Day - 0

**100 Days of Code –**

**A Complete Guide For Beginners & Experienced**

<https://www.geeksforgeeks.org/100-days-of-code-a-complete-guide-for-beginners-and-experienced/>

1. **100 Days of Code**

* You might have tried to do some *preparation* and you might have tried to cover *various* *topics*. You make a *plan*, you try to follow your plan and you start doing preparation through various *resources*, *tutorials*, and *videos*.
* You start learning it but after a couple of days or months, you realize that you’re not going anywhere. You’re not making any progress and things are overwhelming for you.
* You’re not alone and it’s happening with so many newbie programmers and experienced developers. They try to learn to code, and they try to cover various topics but due to the lack of a good ROADMAP and GUIDANCE, things become *messy* and *confusing* for them. During their preparation, they *miss* out on a lot of *important* *things* to cover.
* 100 Days of Code: To help you with this we have created a practical roadmap **“100 days of code”** based on a lot of research and the experience of software developers working in big tech companies. This roadmap will give you a complete guideline to build a strong coding habit and to achieve your goal as a software developer. All you just need is a *laptop*, a working *internet* *connection*, determination to *solve* *problems* *regularly* and within a few months, you can develop the Coding Habit, learn the *essentials*, and *crack* *the* *interviews*.
* **What is 100 Days of Code?**

**What is the reasonable timeframe to crack the interviews of big tech companies?**

* Well, there are a lot of topics and things to cover if you’re targeting some big tech companies. The challenge is the limited timeframe and based on our research we need to say that it may take almost 14 weeks or 100 Days. We have divided the complete topics into 100 days and all you just have to do is to follow it and stay committed to coding for 100 days.
* If you follow this approach and stay committed to coding then not only you will develop a good coding habit but also you will be able to crack the interviews of 60-70% of tech companies.
* We will cover 5-6 main topics in 100 days, and we will target the interviews of big tech companies such as ***Facebook***, ***Google***, ***Microsoft***, ***Amazon***, etc. These topics are given below…

1. **Data Structures and Algorithms (Coding part and problem solving)**
2. **System Design**
3. **Operating System and DBMS (database management system)**
4. **Object-Oriented Design**
5. **Cultural Fit**

* Few things to need to keep in mind before we continue with the roadmap…

1. If you’re a beginner then make the target to solve almost *450-500 coding questions*.
2. If you’re an experienced person then make the target to solve almost *300 to 350 questions*.
3. *System* *design* *round* is mostly asked from *experienced* *people* so beginners have to focus on this section less but experienced one has to focus on this section more.
4. If you’re a beginner then try to commit your time for at least 4-6 hours every day.
5. If you’re an experienced one then you need to commit for 2-3 hours every day.
6. Ideally, you shouldn’t take more than 20–30 minutes to solve any given problem but keep in mind that this rule is not applicable to all the questions.

* Prerequisite: We are assuming that if you’re a beginner then you already have completed some projects in your college and if you’re an experienced developer then surely you might be working on some projects.
  1. **100 Days of Code For Beginners**

Before you go ahead keep in mind that being a *newbie* *programmer* you will be overwhelmed with so many questions and you will face a *lot of difficulty in solving the problems*. It’s okay if you take more time in solving the questions but don’t be discouraged.

* Try to solve the problems on your own first even if it takes a couple of more hours.
* After *spending* many *hours* if you don’t get the solution then you can *stretch yourself for a few days* or *take the hints*. As you will progress you will see improvement in yourself and also you will take less time in solving the questions.

1. Day 1 to Day 10: Choose a Programming Language and Learn The Fundamentals of it.

You need to choose a programming language and you need to be stick with it. Explore various programming languages, read about them, try it out in your code editor, and check if you’re comfortable with it or not.

A lot of questions will arise when you will be choosing a language.

* Is Python better than Java?
* C++ is a faster language, so I should choose this language,
* Java syntax is more complicated but big companies are working on it,
* what if I know more than one language?

