Roadmap to Crack Google/Microsoft with C++ + Python (12 Months Plan)

Phase 1: Foundation (Months 1-3)

A. Basic C++ Topics (for Beginners)

- Introduction to C++
- Structure of a C++ Program
- Input/Output (cin, cout)
- Data Types and Variables
- Operators (Arithmetic, Logical, Relational, Bitwise)
- Conditional Statements (if, if-else, switch)
- Loops (for, while, do-while)
- Functions (declaration, definition, call by value/reference)
- Arrays and Strings Basics
- Pointers and References
- Dynamic Memory Allocation (new/delete)
- Object-Oriented Programming Basics
    - Classes and Objects
    - Constructors and Destructors
    - Inheritance
    - Polymorphism (Function Overloading, Operator Overloading)
    - Encapsulation and Abstraction
- File Handling in C++
- Standard Template Library (STL): vector, map, set, pair, stack, queue

A. Data Structures & Algorithms (using C++)

- Arrays & Strings
- Linked Lists
- Stacks & Queues
- Hashing

- Recursion & Backtracking
- Trees (Binary Tree, BST)
- Heaps
- Graphs (BFS, DFS, Dijkstra, Union-Find)
- Greedy & Two Pointer
- Sliding Window
- Dynamic Programming (0/1 Knapsack, LIS, etc.)

B. Python Fundamentals (for Beginners)

- Introduction to Python
- Installing Python and IDEs (PyCharm, VS Code)
- Python Syntax and Code Structure
- Variables and Data Types (int, float, string, bool, list, tuple, set, dict)
- Type Casting
- Operators (Arithmetic, Logical, Comparison)
- Conditional Statements (if, elif, else)
- Loops (for, while, break, continue)
- Functions (arguments, return values, default parameters)
- Lambda Functions
- List Comprehensions
- String Manipulation and Formatting
- Working with Files (read, write, append)
- Exception Handling (try, except, finally)
- Modules and Packages
- Introduction to OOP (classes, objects, inheritance)
- Using Built-in Libraries (math, datetime, os, random)
- Syntax & Data Types
- Functions & Modules
- File Handling

- OOP in Python
- Exception Handling
- Built-in Libraries (os, datetime, math, etc.)

Phase 2: Core CS + Python Projects (Months 4-6)

A. Core CS Subjects (Theory)

- Operating System: Processes, Threads, Scheduling Algorithms, Deadlocks, Memory Management
- DBMS: SQL, Joins, Normalization, Indexing, ACID, Transactions
- Computer Networks: TCP/IP, OSI Model, HTTP/HTTPS, DNS, IP Addressing, Routing
- OOPs Concepts: Classes, Objects, Inheritance, Polymorphism, Abstraction, Encapsulation

B. Python Projects (1 per month)

- File Organizer Script
- Blog API using Flask/Django
- News Scraper using BeautifulSoup

Phase 3: Resume & Competitive Edge (Months 7-9)

A. Resume Preparation

- Format a 1-page tech resume
- Add C++ problem solving + Python projects
- Add LeetCode/Codeforces + GitHub links

B. Competitive Programming Practice

- Regular contests (Codeforces, AtCoder, LeetCode)
- Solve rated problems
- Timed practice + speed improvement

C. System Design (Basics)

- Load Balancing
- Caching

- REST APIs
- Scalability Concepts
- Database Sharding
- Message Queues
- System Design Primer (GitHub)

Phase 4: Job Preparation (Months 10-12)

A. Aptitude, Reasoning & Language Skills

Quantitative Aptitude:

- Number System
- HCF & LCM
- Time, Speed, and Distance
- Time and Work
- Ratio and Proportion
- Percentages and Averages
- Profit, Loss, and Discount
- Simple and Compound Interest
- Mixtures and Alligations
- Permutations and Combinations
- Probability
- Clocks and Calendars

Logical Reasoning:

- Puzzles
- Blood Relations
- Direction Sense
- Coding-Decoding
- Syllogisms
- Seating Arrangement

- Statement and Conclusion
- Visual Reasoning

Verbal Ability (Grammar & Comprehension):

- Sentence Correction
- Spotting Errors
- Sentence Completion
- Fill in the Blanks
- Synonyms & Antonyms
- Reading Comprehension
- One Word Substitution
- Para Jumbles
- Active & Passive Voice
- Direct & Indirect Speech

Qualitative Analysis:

- Data Interpretation
- Pie Charts, Bar Graphs
- Tables and Line Charts
- Caselets and Data Sufficiency
- Logical Data Analysis

Literature (for polishing communication):

- Reading technical blogs and essays
- Writing summaries of books or articles
- Practicing email and report writing

B. Mock Interviews & Applications

- Pramp, Interviewing.io for mock practice
- Apply on LinkedIn, AngelList, Internshala
- Reach out for referrals

C. Final Revisions

- Revise DSA key topics
- Practice top 150 LeetCode problems
- Resume polish & project final touches

Tools & Resources:

- C++ DSA: Striver Sheet, LeetCode, NeetCode
- Python: RealPython, Automate the Boring Stuff
- CS Subjects: Gate Smashers (YouTube)
- Projects: Flask, Django, GitHub
- Aptitude: IndiaBix, RS Aggarwal, PrepInsta
- System Design: Gaurav Sen, ByteByteGo, Tech Dummies
- Verbal Skills: Grammarly, Hemingway App, Word Power Made Easy (Norman Lewis)

Weekly/Daily Study Schedule (First 3 Months)

