1 Data Structures and Algorithms

1.1 Array, Stack, Queue, Graph, Tree, Traversal, Linked List, Set, Hash Table

- An **array** is like a list where you store items in a specific order, like a row of boxes. - A **stack** is like a pile of plates; you add or remove from the top. - A **queue** is like a line at a store; you add at the end and remove from the front. - A **graph** is a network of points (nodes) connected by lines, like a map of cities. - A **tree** is a special graph that looks like a family tree, with a top (root) and branches. - **Traversal** means visiting each item in a structure, like checking every box in a row. - A **linked list** is a chain of items where each item points to the next, like a train. - A **set** is a collection with no duplicates, like a unique group of friends. - A **hash table** is a fast way to store and find data using keys, like a phonebook (Aho et al., Ch. 1; Kruse et al., Ch. 2).

2 Programming Concepts

2.1 Imperative or Sequential Programming, Modular Programming, Object-Oriented Programming

- **Imperative programming** is giving step-by-step instructions, like following a recipe. - **Modular programming** breaks a big program into smaller, manageable pieces, like chapters in a book. - **Object-oriented programming** treats data as ”objects” (like a car) with actions (like drive), making code reusable (Schiltt, Ch. 3; Liang, Ch. 4).

2.2 Concept, Polymorphism, Inheritance, Static and Dynamic Bindings, Interface, Abstract Class, Generic Programming, Memory Management of C++ vs Java

- **Concept** in OOP means using objects to solve problems. - **Polymorphism** lets one thing work in different ways, like a button doing different actions. - **Inheritance** allows one object to take properties from another, like a child inheriting traits. - **Static binding** decides actions at compile time, while **dynamic binding** decides at runtime. - An **interface** is a plan that objects must follow, like a contract. - An **abstract class** is a blueprint that can’t be used directly but helps create other classes. - **Generic programming** uses general types, like a box that can hold anything. - **Memory management** in **C++** requires manual control, while **Java** handles it automatically with garbage collection (Schiltt, Ch. 5; Liang, Ch. 6).