1. 4-5 days: Explore the languages and pick one which is best for you. ***Don’t choose a language just because it is very popular in the tech industry***. You need to choose a language you’re ***most comfortable*** with. Almost it will take 4-5 days in picking up a language.
2. Learn the fundamentals: Once you choose a language learn the fundamentals of it but if you already know the fundamentals (you might have created your project in that language) then brush up on it.

* It has been seen that a lot of people forget the basic things of their chosen language. So you need to make sure that you have a good command of your chosen language.
* Learn the basic syntax, give some online tests, and remember the basic things of your chosen language. It may take around one week or 7 days to learn the fundamentals of your language.
* Few things you should remember are…

1. Reading/Writing from/to files.
2. Reading input from the console.
3. Split strings
4. 2D array implementation and how to use it.
5. Very basics things of language such as in C/C++ handling null-terminated strings

* Laptop-typing: A lot of companies such as Lyft and Salesforce have a slight difference in their hiring procedure. They want the candidate ***to write code and solve the problems on a laptop***.
* You will have to write the complete program and your code should pass the given test cases. You may have to work on command line arguments or you may ***have to parse the*** CSV ***or*** textfiles.
* Write on whiteboard: Other companies such as Google, Amazon, or Microsoft use a whiteboard. So it’s good that you know the basics of your language to handle both of the coding scenarios in your interview. So almost it will take 10 days to make a decision that which language is best for you and you’re comfortable with.

Resource:

* ***C Programming Language***
* ***C++ Programming Language***
* ***Java Programming Language***
* ***Python Programming Language***

1. Day 11 to Day 20: Learn Data Structures and Algorithms

Now it’s time to learn the most important part of the coding i.e. DATA STRUCTURES and ALGORITHMS. This subject is mostly taught in undergrad and most people ignore the importance of it. But the topic in this subject is the most important thing to cover in the coding interviews.

* Learn the basics of various data structures and algorithms including the complexity part. Topics are given below…

1. Time and Space Complexity Analysis
2. Arrays
3. Stacks
4. Queues
5. Linked List
6. Trees
7. Tries
8. Graphs (BFS and DFS)
9. Hashing
10. Heaps
11. Sorting
12. Searching
13. Cover all the above fundamentals and ***implement various DSA in your code editor***. You also need to cover the String part which is most likely asked in the interviews.
14. If you’re thinking that you already know all the ***topics in DSA then just give some online tests*** and ***test your skill***. You will surely find that you still need to revise the basic concepts.
15. Learn to implement array, linked list, stack, queue, tree, etc. The ***basic foundation helps a lot in solving the coding question*** so do not skip it and have a good grasp on it.

Resource:

* GeeksforGeeks Data Structures
* GeeksforGeeks Algorithms

1. Day 21 to Day 40: Solve Basic Level Problems

Once you get familiar with the basic Data Structure and Algorithm practice for the ***basic coding questions***. Practicing basic level questions will help you in stretching your mind and it will prepare you to tackle medium level problems.

* Do not skip the basic level otherwise, you will face difficulty in solving harder problems and you will have to go back to the basic level again.
* Try to solve almost 200 basic level questions within 10 days.
* Depending on the time and speed you can increase the number of questions.
* Almost 8-12 questions a day within the timeframe of 4-6 hours will help you in building a good foundation.

Sample Array Problems

1. Merge Two Sorted Arrays
2. Subarray with given sum
3. Find duplicates in an array
4. Missing number in an array
5. Array Rotation

Sample Linked List Problems

1. Detect Loop in a linked list
2. Rotate a linked list
3. Merge two sorted linked list
4. Searching in the linked list
5. Find a middle element in a linked list

Sample Stack and Queue Problems

1. Sort values in Stack
2. Implement Two Stacks using one Array

Sample Tree Problems

1. Minimum value in Binary Search Tree
2. Height of a Binary Tree
3. kth maximum value in Binary Search Tree

Sample Graph Problems

1. Implement Breadth-First Search and Depth First Search

Sample Heap Problems

1. Find k smallest and largest elements in a list
2. Day 41 to Day 60: Solve Medium Level Problems

Now it’s time to practice for some medium level questions. Medium level problems will prepare you to solve harder problems and these questions are most likely asked in the interviews. Try to solve almost 150-170 medium level questions within 20 days. Depending on the time and speed you can increase the number of questions. Almost 7-8 questions a day within the timeframe of 4-6 hours are good enough.

