

Competitive Programming – A Complete Guide

What is Competitive Programming and How to Prepare for It?

Fast I/O for Competitive Programming

Which C++ libraries are useful for competitive programming?

Input/Output from external file in C/C++, Java and Python for Competitive Programming

Tips and Tricks for Competitive Programmers | Set 1 (For Beginners)

Python Input Methods for Competitive Programming

C++ Methods of code shortening in competitive programming

Setting up a C++ Competitive Programming Environment

Basics of Array, String, Greedy and Bit

Number Theory and Combinatorics

Searching, Sorting and Basic Data Structure

Competitive Programming – A Complete Guide


Difficulty Level : Easy

🔖

✎

🔄

Competitive Programming is a mental sport which enables you to code a given problem under provided constraints. The purpose of this article is to guide every individual possessing a desire to excel in this sport. This article provides a detailed syllabus for Competitive Programming designed by industry experts to boost the preparation of the readers.




Related Courses

Start Your Coding Journey Now!

LoginRegister

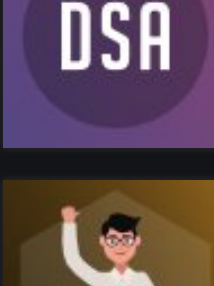
Get ready to level up your programming skills with this [Competitive Programming – Live Course](#). Learn the **Fundamentals of programming, DSA, Mathematical algorithms**, and much more. So, why wait? Dive into the world of Programming by enrolling in this course today!

WHAT'S NEW



Competitive Programming– Live Classes For Students


View Details



DSA

Data Structures & Algorithms– Self Paced Course

View Details



Complete Interview Preparation– Self Paced Course

View Details

Topic:

Introduction

What is Competitive Programming and How to Prepare for It?

Fast I/O: [C++](#), [Java](#), [Python](#)

Useful Libraries: [C++](#), [Java](#), [Python](#)

Input/Output Files: [Set 1](#), [Set 2](#)

Tips and Tricks: [Set 1](#), [Set 2](#)

Input Methods: [C++](#), [Java](#), [Python](#)

Template: [C++](#)

Language: [C++](#), [Java](#), [Python](#)

Time Complexity: [Analysis](#)

Setting up Competitive Programming Environment: [Sublime](#): [C++](#), [Visual Studio](#): [C++](#) and [Python](#)

Basics Of Array , String, Greedy and Bit Manipulation

Reverse an array (Related Problems: [Problem 1](#), [Problem 2](#))

[Sum of Digits](#)

[Program to Check if a Given String is Palindrome in C, Python](#) (Related [Problem](#))

[Sum of array elements](#) (Related [Problem](#))

[Maximum and Minimum element of array](#) (Related [Problem](#))

[Counting frequencies of array elements](#) (Related Problems: [Problem 1](#), [Problem 2](#))

Float and Precision: [C++](#), [Java](#), [Python](#)

[Prefix sum, 2D Prefix Sum Difference Array, I Range update query in O\(1\)](#): (Related Problems: [Problem 1](#), [Problem 2](#))

[Coordinate Compression](#): (Related [Problem](#))

[Kadane Algorithm](#): (Related [Problem](#))

[Activity Selection Problem](#): (Related [Problem](#))

[Job Sequencing Problem](#): (Related [Problem](#))

[Sliding Window](#): (Related [Problem](#))

Logical Operators: [C++ Set 1](#), [Set 2](#), [Java](#), [Python](#)

Bit Manipulation: [Set 1](#), [Set 2](#), [Set 3](#) (Related Problems: [Problem 1](#), [Problem 2](#), [Problem 3](#))

[Bitset C++](#)

<https://www.geeksforgeeks.org/top-50-array-coding-problems-for-interviews/>

<https://www.geeksforgeeks.org/top-50-string-coding-problems-for-interviews/>

Number Theory and Combinatorics

[Prime Number](#) (Related [Problem](#))

