



Introduction to Data Structures and Algorithms

[< Previous](#)[Next >](#)

Data Structures is about how data can be stored in different structures.

Algorithms is about how to solve different problems, often by searching through and manipulating data structures.

Theory about Data Structures and Algorithms (DSA) helps us to use large amounts of data to solve problems efficiently.

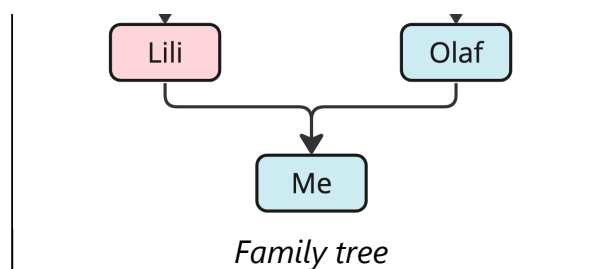
What are Data Structures?

A data structure is a way to store data.

We structure data in different ways depending on what data we have, and what we want to do with it.

First, let's consider an example without computers in mind, just to get the idea.

because we have information about people we are related to and how they are related, and we want an overview so that we can easily find a specific family member, several generations back.



With such a family tree data structure visually in front of you, it is easy to see, for example, who my mother's mother is—it is 'Emma,' right? But without the links from child to parents that this data structure provides, it would be difficult to determine how the individuals are related.

Data structures give us the possibility to manage large amounts of data efficiently for uses such as large databases and internet indexing services.

Data structures are essential ingredients in creating fast and powerful algorithms. They help in managing and organizing data, reduce complexity, and increase efficiency.

In Computer Science there are two different kinds of data structures.

Primitive Data Structures are basic data structures provided by programming languages to represent single values, such as integers, floating-point numbers, characters, and booleans.

Abstract Data Structures are higher-level data structures that are built using primitive data types and provide more complex and specialized operations. Some common examples of abstract data structures include arrays, linked lists, stacks, queues, trees, and graphs.

What are Algorithms?

An algorithm is a set of step-by-step instructions to solve a given problem or achieve a specific goal.

A cooking recipe written on a piece of paper is an example of an algorithm, where the goal is to make a certain dinner. The steps needed to make a specific dinner are described exactly.

of food ingredients, an algorithm uses data structures.

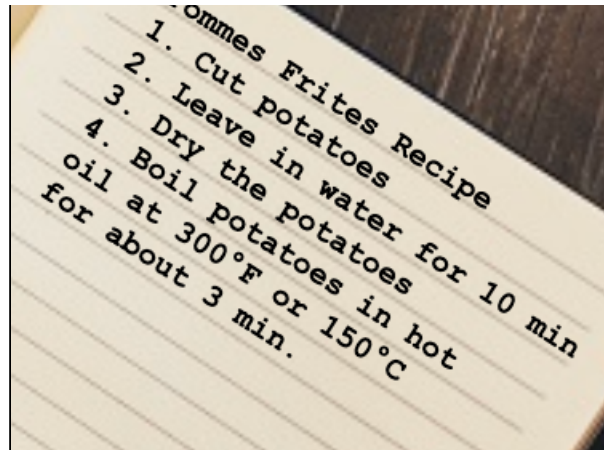
Algorithms are fundamental to computer programming as they provide step-by-step instructions for executing tasks. An efficient algorithm can help us to find the solution we are looking for, and to transform a slow program into a faster one.

By studying algorithms, developers can write better programs.

Algorithm examples:

- Finding the fastest route in a GPS navigation system
- Navigating an airplane or a car (cruise control)
- Finding what users search for (search engine)
- Sorting, for example sorting movies by rating

The algorithms we will look at in this tutorial are designed to solve specific problems, and are often made to work on specific data structures. For example, the 'Bubble Sort' algorithm is designed to sort values, and is made to work on arrays.



Pommes Frites Recipe

Data Structures together with Algorithms

Data structures and algorithms (DSA) go hand in hand. A data structure is not worth much if you cannot search through it or manipulate it efficiently using algorithms, and the algorithms in this tutorial are not worth much without a data structure to work on.

DSA is about finding efficient ways to store and retrieve data, to perform operations on data, and to solve specific problems.

By understanding DSA, you can:

- Decide which data structure or algorithm is best for a given situation.
- Make programs that run faster or use less memory.
- Understand how to approach complex problems and solve them in a systematic way.



Needed?

Data Structures and Algorithms (DSA) are used in virtually every software system, from operating systems to web applications:

- For managing large amounts of data, such as in a social network or a search engine.
- For scheduling tasks, to decide which task a computer should do first.
- For planning routes, like in a GPS system to find the shortest path from A to B.
- For optimizing processes, such as arranging tasks so they can be completed as quickly as possible.
- For solving complex problems: From finding the best way to pack a truck to making a computer 'learn' from data.