Sample Array Problems

1. Inversion of array
2. Maximum of all subarrays of size k
3. Trapping rainwater
4. Kadane’s algorithm

Sample Linked List Problems

1. First non-repeating character in a stream
2. Rotate a linked list
3. Remove loop in a linked list
4. LRU Cache
5. Clone a linked list
6. Find the Intersection point of two linked lists

Sample Stack and Queue Problems

1. Get minimum element from the stack
2. Queue using Two Stacks
3. Stack using Two Queues

Sample Tree Problems

1. Check for BST
2. Vertical traversal of binary tree
3. Boundary traversal of binary tree
4. Delete a node from BST

Sample Graph Problems

1. Detect Cycle in a directed and undirected graph
2. Find the number of islands
3. Word Boggle
4. Implementing Dijkstra
5. Day 61 to Day 75: Solve Hard Level Problems

Once you solved the basic and medium level question, move to the hard problems which are the last ones. Try to solve almost 80-90 hard level questions within 15 days. Depending on the time and speed you can increase the number of questions.

Sample Problems

1. Find median in a stream
2. N-Queen problem
3. Sudoku problem
4. Traveling salesman problem
5. Boolean parenthesization

* Best Resource for Solving Coding Questions: GeeksforGeeks Practice Portal

1. Day 76 to Day 85: System Design

For beginners, this section is not much important but sometimes interviewers asked system design-related questions to the non-experienced candidates as well. So it’s good to know about the system design part as well. Firstly learn the basic concept of system design such as ***Load*** ***Balancing***, ***Caching***, ***Proxies***, ***CAP*** ***Theorem***, ***Consistency***, ***Sharding*** or ***data*** ***partitioning***, etc.

You will be asked to design a web-scale service. So once you’re done with basic concepts, prepare yourself for some most likely system design round questions such as…

1. URL Shortening Service (TinyURL)
2. Design Facebook Newsfeed
3. Design Uber or Lyft

* Preparing these questions will help you in handling this round and you will get to know what exactly you have to do in this round. You will be taking ownership of this round and you will describe the different components of the system.
* Interviewers check that if the candidate is able to build a large and complex system or not. As we have mentioned this round is mostly not asked from the beginners so you don’t need to go into too much depth of this section. However, you should have the knowledge that what exactly happens in this round so in case if it is asked in the interview, you can handle it.
* Must Go Through:

1. 5 Common System Design Concepts for Interview Preparation
2. How to Crack System Design Round in Interviews
3. Top 10 System Design Interview Questions and Answers
4. Design Twitter
5. Design Dropbox
6. Day 86 to Day 92: OS and DBMS

Now it’s time to study the two most important computer science subjects, Operating Systems, and Database Management Systems. Most of the time junior engineers struggle with the OS and DBMS related questions and skip this part. But preparing these two subjects are also important for the interviews.

Go through the Basics Concepts of Operating System such as

1. Process and Process Management
2. Thread and Concurrency
3. Multithreading
4. Lock
5. Process Scheduling
6. Synchronization and Deadlock
7. CPU Scheduling
8. Distributed file system
9. Memory management
10. Critical section problem

* Also, prepare yourself with some database-related questions such as

1. relational and non-relational databases,
2. sharding,
3. scaling,
4. ACID property,
5. indexing, etc.