[Sieve of Eratosthenes](#) (Related [Problem](#))

[Segmented Sieve](#) (Related [Problem](#))

[Find all divisors of a natural number](#) (Related [Problem](#))

[Least prime factor of numbers upto N](#) (Related [Problem](#))

[All prime factors of a number](#) (Related [Problem](#))

[Prime Factorization using Sieve O\(log n\) for multiple queries](#)

[Sum of all factors of a number](#) (Related [Problem](#))

[Gcd of Two numbers, Lcm of two numbers](#) (Related [Problem](#))

[Linear Diophantine Equations](#)

[Euclidean algorithms \(Basic and Extended\)](#)

[Euler's Totient Function](#) (Related [Problem](#))

[Euler's Totient function for all numbers smaller than or equal to n](#)

[Inclusion Exclusion Principle](#)

[Pigeon Hole Principle](#)

[Modular Operations](#)

[Modular Inverse](#): (Related [Problem 1](#), [Problem 2](#))

[Chinese Remainder Theorem](#): [Set 1](#), [Set 2](#)

[Power\(x,y\) in O\(logN\)](#)

[Power\(x,y\) % mod](#): (Related [Problem 1](#), [Problem 2](#))

[Matrix Exponentiation](#): (Related [Problem](#))

[Permutation and Combination](#): [Set 1](#), [Set 2](#), [Quiz 1](#), [Quiz 2](#)

nCr: [Set 1](#), [Set 2](#)

nCr % mod: [Set 1](#), [Set 2](#): (Related [Problem](#))

[nCr % mod for multiple queries](#): (Related [Problem](#))

[Catalan numbers: Applications](#) and Related [Problem](#)

[Gaussian Elimination](#)

Searching, Sorting and Basic Data Structures

[Linear Search](#) (Related Problems : [Problem 1](#), [Problem 2](#))

[Binary Search](#), [Unbounded Binary Search](#) (Related Problems : [Problem 1](#), [Problem 2](#), [Problem 3](#))

[Inbuilt sorting O\(logN\)](#): [C++](#), [Java](#), [Python](#) (Related Problems : [Problem 1](#), [Problem 2](#), [Problem 3](#), [Problem 4](#))

[Merge Sort](#) (Related Problems : [Problem 1](#), [Problem 2](#))

[Quick Sort](#) (Related Problems : [Problem](#))

Stack: Implementation in [C++](#), [Java](#), [Python](#) (Related Problems : [Problem 1](#), [Problem 2](#), [Problem 3](#))

Queue: Implementation in [C++](#), [Java](#), [Python](#) (Related Problems : [Problem 1](#), [Problem 2](#), [Problem 3](#))

Deque: Implementation in [C++](#), [Java](#), [Python](#) (Related Problems : [Problem](#))

Priority Queue: Implementation in [C++](#), [Java](#), [Python](#) (Related Problems : [Problem 1](#), [Problem 2](#), [Problem 3](#))

Tree and Graphs

[Tree BFS, Tree DFS](#) (Related Problems : [Problem 1](#), [Problem 2](#), [Problem 3](#))

[Graph BFS, Graph DFS 2, Graph DFS](#) (Related Problems : [Problem 1](#), [Problem 2](#))

[Dijkstra's Shortest Path Algorithm](#) (Related Problems : [Problem 1](#), [Problem 2](#))

[Bellman – Ford Algorithm](#) (Related [Problem](#))

[Floyd Warshall Algorithm](#) (Related [Problem](#))

[0-1 BFS, Dial's Algorithm](#)

Detect cycle: [Directed](#), [Undirected](#) (Related Problems : [Problem 1](#), [Problem 2](#))

Disjoint set(union-find): [Set 1](#), [Set 2](#), [Set 3](#) (Related [Problem](#))

[Topological Sorting, Kahn's Algorithm](#) (Related [Problem](#))