DSA is fundamental in nearly every part of the software world:

- Operating Systems
- Database Systems
- Web Applications
- Machine Learning
- Video Games
- Cryptographic Systems
- Data Analysis
- Search Engines

Theory and Terminology

As we go along in this tutorial, new theoretical concepts and terminology (new words) will be needed so that we can better understand the data structures and algorithms we will be



Term	Description
Algorithm	A set of step-by-step instructions to solve a specific problem.
Data Structure	A way of organizing data so it can be used efficiently. Common data structures include arrays, linked lists, and binary trees.
Time Complexity	A measure of the amount of time an algorithm takes to run, depending on the amount of data the algorithm is working on.
Space Complexity	A measure of the amount of memory an algorithm uses, depending on the amount of data the algorithm is working on.
Big O Notation	A mathematical notation that describes the limiting behavior of a function when the argument tends towards a particular value or infinity. Used in this tutorial to describe the time complexity of an algorithm.
Recursion	A programming technique where a function calls itself.
Divide and Conquer	A method of solving complex problems by breaking them into smaller, more manageable sub-problems, solving the sub-problems, and combining the solutions. Recursion is often used when using this method in an algorithm.
Brute Force	A simple and straight forward way an algorithm can work by simply trying all possible solutions and then choosing the best one.

Where to Start?

In this tutorial, you will first learn about a data structure with matching algorithms, before moving on to the next data structure.

Further into the tutorial the concepts become more complex, and it is therefore a good idea to learn DSA by doing the tutorial step-by-step from the start.

And as mentioned on the previous page, you should be comfortable in at least one of the most common programming languages, like for example [JavaScript](#), [C](#) or [Python](#), before doing this tutorial.

[Tutorials ▼](#)[References ▼](#)[Exercises ▼](#)[Sign In](#)[≡](#) [SS](#) [JAVASCRIPT](#) [SQL](#) [PYTHON](#) [JAVA](#) [PHP](#) [HOW TO](#) [W3.CSS](#) [C](#) [C](#)

Click the 'Next' button to continue.

[< Previous](#)[Sign in to track progress](#)[Next >](#)

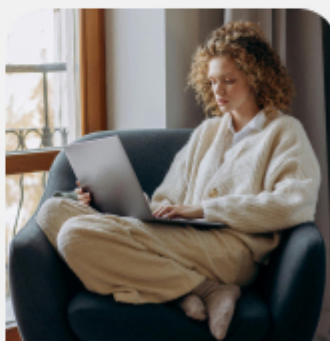
Get Certified!

Document your skills with all of
W3Schools Certificates

~~\$1,995~~

\$499

Save 75% 🎁



COLOR PICKER



[Tutorials ▼](#)[References ▼](#)[Exercises ▼](#)[Sign In](#)[SS](#)[JAVASCRIPT](#)[SQL](#)[PYTHON](#)[JAVA](#)[PHP](#)[HOW TO](#)[W3.CSS](#)[C](#)[C](#)[GET CERTIFIED](#)[FOR TEACHERS](#)[FOR BUSINESS](#)[CONTACT US](#)

Top Tutorials

- [HTML Tutorial](#)
- [CSS Tutorial](#)
- [JavaScript Tutorial](#)
- [How To Tutorial](#)
- [SQL Tutorial](#)
- [Python Tutorial](#)
- [W3.CSS Tutorial](#)
- [Bootstrap Tutorial](#)
- [PHP Tutorial](#)
- [Java Tutorial](#)
- [C++ Tutorial](#)
- [jQuery Tutorial](#)

Top References

- [HTML Reference](#)
- [CSS Reference](#)
- [JavaScript Reference](#)
- [SQL Reference](#)
- [Python Reference](#)
- [W3.CSS Reference](#)
- [Bootstrap Reference](#)
- [PHP Reference](#)
- [HTML Colors](#)
- [Java Reference](#)
- [AngularJS Reference](#)
- [jQuery Reference](#)

Top Examples

- [HTML Examples](#)
- [CSS Examples](#)
- [JavaScript Examples](#)
- [How To Examples](#)
- [SQL Examples](#)
- [Python Examples](#)
- [W3.CSS Examples](#)
- [Bootstrap Examples](#)
- [PHP Examples](#)
- [Java Examples](#)
- [XML Examples](#)
- [jQuery Examples](#)

Get Certified

- [HTML Certificate](#)
- [CSS Certificate](#)
- [JavaScript Certificate](#)
- [Front End Certificate](#)
- [SQL Certificate](#)
- [Python Certificate](#)
- [PHP Certificate](#)
- [jQuery Certificate](#)
- [Java Certificate](#)
- [C++ Certificate](#)
- [C# Certificate](#)
- [XML Certificate](#)

[Tutorials ▼](#)[References ▼](#)[Exercises ▼](#)[Sign In](#)[≡](#) [SS](#) [JAVASCRIPT](#) [SQL](#) [PYTHON](#) [JAVA](#) [PHP](#) [HOW TO](#) [W3.CSS](#) [C](#) [C](#)[FORUM](#) [ABOUT](#) [ACADEMY](#)

W3Schools is optimized for learning and training. Examples might be simplified to improve reading and learning.

Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness

of all content. While using W3Schools, you agree to have read and accepted our [terms of use](#), [cookies](#) and [privacy policy](#).

[Copyright 1999-2026](#) by Refsnes Data. All Rights Reserved. [W3Schools](#) is Powered by [W3.CSS](#).