Resources

* Commonly Asked Operating Systems Interview Questions | Set 1
* Most asked Computer Science Subjects Interview Questions in Amazon, Microsoft, Flipkart
* GeeksforGeeks Operating System
* GeeksforGeeks DBMS

1. Day 93 to Day 99: Object-Oriented Design

Prepare yourself for an object-oriented design question for 7 days. You should have a good understanding of software design patterns, SOLID/DRY principles, and various oops concepts. Some object-oriented design questions are given below…

1. Design an ATM
2. Design Vending Machine
3. Design a Parking Lot
4. Design a Movie Ticket Booking System

* Your main goal is to gather all the requirements and transform that into comprehensible classes.
* You need to identify different objects and entities from a problem statement.
* In interviews, you will have to explain different components and their interface. Also, you will be explaining how different ***components*** are ***interacting*** ***with*** ***each*** ***other*** using the ***interfaces***.
* Prepare yourself with a use ***case*** ***diagram***, ***activity*** ***diagram***, and ***class*** ***diagram***.

1. Day 100: Cultural Fit

* Last day prepare yourself for the cultural round. Companies do not hire toxic people and also they don’t hire candidates who are not passionate about the product. Understand the culture of the company. For example, in Amazon culture is deeply rooted, and they conduct the bar raiser round. They care a lot about the leadership principle.
* In interviews, you will have to show interest in the product and you will have to be honest. Fake people won’t be entertained. So make sure that you keep in mind all these things and you prepare yourself with the right attitude.
  1. **100 Days of Code For Experienced**

Before you go ahead keep in mind that you need to focus on many things. You’re expected to perform well in the

1. coding round,
2. system design round and in the
3. other rounds as well.

You have the advantage to use the knowledge you have learned in your day-to-day job.

Sometimes it’s not the years of experience that matters for the interviewer but the time you spend sharpening your coding or technical skill matters for them.

So it’s not good to judge your skills on the basis of years of experience.

Track your knowledge, give some online tests, and check how far you still have to go.

1. Day 1 to Day 7: Choose a Programming Language and Learn The Fundamentals of it

Being an experienced developer it will be easy for you to choose a programming language and learn the fundamentals of it. You can try out a new language or you can choose a language you’re using in your day-to-day job. Once you make the decision go through all the basic concepts.

* Don’t think that you don’t need to go through the basics just because you’re using your language in your day-to-day job. Many times developers forget the basic concepts, or they miss out on a few things. So don’t skip this part and have a good command of your chosen language.

Few things you should remember are…

1. Reading/Writing from/to files.
2. Reading input from the console.
3. Split strings
4. 2D array implementation and how to use it.
5. Very basics things of language such as in C/C++ handling null-terminated strings

* A lot of companies such as Lyft and Salesforce have a slight difference in their hiring procedure. They want the candidate to write code and solve the problems on a laptop. You will have to write the complete program and your code should pass the given test cases.
* You may have to work on command line arguments or you may have to parse the CSV or text files.
* Other companies such as *Google*, *Amazon*, or *Microsoft* use a *whiteboard*. So it’s good that you know the *basics* of your *language* to handle both of the *coding* *scenarios* in your interview. Almost it will take 7 days to cover this part.

Resource

* C Programming Language
* C++ Programming Language
* Java Programming Language
* Python Programming Language

1. Day 8 to Day 14: Learn Data Structures and Algorithms

The experienced developer already uses *data* *structures* and *algorithms* in their days to day jobs (example: array is most widely used in various applications). They use their coding *logic* with *different* *data* *structures*, and *various* *algorithms* deepening on the *requirements*.

* Still, a lot of them forget the *basic* *implementation*, and they can not perform *simple* *tasks*. So it is better not to skip this part and *jump* *directly* into the *coding*.
* Go through all the fundamentals of data structures such as an array, linked list, queue, stack, tree, etc, and its implementation. Brush up on the basic operations performed on these data structures. Clear all the concepts of complexity, learn about the different algorithms, and their implementation.
* The basic foundation of coding i.e concepts of DSA must be strong to solve the different levels of coding questions. You can check your basic skill by giving some online tests.

