFS, shortest path (BFS/Dijkstra basics), recursion
* Sorting & Searching → quicksort, mergesort, binary search
* Dynamic Programming → subsequence problems (LCS, LIS, knapsack)

📌 **Action Plan**:

* Solve **1-2 LeetCode Easy/Medium daily**
* Mix topics: do 5 arrays, then 5 strings, then 5 trees, etc.
* Keep a **notes doc** for patterns → this will be your revision sheet.

**Resources**:

* LeetCode Patterns
* HackerRank Interview Prep Kit
* Neetcode.io (excellent curated roadmap)

**3️⃣ System Design Basics (for interviews)**

Not full-blown design like senior SWE, but you should know:

* Client-server model
* REST APIs (how they work, status codes)
* Database design basics (tables, primary key, indexing)
* Caching (e.g., Redis basics)
* Scalability → load balancer, replication basics

📌 **Action Plan**:

* Read [System Design Primer](https://github.com/donnemartin/system-design-primer?utm_source=chatgpt.com) (intro sections only)
* Practice explaining **how you’d design a simple URL shortener or ToDo app**

**4️⃣ Programming Language Focus**

Pick **one primary language** (Python or Java recommended).

**Python Focus**:

* Practice writing clean, readable functions
* Learn collections module (Counter, defaultdict, heapq)
* OOP basics: classes, inheritance
* File handling, exceptions

**Java Focus**:

* Learn Collections API (ArrayList, HashMap, HashSet)
* OOP principles (encapsulation, inheritance, polymorphism, abstraction)
* Exception handling & streams

📌 Don’t try to master all 4 languages → focus on 1 solid + be aware of others.

**5️⃣ Git & Collaboration Skills**

Google values teamwork + collaboration.

* Learn basic **GitHub workflows**: clone, commit, branch, PR
* Try **one small open-source contribution** (even documentation counts!)
* Document your code (docstrings, README.md)

**6️⃣ Communication & “Googliness”**

During interviews:

* **Talk out loud** while coding → show how you’re thinking
* Ask clarifying questions → shows problem-solving maturity
* Be positive, curious, collaborative

📌 Practice **mock interviews**:

* Pramp (free peer interviews)
* Interviewing.io (sometimes free mock interviews with real engineers)

**7️⃣ Resume & Application Review**

Since this is an apprenticeship (not a senior SWE role):

* Highlight **projects** (web apps, small games, automation scripts, open source)
* Show **learning ability** → online courses, certifications, hackathons
* Mention teamwork/collaboration (college projects, GitHub collabs)

**8️⃣ 8-Week Prep Plan (Sample Schedule)**

**Week 1-2** → Arrays, Strings, Hashing, Linked Lists (LeetCode Easy/Medium)  
**Week 3-4** → Trees, Graphs, Sorting, Searching  
**Week 5** → Dynamic Programming basics + review past topics  
**Week 6** → System Design basics + REST APIs + databases  
**Week 7** → Mock interviews (Pramp, friends, ChatGPT roleplay)  
**Week 8** → Review notes + final polish (resume, GitHub profile, projects)

✅ If you follow this plan, you’ll be ready for:

* Technical screen (DSA + problem solving)
* Interviews (clear thought process + system design basics)
* Application (resume + projects that show learning ability)

Would you like me to **create a daily/weekly detailed study schedule (with resources + exact LeetCode problems)** so you can just follow it step by step till interview?

**You said:**

yes

**ChatGPT said:**

Perfect 👍 I’ll build you a **Daily/Weekly Google Apprenticeship Study Schedule (8 weeks)**.  
This plan balances **DSA practice, system design basics, projects, and communication skills**.

**📅 8-Week Google Apprenticeship Prep Schedule (2025)**

**Week 1: Arrays & Strings (Foundations)**

**Goal**: Build strong base + get used to problem-solving format.

* **DSA (LeetCode Easy/Medium – 2 per day)**
  + Two Sum
  + Best Time to Buy and Sell Stock
  + Valid Anagram
  + Group Anagrams
  + Longest Substring Without Repeating Characters
  + Product of Array Except Self
  + Maximum Subarray
* **Concepts**: Time complexity (O, Ω, Θ), Big-O for loops/nested loops.
* **Extra**: Start keeping a **“Problem-Solving Notes” doc** → write approach + edge cases.