Minimum Spanning Tree: [Prim's Algorithm](#), [Kruskal Algorithm](#) (Related [Problem](#))

[Bipartite or not, M-Coloring](#) (Related Problems : [Problem 1](#), [Problem 2](#), [Problem 3](#))

Strongly Connected Components: [Tarjan](#), [Kosaraju](#) (Related Problems : [Problem 1](#), [Problem 2](#))

Euler Path: [Undirected](#), [Directed](#) (Related [Problem](#))

Flow Algorithms: [Set 1](#), [Set 2](#), [Dinic's Algorithm](#) (Related Problems : [Problem 1](#), [Problem 2](#))

[Diameter of Tree](#)

[Centroid Decomposition](#)

[Lowest Common Ancestor](#)

<https://www.geeksforgeeks.org/top-50-tree-coding-problems-for-interviews/>

Recursion and Dynamic Programming

[Recursion](#): [Quiz 1](#), [Quiz 2](#), [Quiz 3](#), [Quiz 4](#), [Quiz 5](#), [Quiz 6](#), [Quiz 7](#) (Related Problems : [Problem 1](#), [Problem 2](#), [Problem 3](#))

[Backtracking](#): (Related Problems : [Problem 1](#), [Problem 2](#))

Dp Introduction: [Set 1](#), [Set 2](#), [Set 3](#), [Set 4](#), [Set 5](#)

[Most useful Dynamic Programming questions](#)

Additional DP Problems : [Problem 1](#), [Problem 2](#), [Problem 3](#), [Problem 4](#)

Dp on Trees: [Set 1](#), [Set 2](#)

Dp on Bit Masking: [Set 1](#), [Set 2](#), [Set 3](#)

[Digit Dp](#)

<https://www.geeksforgeeks.org/top-50-dynamic-programming-coding-problems-for-interviews/>

[Suffix Tree](#): [Set 1](#), [Set 2](#)

[Z Algorithm](#)

[KMP Algorithm, Rabin-Karp Algorithm](#) (Related [Problem](#))

[Manacher's Algorithm](#): [Set 1](#), [Set 2](#), [Set 3](#), [Set 4](#)

[Suffix Automation](#): [Set 1](#), [Set 2](#)

Geometry and Game Theory

[Closest Pair of Points](#)

[How to check if two given line segments intersect?](#) (Related [Problem](#))

[How to check if a given point lies inside or outside a polygon?](#)

Convex Hull: [Set 1](#), [Set 2](#) (Related [Problem](#))

[Given n line segments, find if any two segments intersect](#)

[Check whether a given point lies inside a triangle or not](#)

[How to check if given four points form a square](#): (Related [Problem](#))

Combinatorial Game Theory: [Set 1](#), [Set 2](#), [Set 3](#), [Set 4](#)

Minimax Algorithm in Game Theory: [Set 1](#), [Set 2](#), [Set 3](#), [Set 4](#), [Set 5](#)

[Variation in Nim Game](#)

[Find the winner in nim-game](#)

[Optimal Strategy for a Game](#)

Advance Data Structures

[Trie](#): [Set 1](#), [Set 2](#), [Set 3](#), (Related Problems: [Problem 1](#), [Problem 2](#), [Problem 3](#), [Problem 4](#), [Problem 5](#))

[Fenwick Tree](#): [Set 1](#), [Set 2](#), [Set 3](#), [Set 4](#), (Related [Problem](#))

[Segment Tree](#): [Set 1](#), [Set 2](#), [Set 3](#) (Related [Problem](#))

[Sparse Table](#): [Set 1](#), [Set 2](#)

[Sqrt Decomposition](#): [Set 1](#), [Set 2](#)

[Heavy Light Decomposition](#): [Set 1](#), [Set 2](#)

[Meet in the Middle](#)

[MO's Algorithm, Problem](#)

[Policy based Data Structure](#)

You may also check [Geeksforgeeks Online Courses](#) to Learn Data Structures and Algorithms, well designed courses taught by Industry Experts.

