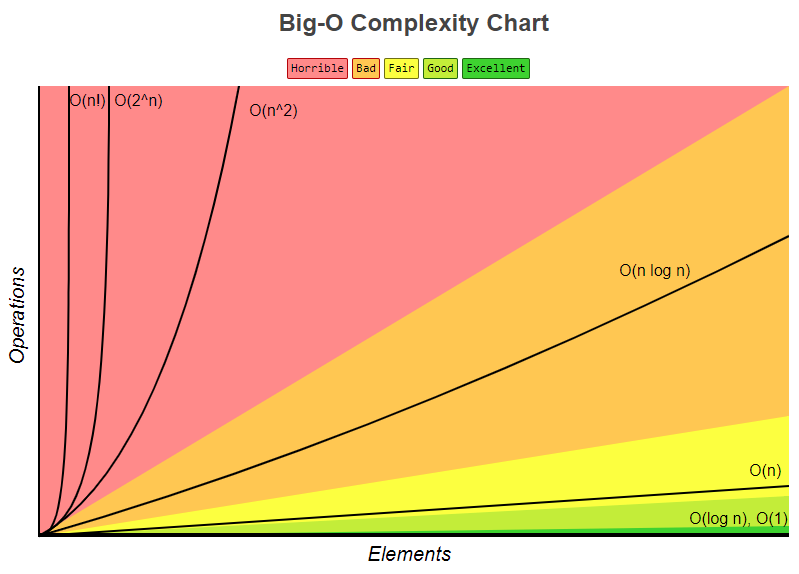
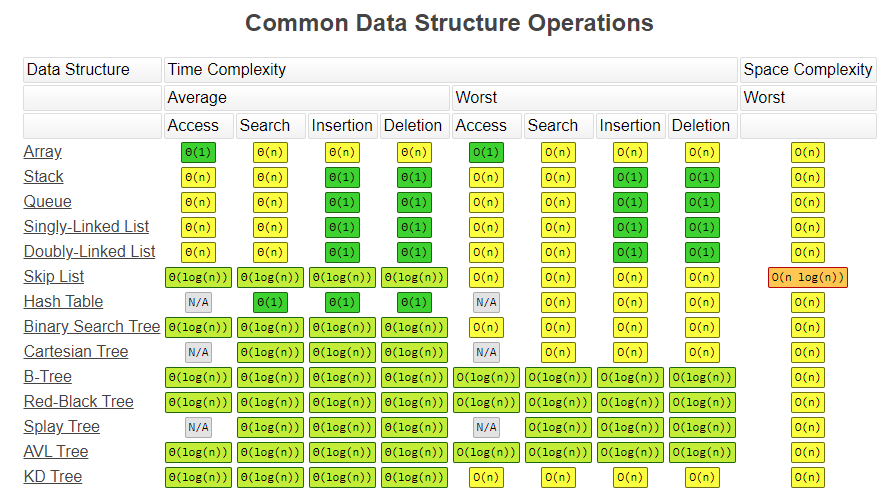
# Data Structures

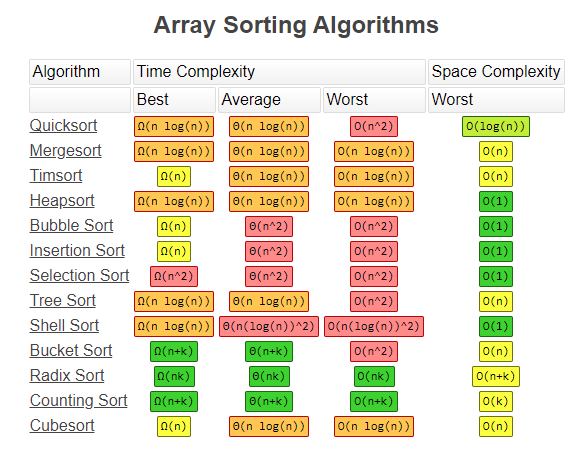
|  |  |
| --- | --- |
| Arrays |  |
| Stacks |  |
| Queues |  |
| Linked Lists |  |
| Trees |  |
| Tries |  |
| Graphs | Directed or Undirected Cyclic or Acyclic  Weighted or Unweighted |
| Hash Tables | O(1) lookup |

## Big-O



|  |  |  |
| --- | --- | --- |
| O(1) | Constant – no loops | Rule 1: Always worst case |
| O(log n) | Logarithmic – searching sorted data | Rule 2: Remove constants |
| O(n) | Linear – for loops and while loops | Rule 3: Different inputs should |
| O(n \* log(n)) | Log Linear – sorting operations | Have different variables. |
| O(n^2) | Quadratic – nested loops | O(n + m) or O(n \* m) |
| O(2^n) | Exponential – recursive algorithms | Rule 4: Drop non-dominant |
| O(n!) | Factorial – a loop for every element | terms |





# Algorithms

## Sorting

## Dynamic Programming

## Depth First Search (DFS)

## Breadth First Search (BFS)

## Recursion

Used for repeating sub-tasks

Key is setting up the base case

# Object Oriented Programming and Patterns

\* Encapsulation - Reduce complexity and increase reusability

\* Abstractions - Reduce complexity and isolate impact of changes

\* Inheritance - Eliminate redundant code

\* Polymorphism - Refactor ugly switch/case statements

**Static Polymorphism**

\* occurs at compile time

\* early binding process

\* faster

\* overloaded functions and templates that happen at compile time

\* Static Polymorphism is the linking of a function with an object during compile time is called static. It is also called static binding. C# provides two techniques to implement static polymorphism i.e. Function overloading and Operator overloading.

**Dynamic Polymorphism**

\* occurs at runtime

\* late binding

\* interfaces that happen at run-time

**OBJECT ORIENTED ANALSYS and DESIGN PRINCIPLES**

\* Single Responsibility Principle -

\* Open-Close Principle -

\* Liskov Substituion Princile -

\* Interface Segregation -

\* Dependency Injection -

# System Design