Resource

* GeeksforGeeks Data Structures
* GeeksforGeeks Algorithms

1. Day 15 to Day 28: Solve Basic Level Problems

Once you brush up on the *data* *structure* and *algorithms*, practice for the basic level coding questions. With *experienced* *developers*, most of the time basic level questions are not asked in the *interviews* *round*. Even if it is then it’s just a kind of *warm-up problem*. During the *phone* *interview*, you can encounter these *kinds* of *problems*. So do not ignore this part and solve the basic questions first before you move to the complex one.

* Give *14 days* to *basic level* questions and try to solve almost *70-80 questions*. Depending on the time and speed you can increase the number of questions. Almost *5-6 questions* a day within the *timeframe* of *2-3 hours* is good enough to prepare you for the phone interviews, warm-up kind of problems, and for building a basic foundation as well.

Sample Array Problems

1. Merge Two Sorted Arrays
2. Subarray with given sum
3. Find duplicates in an array
4. Missing number in an array
5. Array Rotation

Sample Linked List Problems

1. Detect Loop in a linked list
2. Rotate a linked list
3. Merge two sorted linked list
4. Searching in the linked list
5. Find a middle element in a linked list

Sample Stack and Queue Problems

1. Sort values in Stack
2. Implement Two Stacks using one Array

Sample Tree Problems

1. Minimum value in Binary Search Tree
2. Height of a Binary Tree
3. kth maximum value in Binary Search Tree

Sample Graph Problems

1. Implement Breadth-First Search and Depth First Search

Sample Heap problems

1. Find k smallest and largest elements in a list
2. Day 29 to Day 49: Solve Medium Level Problems

After practicing the basic level coding questions move to the medium level questions and start solving it. Medium level problems will prepare you to solve harder problems and these questions are most likely asked in the interviews. Try will have to solve almost 120-130 medium level questions within 21 days. Depending on the time and speed you can increase the number of questions. Almost 5-6 questions a day within the timeframe of 2-3 hours is good to solve.

Sample Array Problems

1. Inversion of array
2. Maximum of all subarrays of size k
3. Trapping rainwater
4. Kadane’s algorithm

Sample Linked List Problems

1. First non-repeating character in a stream
2. Rotate a linked list
3. Remove loop in a linked list
4. LRU Cache
5. Clone a linked list
6. Find the Intersection point of two linked lists

Sample Stack and Queue Problems

1. Get minimum element from the stack
2. Queue using Two Stacks
3. Stack using Two Queues

Sample Tree Problems

1. Check for BST
2. Vertical traversal of binary tree
3. Boundary traversal of binary tree
4. Delete a node from BST

Sample Graph Problems

1. Detect Cycle in a directed and undirected graph
2. Find the number of islands
3. Word Boggle
4. Implementing Dijkstra
5. Day 50 to Day 70: Solve Hard Level Problems

* Once you solved the basic and medium level question, move to the complex or hard level problems, and solve it. Give it 21 days and try to solve almost 100-110 questions. Depending on the time and speed you can increase the number of questions.

Sample Problems

1. Find median in a stream
2. N-Queen problem
3. Sudoku problem
4. Traveling salesman problem
5. Boolean parenthesization

* Best Resource for Solving Coding Questions: GeeksforGeeks Practice Portal

1. Day 71 to Day 85: System Design

This section is more important for experienced developers to focus on. *System* *design* is the most *important* *rounds* in interviews specially for *senior* *engineers*. Experienced developers need to focus on this part more and in interviews, they will have to tell the approach to the interviewer to design a highly scalable system.

* Firstly learn all the basic concepts of system design such as *Load* *Balancing*, *Caching*, *Proxies*, *CAP* Theorem, *Consistency*, *Sharding* or *data* *partitioning*, etc. Once you’re done with basic concepts, prepare yourself for some most likely system design round questions such as…

1. URL shortening service (TinyURL)
2. Design Facebook Newsfeed
3. Design Netflix/Youtube
4. Design Uber or Lyft

* Preparing these questions will help you in handling this round and you will get to know what exactly you have to do in this round. You will be taking ownership of this round and you will describe the different components of the system. In this round, interviewers check that if the candidate is able to build a large and complex system or not.
* After *preparing* around *5-6 questions* you will have an idea that how *this round* looks *like*, what exactly you’re *supposed to do* and how you need to handle it.

