

# Data Structures and Algorithms<sup>1</sup>

A Study Guide for Students of Sorsogon State  
University - Bulan Campus<sup>2</sup>

JARRIAN VINCE G. GOJAR<sup>3</sup>

September 1, 2024

<sup>1</sup>A course in the Bachelor of Science in Computer Science

<sup>2</sup>This book is a study guide for students of Sorsogon State University - Bulan Campus taking up the course Data Structures and Algorithms.

<sup>3</sup><https://github.com/godkingjay>

Sorsogon State University - Bulan Campus

# Contents

<b>Contents</b>	<b>ii</b>
<b>List of Figures</b>	<b>viii</b>
<b>List of Tables</b>	<b>ix</b>
<b>List of Codes</b>	<b>x</b>
<b>1 Introduction to Data Structures and Algorithms</b>	<b>2</b>
1.1 Introduction . . . . .	2
1.2 Setup and Installation . . . . .	2
1.2.1 C++ Compiler Installation . . . . .	2
1.2.1.1 Windows . . . . .	2
1.2.2 Visual Studio Code Installation . . . . .	3
1.2.3 Testing the Installation . . . . .	3
1.3 What are Data Structures? . . . . .	4
1.4 What are Algorithms? . . . . .	4
1.5 Why Study Data Structures and Algorithms? . . . . .	4
1.6 Basic Terminologies . . . . .	4
1.6.1 Data . . . . .	4
1.6.2 Data Object . . . . .	4
1.6.3 Data Structure . . . . .	4
1.6.4 Data Type . . . . .	4
1.6.4.1 Primitive Data Types . . . . .	4
1.6.4.2 Non-primitive Data Types . . . . .	4
1.6.5 Abstract Data Type . . . . .	4
1.6.6 Algorithm . . . . .	4
1.6.7 Complexity of an Algorithm . . . . .	4
1.6.7.1 Time Complexity . . . . .	4
1.6.7.2 Space Complexity . . . . .	4
1.7 Asymptotic Notations . . . . .	4
1.7.1 Big-O Notation . . . . .	4
1.7.2 Omega Notation . . . . .	4
1.7.3 Theta Notation . . . . .	4
1.8 Summary . . . . .	4
<b>2 Arrays and Linked Lists</b>	<b>5</b>
2.1 Introduction . . . . .	6
2.2 Arrays . . . . .	6
2.2.1 Types of Arrays . . . . .	6

2.2.1.1	One-dimensional Array	6
2.2.1.2	Multi-dimensional Array	6
2.2.2	Array Operations	6
2.2.2.1	Insertion	6
2.2.2.2	Deletion	6
2.2.2.3	Searching	6
2.2.3	Complexity Analysis of Arrays	6
2.3	Linked Lists	6
2.3.1	Types of Linked Lists	6
2.3.1.1	Singly Linked List	6
2.3.1.2	Doubly Linked List	6
2.3.1.3	Circular Linked List	6
2.3.2	Operations on Linked Lists	6
2.3.2.1	Insertion	6
2.3.2.2	Deletion	6
2.3.2.3	Searching	6
2.3.3	Complexity Analysis of Linked Lists	6
2.4	Comparison of Arrays and Linked Lists	6
2.5	Summary	6
<b>3</b>	<b>Stacks and Queues</b>	<b>7</b>
3.1	Introduction	8
3.2	Stacks	8
3.2.1	Operations on Stacks	8
3.2.1.1	Push	8
3.2.1.2	Pop	8
3.2.1.3	Peek	8
3.2.1.4	isEmpty	8
3.2.1.5	isFull	8
3.2.2	Complexity Analysis of Stacks	8
3.2.3	Implementation of Stacks Using Arrays	8
3.2.4	Implementation of Stacks Using Linked Lists	8
3.3	Queues	8
3.3.1	Types of Queues	8
3.3.1.1	Linear Queue	8
3.3.1.2	Circular Queue	8
3.3.1.3	Priority Queue	8
3.3.1.4	Double-ended Queue (Deque)	8
3.3.2	Operations on Queues	8
3.3.2.1	Enqueue	8
3.3.2.2	Dequeue	8
3.3.2.3	Front	8
3.3.2.4	Rear	8
3.3.3	Complexity Analysis of Queues	8
3.3.4	Implementation of Queues Using Arrays	8
3.3.5	Implementation of Queues Using Linked Lists	8
3.4	Comparison of Stacks and Queues	8
3.5	Summary	8
<b>4</b>	<b>Trees</b>	<b>9</b>
4.1	Introduction	10
4.2	Properties of Trees	10