📌 **Resources**: Neetcode Arrays & Strings playlist

**Week 2: Linked Lists, Hashing, Stacks, Queues**

**Goal**: Learn how to handle pointers + data lookup.

* **DSA Problems**:
  + Reverse Linked List
  + Merge Two Sorted Lists
  + Linked List Cycle
  + LRU Cache (use hashmap + doubly linked list)
  + Valid Parentheses (stack)
  + Min Stack
* **Concepts**: Hashmaps, Sets, Stack/Queue operations
* **System Design**: What is an API? Example of REST API call.

📌 **Project Idea**: Build a small **To-Do CLI app in Python/Java** with add/remove tasks. Use GitHub.

**Week 3: Trees & Binary Search**

**Goal**: Recursive thinking + divide & conquer.

* **DSA Problems**:
  + Binary Tree Inorder Traversal
  + Maximum Depth of Binary Tree
  + Validate Binary Search Tree
  + Lowest Common Ancestor of BST
  + Binary Search
  + Search in Rotated Sorted Array
  + Find First and Last Position in Sorted Array
* **Concepts**: Recursion, Tree traversal (DFS/BFS), Binary Search
* **System Design**: Database basics (SQL tables, primary/foreign keys).

📌 **Extra**: Try SQL basics on HackerRank (SELECT, WHERE, GROUP BY).

**Week 4: Graphs + Sorting**

**Goal**: BFS/DFS + sorting mastery.

* **DSA Problems**:
  + Number of Islands (BFS/DFS)
  + Clone Graph
  + Course Schedule (Topological sort)
  + Pacific Atlantic Water Flow
  + Merge Intervals
  + Insert Interval
  + Meeting Rooms II
* **Concepts**: BFS, DFS, Graph representation (Adj list vs Matrix)
* **System Design**: Client-server architecture explained with **Google Docs (collaboration)** example.

📌 **Mini Project**: Implement **Graph traversal visualizer** (optional, Python with networkx or just text).

**Week 5: Dynamic Programming (DP)**

**Goal**: Get comfortable with subproblems & memoization.

* **DSA Problems**:
  + Climbing Stairs
  + House Robber
  + Coin Change
  + Longest Increasing Subsequence
  + Longest Common Subsequence
  + Edit Distance
* **Concepts**: Recursion → Memoization → Tabulation
* **System Design**: Caching (ex: how login sessions work).

📌 **Extra**: Read about **Redis caching basics**.

**Week 6: Full-Stack Basics + Mock Coding**

**Goal**: Combine coding + real-world knowledge.

* **DSA**: Mix review of all past topics (3 problems/day random).
* **Web Dev Basics**:
  + HTML/CSS basics
  + JavaScript DOM manipulation
  + Build a simple React app (if time permits)
* **System Design**: Explain how to build a **URL shortener** or **Chat app** (basic).
* **GitHub Practice**: Branch → PR → Merge flow.

📌 **Mini Project**: Make a **simple blog app** (Python Flask/Node.js) → store posts in SQLite/JSON.

**Week 7: Interview Simulation + Polishing**

**Goal**: Transition from coding → interviews.

* **Daily Routine**:
  + Solve 1 new LeetCode problem
  + Review 2-3 solved problems (explain aloud as if in interview)
* **Mock Interviews**:
  + Pramp (peer interviews)
  + Interviewing.io (if available)
  + Ask friend/mentor/ChatGPT to roleplay
* **System Design**: Practice whiteboarding **design for To-Do app, URL shortener, E-commerce cart**
* **Communication Practice**: Practice explaining **thought process clearly aloud**

**Week 8: Final Prep & Revision**

**Goal**: Be confident + well-rested.

* **DSA**: Revise notes + common patterns (sliding window, two pointers, binary search, DP).
* **System Design**: Review your answers → keep it simple and structured.
* **Resume Polish**: Highlight projects + GitHub contributions.
* **Mindset**: Google wants curiosity, collaboration, and learning agility → show passion.

📌 Day before interview: Do **2 easy problems only** → don’t stress.

**✅ Final Notes**

* **1-2 hours daily** if you’re consistent = enough to clear
* Focus on **clarity > complexity** in answers
* Google values **clean code, collaboration, curiosity**