GeeksforGeeks Courses

Competitive Programming – Live Course

Get ready to take your programming skills to the next level? This [Competitive Programming – Live Course](#) will help you enhance your problem-solving skills to be a programmer for a top company and gain a competitive edge over other candidates in SDE interviews. Learn **Basics of programming, Data structure and algorithms, Efficient implementation of mathematical algorithms** and much more. Then, why wait? Take your first step towards becoming a better programmer, see you in the course!

DSA Self Paced

Master Data Structures and Algorithms, trusted by over 75,000 students! Prepare for the interviews with leading IT giants like Microsoft, Amazon, Adobe, etc. Built with years of experience by top industry experts and gives you a complete package of video lectures, practise problems, quizzes, discussion forums and contests. Learn and master DSA at the most affordable price possible with GeeksforGeeks [DSA Self-Paced Course](#). Join Today!

Language Foundation Courses[ [C Programming](#) / [C++](#) / [JAVA](#) / [Python](#) ]

Master any programming language from scratch and understand all its core fundamental concepts for a strong programming foundation at budget-friendly prices with help of GeeksforGeeks Language Foundation Courses - [C Programming](#) | [Java Foundation](#) | [Python Foundation](#) | [C++ Foundation](#). These courses are for complete beginners who want to get started with programming and build their foundations. Start your coding journey today!.

Exclusive Hiring Challenge For SDE & SDE 2

GeeksforGeeks

Partnered with

amazon

Amazon and the Amazon logo are trademarks of Amazon.com, Inc. or its affiliates

👍 Like 364

Previous

Combinatorial Game Theory | Set 1 (Introduction)

Next

Trie | (Insert and Search)

RECOMMENDED ARTICLES

Page: 1 2 3

01

Tips and Tricks for Competitive Programmers | Set 2 (Language to be used for Competitive Programming)

20, Mar 16

02

Top Programming Languages For Competitive Programming

10, Jul 22

03

How to become a master in competitive programming?

17, Feb 16

04

Competitive Programming: Conquering a given problem

28, Mar 16

05

A Better Way To Approach Competitive Programming

06, Apr 16

06

getchar\_unlocked() – Faster Input in C/C++ For Competitive Programming

18, Apr 16

07

Frequency Measuring Techniques for Competitive Programming


03, Sep 18

08

Input/Output from external file in C/C++, Java and Python for Competitive Programming

24, Feb 17

● ● ●

Article Contributed By :  GeeksforGeeks

Vote for difficulty

Current difficulty : [Easy](#)

EasyNormalMediumHardExpert


Improved By : varshagumber26, raj2002

Article Tags : Competitive Programming

Improve ArticleReport Issue

Writing code in comments? Please use [@geeksforgeeks.org](#), generate link and share the link here.

Load Comments

 **GeeksforGeeks**

📍 A-143, 9th Floor, Sovereign Corporate Tower, Sector-136, Noida, Uttar Pradesh – 201305

✉️ [feedback@geeksforgeeks.org](mailto:feedback@geeksforgeeks.org)

🌐

📷

🌐

🐦

📺

📺

Company

About Us

Careers

In Media

Contact Us

Privacy Policy

Copyright Policy

Learn

Algorithms

Data Structures

SDE Cheat Sheet

Machine Learning

CS Subjects

Video Tutorials

Courses

News

Top News

Technology

Work & Career

Business

Finance

Lifestyle

Knowledge

Languages

Python

Java

Golang

C#

SQL

Kotlin

Web Development

Web Tutorials

Django Tutorial

HTML

JavaScript

Bootstrap

ReactJS

NodeJS

Contribute

Write an Article

Improve an Article

Pick Topics to Write

Write Interview Experience

Internships

Video Internship

@geeksforgeeks, Some rights reserved