4.2.1	Root Node	10
4.2.2	Parent Node	10
4.2.3	Child Node	10
4.2.4	Leaf Node	10
4.2.5	Ancestors	10
4.2.6	Siblings	10
4.2.7	Descendants	10
4.2.8	Height of a Tree	10
4.2.9	Depth of a Node	10
4.2.10	Degree of a Node	10
4.2.11	Level of a Node	10
4.2.12	Subtree	10
4.3	Types of Trees	10
4.3.1	Binary Tree	10
4.3.1.1	Types of Binary Trees	10
	Left-skewed Binary Tree	10
	Right-skewed Binary Tree	10
	Complete Binary Tree	10
4.3.2	Ternary Tree	10
4.3.3	N-ary Tree	10
4.3.4	Binary Search Tree	10
4.3.5	AVL Tree	10
4.3.6	Red-Black Tree	10
4.4	Basic Operations on Trees	10
4.4.1	Creation of a Tree	10
4.4.2	Insertion	10
4.4.3	Deletion	10
4.4.4	Searching	10
4.4.5	Traversal	10
4.4.5.1	Preorder Traversal	10
4.4.5.2	Inorder Traversal	10
4.4.5.3	Postorder Traversal	10
4.4.5.4	Level-order Traversal	10
4.5	Complexity Analysis of Trees	10
4.6	Summary	10
<b>5</b>	<b>Graphs</b>	<b>11</b>
5.1	Introduction	12
5.2	Properties of Graphs	12
5.2.1	Vertex	12
5.2.2	Edge	12
5.2.3	Degree of a Vertex	12
5.2.4	Path	12
5.3	Types of Graphs	12
5.3.1	Finite Graph	12
5.3.2	Infinite Graph	12
5.3.3	Trivial Graph	12
5.3.4	Simple Graph	12
5.3.5	Multi Graph	12
5.3.6	Null Graph	12
5.3.7	Complete Graph	12

5.3.8	Pseudo Graph	12
5.3.9	Regular Graph	12
5.3.10	Bipartite Graph	12
5.3.11	Labelled Graph	12
5.3.12	Weighted Graph	12
5.3.13	Directed Graph	12
5.3.14	Undirected Graph	12
5.3.15	Connected Graph	12
5.3.16	Disconnected Graph	12
5.3.17	Cyclic Graph	12
5.3.18	Acyclic Graph	12
5.3.19	Directed Acyclic Graph (DAG)	12
5.3.20	Digraph	12
5.3.21	Subgraph	12
5.4	Operations on Graphs	12
5.4.1	Creation of a Graph	12
5.4.2	Insertion	12
5.4.2.1	Insertion of a Vertex	12
5.4.2.2	Insertion of an Edge	12
5.4.3	Deletion	12
5.4.3.1	Deletion of a Vertex	12
5.4.3.2	Deletion of an Edge	12
5.4.4	Traversal	12
5.4.4.1	Depth First Search (DFS)	12
5.4.4.2	Breadth First Search (BFS)	12
5.4.5	Shortest Path	12
5.4.6	Minimum Spanning Tree	12
5.5	Complexity Analysis of Graphs	12
5.6	Summary	12
<b>6</b>	<b>Sorting and Searching</b>	<b>13</b>
6.1	Introduction	14
6.2	Sorting	14
6.2.1	Types of Sorting Algorithms	14
6.2.1.1	Bubble Sort	14
6.2.1.2	Selection Sort	14
6.2.1.3	Insertion Sort	14
6.2.1.4	Merge Sort	14
6.2.1.5	Quick Sort	14
6.2.1.6	Heap Sort	14
6.2.1.7	Radix Sort	14
6.2.1.8	Counting Sort	14
6.2.1.9	Bucket Sort	14
6.2.2	Comparison of Sorting Algorithms	14
6.3	Searching	14
6.3.1	Types of Searching Algorithms	14
6.3.1.1	Linear Search	14
6.3.1.2	Binary Search	14
6.3.1.3	Jump Search	14
6.3.1.4	Interpolation Search	14
6.3.1.5	Exponential Search	14