Phase 1: Foundation (Weeks 1–12)

Week 1–2:

- **C++**: Introduction, Structure, I/O, Variables, Data Types, Operators, Conditionals
- **Python**: Introduction, Installation, Syntax, Variables, Data Types, Operators
- **DSA**: Arrays Basics

Week 3–4:

- **C++**: Loops, Functions (Basics), Arrays, Strings
- **Python**: Loops, Functions, String Manipulation
- **DSA**: Strings, Linked Lists

Week 5–6:

- **C++**: Pointers, Dynamic Memory, OOP (Classes & Objects)
- **Python**: File Handling, List/Tuple/Set/Dict Operations
- **DSA**: Stacks, Queues, Hashing

Week 7–8:

- **C++**: Constructors, Inheritance, Polymorphism, STL Basics (vector, map, set)
- **Python**: Exception Handling, OOP in Python, Modules
- **DSA**: Recursion, Backtracking, Trees

Week 9–10:

- **C++**: File Handling, STL (stack, queue, pair)
- **Python**: Built-in Libraries (math, datetime, os)
- **DSA**: Heaps, Graph Basics

Week 11–12:

- **C++ & Python**: Revise OOP + Practice Problems (HackerRank, LeetCode Basics)
- **DSA**: Graph Algorithms (BFS, DFS), Sliding Window, Greedy, Intro to DP

Spend 2 hours/day minimum:

- 1 hour DSA (C++)
- 30 mins Python fundamentals
- 30 mins theory/practice

Weekly/Daily Study Schedule (Months 4–6: Core CS + Python Projects)

Phase 2: Core CS + Python Projects (Weeks 13–24)

Week 13–14:

- **CS Theory**: Operating System Basics (Processes, Threads, Scheduling)
- **Python Project**: File Organizer (work on features step by step)
- **DSA**: Continue practicing Trees, Recursion, and revisit Backtracking

Week 15–16:

- **CS Theory**: OS (Deadlocks, Memory Mgmt), DBMS (SQL, Joins, Normalization)
- **Python Project**: Flask Blog API – Set up, Routing, CRUD
- **DSA**: Binary Search Trees, Practice LeetCode Easy-Medium

Week 17–18:

- **CS Theory**: DBMS (Indexing, ACID), CN (TCP/IP, OSI Layers)
- **Python Project**: Flask Blog API – Auth, Deployment
- **DSA**: Heaps, Graph Basics (BFS, DFS)

Week 19–20:

- **CS Theory**: CN (Routing, HTTP, DNS), OOPs Full Review
- **Python Project**: News Scraper with BeautifulSoup
- **DSA**: Dijkstra, Union-Find, Shortest Path Algorithms

Week 21–22:

- **Python Projects**: Improve UI or APIs, add GitHub README
- **Resume Draft**: Add projects, GitHub links, initial resume formatting
- **DSA**: Sliding Window, Greedy, Intro to DP

Week 23–24:

- **System Design Intro**: Load Balancing, Caching, Scalability
- **CS Review**: Revisit DBMS + OS key concepts
- **DSA**: Practice 10-15 mixed questions (trees, heaps, recursion)

Spend 3 hours/day:

- 1 hour CS Theory
- 1 hour Python Project
- 1 hour DSA Practice

Weekly/Daily Study Schedule (Months 7–9: Competitive Edge + Resume)

Phase 3: Resume & Competitive Edge (Weeks 25–36)

Week 25–26:

- **Resume**: Final polish, include GitHub + LinkedIn profile
- **DSA**: Start solving rated problems (Codeforces/LeetCode)
- **System Design**: Read System Design Primer

Week 27–28:

- **Mock Interviews**: Pramp, Interviewing.io (2 per week)
- **Competitive Programming**: Timed contests
- **Python**: Use Python to script common CP utilities

Week 29–30:

- **System Design**: REST APIs, Sharding, Message Queues

- **DSA**: 5 Medium problems/day from LeetCode (DP, Graphs)
- **CS Theory**: Revise OS, DBMS, CN for interviews

Week 31–33:

- **CP**: Solve 30+ problems, work on speed
- **Projects**: Clean code, final commits, documentation
- **Mock Interview**: With friends or online platform

Week 34–36:

- **System Design**: Design full systems (TinyURL, Instagram)
- **Final Resume Touch-Up**: Use feedback, finalize
- **Interview Prep**: HR questions, intro, projects explanation

Weekly/Daily Study Schedule (Months 10–12: Job Prep)

Phase 4: Job Preparation (Weeks 37–52)

Week 37–40:

- **Aptitude**: Number Systems, Ratio, TSD, Averages
- **Reasoning**: Puzzles, Blood Relations, Directions
- **Verbal**: Sentence Correction, Fill in the blanks, RC
- **Qualitative**: Bar/Pie Charts, Data Interpretation

Week 41–44:

- **Aptitude**: Profit-Loss, Time & Work, Permutations, Probability
- **Reasoning**: Syllogism, Seating, Statement & Conclusion
- **Verbal**: Voice, Speech, Parajumbles, Synonyms/Antonyms
- **Literature**: Write 1 summary/week, email practice

Week 45–48:

- **Mock Interviews**: 2–3/week on Pramp or peers
- **DSA**: Top 150 LeetCode questions
- **System Design**: Revise concepts and examples

Week 49–52:

- **Full Revision Week**:

- DSA core topics recap
- Python mini-recap
- OS, DBMS, CN quick revision
- Resume polish, GitHub cleanup
- Interview questions and practice answers