**Just follow my guidelines.**

* Download latest JDK from Oracle website: [Java SE - Downloads](http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html)
* Start reading Java from Oracle’s Tutorial: [The Java™ Tutorials](https://docs.oracle.com/javase/tutorial/)
* This is good article to run a Java code in 10 minutes: [Learn Java in 10 Minutes](http://java.sys-con.com/node/2501666)

Once you are comfortable with JDK and basic Java, search on web study these topics:

**Self Study Learning Part #1 :Java Concepts**

* Java Class Design
* Java Methods
* Java Operators and Assignments (precedence as well)
* Java Conditional Statements
* Java Loops (for and enhanced for loop, while, do/while, switch)
* Exception Handling
* Java Threads (it may be useful in your application)
* Java arrays and Java Collections (useful for swap/search)

The better way to improve Java programming skill is by Oracle’s Java programmer certification preparation. You no need to attend exam, but the preparation will shape-up your programming skills.

* [Choose Your Java Certification From Oracle](http://javacertification.net/)
* [Java Certifications MyExamCloud Exam Collections](http://www.myexamcloud.com/onlineexam/collection/javacertification)

**Self Study Learning Part #2 : Improve your Java program algorithm in the following areas:**

1) **Graph algorithms:**Breadth first search(BFS), Depth first search(DFS), Strongly connected components(SCC), Dijkstra, Floyd-Warshall, Minimum spanning tree(MST), Topological sort.

2) **Dynamic programming:**Standard dynamic programming problems such as Rod Cutting, Knapsack, Matrix chain multiplication etc.

3) **Number theory:**Modular arithmetic, Fermat’s theorem, Chinese remainder theorem(CRT), Euclidian method for GCD, Logarithmic

Exponentiation, Sieve of Eratosthenes, Euler’s totient function.

3) **Greedy:**Standard problems such as Activity selection.

4) **Search techniques:** Binary search, Ternary search and Meet in the middle.

5) **Data structures (Basic):**Stacks, Queues, Trees and Heaps.

6) **Data structures (Advanced):**Trie, Segment trees, Fenwick tree or Binary indexed tree(BIT), Disjoint data structures.

7) **Strings:**Knuth Morris Pratt(KMP), Z algorithm, Suffix arrays/Suffix trees. These are bit advanced algorithms.

8) **Computational geometry:**Graham-Scan for convex hull, Line sweep.

9) **Game theory:**Basic principles of Nim game, Grundy numbers, Sprague-Grundy theorem.

Once you have sufficient knowledge of popular algorithms, you can start solving the medium level problems.

Learning to code is all about practicing. Participate regularly in the Java programming contests. Solve the ones that you cannot solve in the contest, after the contest. Apart from Topcoder and Codeforces you can also look at HackerEarth Challengesor Codechef contests.

Programming is a very practical and hands on skill. You have to continuously do it to be good at it. It’s not enough to solve the problem theoretically, you have to code it and get the solution accepted. Knowing which algorithm/logic to use and implementing it are two different things. It takes both to be good at programming.

**Self Study Learning Part #3: Study Advanced Java Topics**

Read my article: [Ganesh P's answer to What should I learn next after basic Java? I’ve learned the basics of the language and still need to learn some types of containers. Is it a good time for me to learn design patters? If so, which are the most important ones?](https://www.quora.com/What-should-I-learn-next-after-basic-Java-I%E2%80%99ve-learned-the-basics-of-the-language-and-still-need-to-learn-some-types-of-containers-Is-it-a-good-time-for-me-to-learn-design-patters-If-so-which-are-the-most-important-ones/answer/Ganesh-P-9)