6.3.1.6	Fibonacci Search . . . . .	14
6.3.1.7	Ternary Search . . . . .	14
6.3.2	Comparison of Searching Algorithms . . . . .	14
6.4	Summary . . . . .	14
<b>7</b>	<b>Hashing</b>	<b>15</b>
7.1	Introduction . . . . .	15
7.2	Hash Table . . . . .	15
7.3	Hash Function . . . . .	15
7.4	Collision Resolution Techniques . . . . .	15
7.4.1	Separate Chaining . . . . .	15
7.4.2	Open Addressing . . . . .	15
7.4.2.1	Linear Probing . . . . .	15
7.4.2.2	Quadratic Probing . . . . .	15
7.4.2.3	Double Hashing . . . . .	15
7.5	Complexity Analysis of Hashing . . . . .	15
7.6	Summary . . . . .	15
<b>8</b>	<b>Advanced Data Structures and Algorithms</b>	<b>16</b>
8.1	Introduction . . . . .	17
8.2	Advanced Data Structures . . . . .	17
8.2.1	Segment Tree . . . . .	17
8.2.2	Fenwick Tree . . . . .	17
8.2.3	Suffix Tree . . . . .	17
8.2.4	Suffix Array . . . . .	17
8.2.5	Trie . . . . .	17
8.2.6	Heap . . . . .	17
8.2.7	Disjoint Set . . . . .	17
8.2.8	Skip List . . . . .	17
8.2.9	Splay Tree . . . . .	17
8.2.10	Bloom Filter . . . . .	17
8.2.11	KD Tree . . . . .	17
8.2.12	Quad Tree . . . . .	17
8.2.13	Octree . . . . .	17
8.2.14	B-Tree . . . . .	17
8.2.15	B+ Tree . . . . .	17
8.2.16	R-Tree . . . . .	17
8.2.17	X-Tree . . . . .	17
8.2.18	Y-Tree . . . . .	17
8.2.19	Z-Tree . . . . .	17
8.3	Advanced Algorithms . . . . .	17
8.3.1	Dynamic Programming . . . . .	17
8.3.2	Greedy Algorithms . . . . .	17
8.3.3	Backtracking . . . . .	17
8.3.4	Divide and Conquer . . . . .	17
8.3.5	Branch and Bound . . . . .	17
8.3.6	Randomized Algorithms . . . . .	17
8.3.7	Approximation Algorithms . . . . .	17
8.3.8	String Matching Algorithms . . . . .	17
8.3.9	Pattern Searching Algorithms . . . . .	17
8.3.10	Cryptography Algorithms . . . . .	17
8.3.11	Geometric Algorithms . . . . .	17

8.3.12	Graph Algorithms	17
8.3.13	Network Flow Algorithms	17
8.3.14	Game Theory Algorithms	17
8.3.15	Quantum Algorithms	17
8.4	Summary	17
<b>9</b>	<b>Applications of Data Structures and Algorithms</b>	<b>18</b>
9.1	Applications in Computer Science	19
9.1.1	Operating Systems	19
9.1.2	Database Management Systems	19
9.1.3	Compiler Design	19
9.1.4	Networking	19
9.1.5	Artificial Intelligence	19
9.1.6	Machine Learning	19
9.1.7	Computer Graphics	19
9.1.8	Computer Vision	19
9.1.9	Robotics	19
9.1.10	Web Development	19
9.1.11	Mobile Development	19
9.1.12	Game Development	19
9.1.13	Cybersecurity	19
9.1.14	Quantum Computing	19
9.2	Applications in Real Life	19
9.2.1	Social Media	19
9.2.2	E-commerce	19
9.2.3	Healthcare	19
9.2.4	Finance	19
9.2.5	Transportation	19
9.2.6	Education	19
9.2.7	Agriculture	19
9.2.8	Manufacturing	19
9.2.9	Entertainment	19
9.2.10	Sports	19
9.2.11	Travel	19
9.2.12	Telecommunications	19
9.2.13	Energy	19
9.2.14	Environment	19
9.2.15	Politics	19
9.2.16	Military	19
9.3	Summary	19
<b>10</b>	<b>References</b>	<b>20</b>



