# DSA TIPS

DSA can be seen as three step process, A. Learn a language, B. Learn a data structure, C. Apply it in algorithms. Let's dig in shall we?

Learning a language is THE MOST asked question so, here's an answer. What language do I pick, short answer is it depends, the long of it is this. You have primarily three options,

C/C++

Java

Python

What to pick and why? Simply speaking, most programming languages have the same skeleton just different ways of doing it. Therefore, they are used in different ways.

C++ is a powerful language which gives most of the power in your hands, this includes handling how much power or resource you would demand from a system. What kind of system memory you wish to utilise. If you have a timeline of over a year or more this is the best option for you. Though this has a long learning curve and does take time, it helps a lot in the future since almost every other language on this list has features taken from C++. Lastly, if you wish to join Competitive Programming this is the way to go.

Java is a language that falls somewhere in the middle where it's not too easy but not too frustrating as well. Most service based companies who are in the Asian belt prefer Java as a language in many of their projects. Bearing that in mind. If you're aiming for a job in a service based firm this is the way to go. The learning curve is a bit less than C++ but again the concepts are quite empowering for a programmer in their career.

Python is the kingpin that runs the Wild West that's IT these days, Python would be the ideal choice if you're a short deadline and need a job ASAP. The concepts are fewer and the language is less verbose.

Resources to learn the languages:

For C++

1. First two lectures on Harvard's CS50 (David.J.Malan is one of the best explainer of Intro To Programming I've ever seen)
2. W3Schools [www.w3schools.com](https://www.w3schools.com/cpp/default.asp) (Works great as a course material and reference)
3. [www.learncpp.com](https://www.learncpp.com/) (Frankly, the best place to learn imho)These should be enough for learning C++

For Java

1. [Derek Banas](https://www.youtube.com/playlist?list=PLE7E8B7F4856C9B19) has a fantastic tutorial on YT. [Tim Bulchaka’s Java Masterclass](https://www.udemy.com/course/java-the-complete-java-developer-course/) course (Paid Resource)[Coding With John](https://www.youtube.com/playlist?list=PLkeaG1zpPTHiMjczpmZ6ALd46VjjiQJ_8) is another fabulous resource.

For Python

[Corey Schafer](https://www.youtube.com/playlist?app=desktop&list=PL-osiE80TeTt2d9bfVyTiXJA-UTHn6WwU) on YT

[W3Schools](https://www.w3schools.com/python/default.asp) (Works great as a course material and reference)

[The Python Tutorial on Python.Org](https://wiki.python.org/moin/BeginnersGuide) is a good reference to work as a pair with any of the above listed resources.

So, what all must you know from a language agnostic view point,The basics - Variables, if-else, strings, loops, functions.OOPs (Object Oriented Programming) - Classes, Methods, instances, etc.

At this stage you can move on to learning Data Structures, I'll be listing the most common ones and what approaches are necessary. This is not an exhaustive list nor it is a rulebook for solving problems.

Tweak and learn as per your need and adapt.

I would suggest to go through these data structures.

Linked lists

Stacks

Queues

Trees

Graphs

Heaps

Hash tables

These would allow you to clear any interview or start solving competitive programming problems.

A playlist that helped me a lot for data structures was [William Fiset's](https://www.youtube.com/watch?v=RBSGKlAvoiM) video.

If you have taken Java as a language [Princeton University's Algorithms](https://www.coursera.org/learn/algorithms-part1) would be the go-to resource.

[Tech Interview Handbook](https://www.techinterviewhandbook.org/) is another resource that would be helpful.

[Abdul Bari](https://www.youtube.com/playlist?list=PLDN4rrl48XKpZkf03iYFl-O29szjTrs_O) is a fantastic resource for Algorithms. His explanations are top tier.

Though in JavaScript [this resource](https://www.youtube.com/playlist?list=PLC3y8-rFHvwjPxNAKvZpdnsr41E0fCMMP) for it's logic explanations are great.

Simultaneous to this would be recommended to solve problems from sites such as HackerRank, LeetCode.

In the case for LeetCode go for Easy Problems at first then go to medium problems. Hard Problems are better suited for Competitive Problems only.

The way I used to solve problems was this, I set a timer of 20 minutes and read the problem trying to solve the problem. After the 20 mins were over, regardless of if I had solved the problem or not reading through the editorials or looking through on Google for solutions helped me see methods or logic I hadn't thought of before.

Form a habit of solving at least 2 problems a day, which helps your mind work everyday and allow you to go.

Some Tips:

Getting an error is the rule, the program running perfect is the exception. This is a mindset which would allow you to get over the hesitation of feeling incompetent and giving up. StackOverFlow, Reddit and other such resources have millions of people solving, asking problems. Which simply means you're not alone.

You can always edit bad code, a blank page is depressing anyway, Write the code once you've got a solution. You can then edit it and make it better. Writing on paper is also a great habit to have.

The better programmer keeps going one more time than the person before them.

Even the greatest programmer today once didn't know how to declare a variable.

Good luck!

SOURCE : reddit