Must Read:

1. 5 Common System Design Concepts for Interview Preparation
2. How to Crack System Design Round in Interviews
3. Top 10 System Design Interview Questions and Answers
4. Design Twitter
5. Design Dropbox
6. Day 86 to Day 92: OS and DBMS

Now it’s time to study the two most important computer science subjects, operating systems, and database management systems. Senior engineers are supposed to perform well when the concepts of these two subjects are asked in the interviews because they would be taking the help of multiple cores and threads while writing the code in their daily job.

In the operating system brush up on some basic concepts which you left a long time back during your undergrad program…

1. Process and Process management
2. Thread and Concurrency
3. Multithreading
4. Lock
5. Process Scheduling
6. Synchronization and Deadlock
7. CPU Scheduling
8. Distributed file system
9. Memory management
10. Critical section problem

* Also, prepare yourself with some database-related questions such as r

1. elational and non-relational databases,
2. sharding,
3. scaling,
4. ACID property,
5. indexing, etc.

Resources

* Commonly Asked Operating Systems Interview Questions | Set 1
* Most asked Computer Science Subjects Interview Questions in Amazon, Microsoft, Flipkart
* GeeksforGeeks Operating System
* GeeksforGeeks DBMS

1. Day 93 to Day 99: Object-Oriented Design

Prepare yourself for an object-oriented design question for 7 days. You should have a good understanding of software design patterns, SOLID/DRY principles, and various oops concepts. Some object-oriented design questions are given below…

1. Design an ATM
2. Design Vending Machine
3. Design a Parking Lot
4. Design a Movie Ticket Booking System

* Your main goal is to gather all the requirements and transform that into comprehensible classes. You need to identify different objects and entities from a problem statement.
* In interviews, you will have to explain different components and their interface. Also, you will be explaining how different components are interacting with each other using the interfaces. Prepare yourself with a use case diagram, activity diagram, and class diagram.

1. Day 100: Cultural Fit

Last day prepare yourself for the cultural round. Companies do not hire toxic people and also they don’t hire candidates who are not passionate about the product. Understand the culture of the company. For example, in Amazon culture is deeply rooted, and they conduct the bar raiser round. They care a lot about the leadership principle.

In interviews, you will have to show interest in the product and you will have to be honest. Fake people won’t be entertained. So make sure that you keep in mind all these things and you prepare yourself with the right attitude.

Geeksforgeeks Courses:

1. Language Foundation Courses [C++ / JAVA / Python ]

Learn any programming language from scratch and understand all its fundamentals concepts for a strong programming foundation in the easiest possible manner with help of GeeksforGeeks Language Foundation Courses – Java Foundation | Python Foundation | C++ Foundation

2. Geeks Classes Live

Get interview-centric live online classes on Data Structure and Algorithms from any geographical location to learn and master DSA concepts for enhancing your problem-solving & programming skills and to crack the interview of any product-based company – Geeks Classes: Live Session

3. Complete Interview Preparation

Get fulfilled all your interview preparation needs at a single place with the Complete Interview Preparation Course that provides you all the required stuff to prepare for any product-based, service-based, or start-up company at the most affordable prices.

4. DSA Self Paced

Start learning Data Structures and Algorithms to prepare for the interviews of top IT giants like Microsoft, Amazon, Adobe, etc. with DSA Self-Paced Course where you will get to learn and master DSA from basic to advanced level and that too at your own pace and convenience.

5. Company Specific Courses – Amazon, Microsoft, TCS & Wipro

Crack the interview of any product-based giant company by specifically preparing with the questions that these companies usually ask in their coding interview round. Refer GeeksforGeeks Company Specific Courses: Amazon SDE Test Series, etc.