# List of Figures

# List of Tables

# List of Codes

1.1	Hello World Program . . . . .	3
1.2	Compiling the Program . . . . .	3
1.3	Running the Program . . . . .	3

# Preface

*“Bad programmers worry about the code. Good programmers worry about data structures and their relationships.”*

– Linus Torvalds

Jarrian Vince G. Gojar

<https://github.com/godkingjay>

# 1

# Introduction to Data Structures and Algorithms

## 1.1 Introduction

Data structures and algorithms are the building blocks of computer science. They are essential for solving complex problems efficiently and effectively. Data structures are used to store and organize data in a computer so that it can be accessed and manipulated efficiently. Algorithms are step-by-step procedures or formulas for solving a problem. They are the instructions that tell a computer how to perform a task.

In this course, we will learn about the fundamental data structures and algorithms that are used in computer science. We will study how to design, implement, and analyze data structures and algorithms to solve real-world problems. By the end of this course, you will have a solid foundation in data structures and algorithms that will help you become a better programmer and problem solver.

## 1.2 Setup and Installation

In this course, we will be using the C++ programming language to implement data structures and algorithms. C++ is a powerful and versatile programming language that is widely used in the field of computer science. To get started, you will need to install a C++ compiler and an integrated development environment (IDE) on your computer.

### 1.2.1 C++ Compiler Installation

The first step is to install a C++ compiler on your computer. A compiler is a program that translates source code written in a programming language into machine code that can be executed by a computer. There are several C++ compilers available, but we recommend using the GNU Compiler Collection (GCC) which is a free and open-source compiler that supports multiple programming languages including C++.

#### 1.2.1.1 Windows

To install GCC on Windows, you can use the MinGW (Minimalist GNU for Windows) project which provides a port of GCC to Windows. You can download the MinGW installer from the MinGW website and follow the installation instructions. You can install MinGW

by following the instructions here: [https://code.visualstudio.com/docs/languages/cpp#\\_example-install-mingwx64-on-windows](https://code.visualstudio.com/docs/languages/cpp#_example-install-mingwx64-on-windows)

### 1.2.2 Visual Studio Code Installation

The next step is to install an integrated development environment (IDE) on your computer. An IDE is a software application that provides comprehensive facilities to computer programmers for software development. We recommend using Visual Studio Code which is a free and open-source IDE developed by Microsoft. You can download Visual Studio Code from the official website and follow the installation instructions: <https://code.visualstudio.com/Download>

Other than Visual Studio Code, you also need to install the C/C++ extension for Visual Studio Code. You can install the C/C++ extension by following the instructions here: <https://code.visualstudio.com/docs/languages/cpp>

### 1.2.3 Testing the Installation

To test if the installation was successful, you can create a simple C++ program and compile it using the C++ compiler. Open Visual Studio Code and create a new file with the following C++ code:

```
1 #include <iostream>
2 namespace std;
3
4 int main() {
5     cout << "Hello, World!" << endl;
6     return 0;
7 }
```

Code 1.1: Hello World Program

Save the file with a .cpp extension (e.g., hello.cpp) and open a terminal window in Visual Studio Code. Compile the program using the following command:

```
1 g++ hello.cpp -o hello
```

Code 1.2: Compiling the Program

If there are no errors, you can run the program by executing the following command:

```
1 ./hello
```

Code 1.3: Running the Program

If everything is set up correctly, you should see the output "Hello, World!" printed on the screen.

