

# Advanced Data Structures

## B-Tree, Splay, Priority Queue, Complexity, Searching, Sorting, Hashing, Algorithms, Implementation

In this specialist course we examine modern data structures from a mathematical viewpoint.

We explore how to correctly design modern data structures, how to implement the algorithms that interact with them, memory management, traversal, item searching and sorting, merging and splitting, and lots more. We use both imperative and functional programming styles. We also cover the wide range of proven data structure layouts already used in industry - each with its own capabilities, challenges and recommended usage domains.

We are particularly interested in where data structures and mathematics meet. For example, this course explores order theory and complexity theory and brings the richness of certain mathematical structures to common programming libraries (e.g. mathematical set has extra useful features that are not present in common implementations of sets in various runtimes). We also pay attention to correct usage of terminology to avoid misnaming – e.g. we see in C++ STL use of the term “vector” for something that is not a mathematical vector (an STL vector is a mathematical sequence [dynamic array]) - [it is now too late to correct that misnaming](#).

| <b>Contents of One-Day Training Course</b>                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>Target Audience</b><br/>This course is aimed at mathematicians and developers who wish to gain a richer understanding of modern approaches to data structures.</p> <p><b>Prerequisites</b><br/>Attendees need a good background in mathematics and programming.</p> | <p><b>Goals of Data Structures</b><br/>Designing data structure the right way<br/>Important considerations (future evolution, performance, compression, ..)<br/>Accessibility (random vs. sequential)<br/>Visibility (public vs. internal functionality)<br/>Complexity and Big O notation</p> <p><b>Common Data Structures</b><br/>Review of common data structures from a mathematical viewpoint<br/>Mathematical sequence = array<br/>Mathematical set = set<br/>Mathematical morphism = map (sometimes called dictionary)<br/>How best to implement</p> <p><b>Additional Data Structures</b><br/>Linked list (how to get same performance from singly vs. doubly linked list)<br/>FIFO queue (stack)<br/>LIFO queue<br/>Does each data structure need to be independently created, or can it be layered on top of another (adaptors)?</p> <p><b>B-Tree</b><br/>B-Tree is the most important data structure for storage architectures<br/>Performance characteristics of B-Trees<br/>Variations of B-Tree layout</p> <p><b>Splay Tree</b><br/>Introduction to binary search tree<br/>A splay tree is a binary search tree<br/>Recently used items quick to re-access<br/>The splaying operation</p> |
|                                                                                                                                                                                                                                                                           | <p><b>Priority Queue</b><br/>Priority for an item<br/>Sorting on that</p> <p><b>Hashing</b><br/>Hashing plays a particular importance in many searching and sorting algorithms<br/>Examine from first principles how efficient hashing works</p> <p><b>Common Operations</b><br/>Searching<br/>Sorting<br/>Merging<br/>Splitting<br/>Parallel access</p> <p><b>Data Structures For Functional Programming</b><br/>There are particular considerations to designing data structures for use with functional programming languages<br/>Immutability and versioning</p> <p><b>Algorithms</b><br/>Structure independent algorithms<br/>Applying algorithms in a uniform way<br/>Use of iterators to traverse structures</p> <p><b>Memory</b><br/>For data structures that grow and contract over time, need good memory management<br/>Memory management and allocators<br/>Slab allocators</p> <p><b>Project</b><br/>Creating a custom graph engine to store and interact with large-scale graphs efficiently</p>                                                                                                                                                                                         |