## 1.3 What are Data Structures?

## 1.4 What are Algorithms?

## 1.5 Why Study Data Structures and Algorithms?

## 1.6 Basic Terminologies

### 1.6.1 Data

### 1.6.2 Data Object

### 1.6.3 Data Structure

### 1.6.4 Data Type

#### 1.6.4.1 Primitive Data Types

#### 1.6.4.2 Non-primitive Data Types

### 1.6.5 Abstract Data Type

### 1.6.6 Algorithm

### 1.6.7 Complexity of an Algorithm

#### 1.6.7.1 Time Complexity

#### 1.6.7.2 Space Complexity

## 1.7 Asymptotic Notations

### 1.7.1 Big-O Notation

### 1.7.2 Omega Notation

### 1.7.3 Theta Notation

## 1.8 Summary





## 2

# Arrays and Linked Lists

## 2.1 Introduction

## 2.2 Arrays

### 2.2.1 Types of Arrays

#### 2.2.1.1 One-dimensional Array

#### 2.2.1.2 Multi-dimensional Array

### 2.2.2 Array Operations

#### 2.2.2.1 Insertion

#### 2.2.2.2 Deletion

#### 2.2.2.3 Searching

### 2.2.3 Complexity Analysis of Arrays

## 2.3 Linked Lists

### 2.3.1 Types of Linked Lists

#### 2.3.1.1 Singly Linked List

#### 2.3.1.2 Doubly Linked List

#### 2.3.1.3 Circular Linked List

### 2.3.2 Operations on Linked Lists

#### 2.3.2.1 Insertion

#### 2.3.2.2 Deletion

#### 2.3.2.3 Searching

### 2.3.3 Complexity Analysis of Linked Lists

## 2.4 Comparison of Arrays and Linked Lists

## 2.5 Summary



# 3

## Stacks and Queues

### 3.1 Introduction

### 3.2 Stacks

#### 3.2.1 Operations on Stacks

##### 3.2.1.1 Push

##### 3.2.1.2 Pop

##### 3.2.1.3 Peek

##### 3.2.1.4 isEmpty

##### 3.2.1.5 isFull

#### 3.2.2 Complexity Analysis of Stacks

#### 3.2.3 Implementation of Stacks Using Arrays

#### 3.2.4 Implementation of Stacks Using Linked Lists

### 3.3 Queues

#### 3.3.1 Types of Queues

##### 3.3.1.1 Linear Queue

##### 3.3.1.2 Circular Queue

##### 3.3.1.3 Priority Queue

##### 3.3.1.4 Double-ended Queue (Deque)

#### 3.3.2 Operations on Queues

##### 3.3.2.1 Enqueue

##### 3.3.2.2 Dequeue

##### 3.3.2.3 Front

##### 3.3.2.4 Rear

#### 3.3.3 Complexity Analysis of Queues

#### 3.3.4 Implementation of Queues Using Arrays

#### 3.3.5 Implementation of Queues Using Linked Lists

### 3.4 Comparison of Stacks and Queues



# 4

## Trees

### 4.1 Introduction

### 4.2 Properties of Trees

#### 4.2.1 Root Node

#### 4.2.2 Parent Node

#### 4.2.3 Child Node

#### 4.2.4 Leaf Node

#### 4.2.5 Ancestors

#### 4.2.6 Siblings

#### 4.2.7 Descendants

#### 4.2.8 Height of a Tree

#### 4.2.9 Depth of a Node

#### 4.2.10 Degree of a Node

#### 4.2.11 Level of a Node

#### 4.2.12 Subtree

### 4.3 Types of Trees

#### 4.3.1 Binary Tree

##### 4.3.1.1 Types of Binary Trees

Left-skewed Binary Tree

Right-skewed Binary Tree

Complete Binary Tree

#### 4.3.2 Ternary Tree

#### 4.3.3 N-ary Tree

#### 4.3.4 Binary Search Tree

#### 4.3.5 AVL Tree

#### 4.3.6 Red-Black Tree

#### 4.4 Basic Operations on Trees



# 5

## Graphs

### 5.1 Introduction

### 5.2 Properties of Graphs

#### 5.2.1 Vertex

#### 5.2.2 Edge

#### 5.2.3 Degree of a Vertex

#### 5.2.4 Path

### 5.3 Types of Graphs

#### 5.3.1 Finite Graph

#### 5.3.2 Infinite Graph

#### 5.3.3 Trivial Graph

#### 5.3.4 Simple Graph

#### 5.3.5 Multi Graph

#### 5.3.6 Null Graph

#### 5.3.7 Complete Graph

#### 5.3.8 Pseudo Graph

#### 5.3.9 Regular Graph

#### 5.3.10 Bipartite Graph

#### 5.3.11 Labelled Graph

#### 5.3.12 Weighted Graph

#### 5.3.13 Directed Graph

#### 5.3.14 Undirected Graph

#### 5.3.15 Connected Graph

#### 5.3.16 Disconnected Graph

#### 5.3.17 Cyclic Graph

#### 5.3.18 Acyclic Graph

#### 5.3.19 Directed Acyclic Graph (DAG)





## 6

# Sorting and Searching

## 6.1 Introduction

## 6.2 Sorting

### 6.2.1 Types of Sorting Algorithms

#### 6.2.1.1 Bubble Sort

#### 6.2.1.2 Selection Sort

#### 6.2.1.3 Insertion Sort

#### 6.2.1.4 Merge Sort

#### 6.2.1.5 Quick Sort

#### 6.2.1.6 Heap Sort

#### 6.2.1.7 Radix Sort

#### 6.2.1.8 Counting Sort

#### 6.2.1.9 Bucket Sort

### 6.2.2 Comparison of Sorting Algorithms

## 6.3 Searching

### 6.3.1 Types of Searching Algorithms

#### 6.3.1.1 Linear Search

#### 6.3.1.2 Binary Search

#### 6.3.1.3 Jump Search

#### 6.3.1.4 Interpolation Search

#### 6.3.1.5 Exponential Search

#### 6.3.1.6 Fibonacci Search

#### 6.3.1.7 Ternary Search

### 6.3.2 Comparison of Searching Algorithms

## 6.4 Summary

# 7

## Hashing

### 7.1 Introduction

### 7.2 Hash Table

### 7.3 Hash Function

### 7.4 Collision Resolution Techniques

#### 7.4.1 Separate Chaining

#### 7.4.2 Open Addressing

##### 7.4.2.1 Linear Probing

##### 7.4.2.2 Quadratic Probing

##### 7.4.2.3 Double Hashing

### 7.5 Complexity Analysis of Hashing

### 7.6 Summary



## 8

# Advanced Data Structures and Algorithms

## 8.1 Introduction

## 8.2 Advanced Data Structures

### 8.2.1 Segment Tree

### 8.2.2 Fenwick Tree

### 8.2.3 Suffix Tree

### 8.2.4 Suffix Array

### 8.2.5 Trie

### 8.2.6 Heap

### 8.2.7 Disjoint Set

### 8.2.8 Skip List

### 8.2.9 Splay Tree

### 8.2.10 Bloom Filter

### 8.2.11 KD Tree

### 8.2.12 Quad Tree

### 8.2.13 Octree

### 8.2.14 B-Tree

### 8.2.15 B+ Tree

### 8.2.16 R-Tree

### 8.2.17 X-Tree

### 8.2.18 Y-Tree

### 8.2.19 Z-Tree

## 8.3 Advanced Algorithms

### 8.3.1 Dynamic Programming

### 8.3.2 Greedy Algorithms



## 9

# Applications of Data Structures and Algorithms

## 9.1 Applications in Computer Science

9.1.1 Operating Systems

9.1.2 Database Management Systems

9.1.3 Compiler Design

9.1.4 Networking

9.1.5 Artificial Intelligence

9.1.6 Machine Learning

9.1.7 Computer Graphics

9.1.8 Computer Vision

9.1.9 Robotics

9.1.10 Web Development

9.1.11 Mobile Development

9.1.12 Game Development

9.1.13 Cybersecurity

9.1.14 Quantum Computing

## 9.2 Applications in Real Life

9.2.1 Social Media

9.2.2 E-commerce

9.2.3 Healthcare

9.2.4 Finance

9.2.5 Transportation

9.2.6 Education

9.2.7 Agriculture

9.2.8 Manufacturing

9.2.9 Entertainment

# References

## A. Books

- Vishwas R. (2023). Data Structure Handbook. Dr. Vishwas Raval. ISBN: 978-9359063591
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Introduction to algorithms. MIT press. ISBN: 978-0262046305
- Erickson, J. (2019). Algorithms. ISBN: 978-1792644832

## B. Other Sources

- Tutorialspoint. (n.d.). Data Structures Basics. Data Structure Basics. [https://www.tutorialspoint.com/data\\_structures\\_algorithms/data\\_structures\\_basics.htm](https://www.tutorialspoint.com/data_structures_algorithms/data_structures_basics.htm)
- Algorithm Archive · Arcane Algorithm Archive. (n.d.). <https://www.algorithm-archive.org/>