

**A INDUSTRIAL TRAINING REPORT ON**

**C++ AND DATA STRUCTURE AND ALGORITHM**

**AT CODING NINJAS AND GREAT LEARNING**

**Submitted in partial fulfillment for the Degree**

**Of B.Tech In**

**Computer Science Engineering**

**Session: 2022-23**

**Submitted by**

**Rahul Saini 19EBKCS091**

DEPARTMENT OF COMPUTER SCIENCE

**B.K. BIRLA INSTITUTE OF ENGINEERING &TECHNOLOGY**

**PILANI - 333 031 (RAJ)**

(Affiliated to Bikaner Technical University, Bikaner)

**Acknowledgments**

The experience I had in the internship has been of great value in terms of learning and industrial experience. I take this opportunity to express my deepest gratitude and special thanks to the Coding ninjas and Great Learning for the training.

I am fortunate for being part of BKBIET, the institution that stood by my side and assisted me towards successfully completing the industrial training. I also take this opportunity to thank the **Department of Computer Science and Engineering, BKBIET Pilani**, for their tremendous help and guidance.

I perceive as this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, in order to attain desired career objectives. Hope to continue cooperation with all of you in the future.

**BY**

**Rahul saini**

**19EBKCS091**

**CERTIFICATE**

****

****

****

**ABSTRACT**

Industrial training is an important phase of a student life. A well planned, properly executed, and evaluated industrial training helps a lot in developing a professional attitude. The aim and motivation of this industrial training is to receive discipline, skills, teamwork, and technical knowledge through a proper training environment, which will help a person to develop a responsiveness of the self-disciplinary nature of problems in information and communication technology.

A data structure is a specialized format for organizing, processing, retrieving and storing data. There are several basic and advanced types of data structures, all designed to arrange data to suit a specific purpose. Data structures make it easy for users to access and work with the data they need in appropriate ways. Most importantly, data structures frame the organization of information so that machines and humans can better understand it.

For instance, in an object-oriented programming language, the data structure and its associated methods are bound together as part of a class definition. In non-object-oriented languages, there may be functions defined to work with the data structure, but they are not technically part of the data structure.

Hence, this training has given us a great opportunity to learn various topics about the Data Structure and Algorithm that has been widely used in industry currently.

I have learned various topics like Array, String, linked List, Stack, Queue, Tree, and Graph.

**TABLE OF CONTENTS**

**CONTENT PAGE NO.**

1. Cover Page …………………………………………………………………………….…… 1

2. Acknowledgement………………………………………………………………………. 2

3. Certificate……………………………………………………………………………………. 3

4. Abstract………………………………………………………………………………………. 4

**5. Introduction**

5.1 Introduction…………………………………………… 7

5.2 Objective…………………………………………… 8

5.2.1 To gain skills and knowledge

5.2.2 To get field work experience

5.2.3 To enhance our communication skills

5.2.4 To link theory with practice

**6. Modules Of Training**

6.1 Program, Flowchart, Pseudocode ……………………………………………….9

6.2 Array……………………………………………10

6.3 String……………………………………………11

6.4 Recursion………………………………………………………….12

6.5 Linked list……………………………………………………………13

6.6 Stack………………………………………………..15

6.7 Queue…………………………………………………16

6.8 Hashing………………………………………………….18

6.9 Tree…………………………………………………………20

6.10 Graph………………………………………………………22

**7. Conclusion and Future Scope**

**7.1 Conclusion………………………………………………………………………………….23**

**7.2 Future Scope………………………………………………………………………………23**

**INTRODUCTION**

* 1. **Introduction**

A data structure is a way of organizing and storing data in a computer so that it can be accessed and modified efficiently. Different types of data structures are suited to different kinds of applications, and some are highly specialized to specific tasks. Some common data structures include:

* Arrays: An array is a collection of items that are stored in a contiguous block of memory. Each item in an array is accessed by its index, which is the position of the item in the array. Arrays are simple and efficient, but they are not well suited to inserting or deleting items in the middle of the collection.
* Linked lists: A linked list is a collection of items where each item points to the next one in the list. Linked lists are dynamic and can be easily modified, but they are not as efficient as arrays for accessing individual items.
* Stacks and queues: Stacks and queues are special types of data structures that are designed to operate in a last-in, first-out (LIFO) or first-in, first-out (FIFO) manner, respectively. Stacks are often used to implement function calls in programming languages, while queues are used to store data that needs to be processed in a specific order.
* Trees and graphs: Trees and graphs are data structures that are used to represent hierarchical or relationship-based data. Trees are collections of nodes that are connected by edges, with the top node (called the root) serving as the starting point. Graphs are collections of nodes that may be connected to one another in any way.
* Hash tables: A hash table is a data structure that stores data in an array-like structure, using a hash function to map the data to a specific index in the array. Hash tables are used for fast lookups and insertions, and are commonly used in database indexing and caching systems.

Data structures are an important part of computer science and are used in many different areas, including databases, operating systems, and programming languages.

* 1. **Objective**

Data structures and algorithms are important concepts of computer science and it is known as pillars of computer science. Data structures and algorithms are one of the most important skills that have been expected from top-tier companies.

Data structures and algorithms are interdependent, they need to work together to make the code efficient and optimize the process of programming. Nowadays people who enter the world of programming, coding, and developing are aware of the importance of data structures and algorithms.

**1.2.1 To gain skills and knowledge**

This training provided us essential skills and knowledge one requires in the field of coding. The crucial tools used during the tenure helped us in gaining knowledge about programming languages.

**1.2.2 To get field work experience**

By taking this training we enhanced our knowledge in Data structure & algorithm and got insight in how the codes are actually work.

**1.2.3 To enhance our communication skills**

By interacting with my classmates I got to learn a lot. It helped me to enhance my communicative skills and represent my work with confidence. It boosted my confidence to design more webpages and create some great designs just for fun.

**1.2.4 To link theory with practice**

First we learned the theory aspect and then we put that into practice. By doing the practical work, our concept got clearer and it was easy to code into HTML once we got familiar with it. By putting our theoretical knowledge into practical, coding became more fun.

**MODULES OF TRAINING**

**6.1 Program:-**

A computer program is a sequence or set of instructions in a programming language for a computer to execute. Computer programs are one component of software, which also includes documentation and other intangible components.

**Flowchart:-**

A computer program is a sequence or set of instructions in a programming language for a computer to execute. Computer programs are one component of software, which also includes documentation and other intangible components**.**

**Pseudocode:-**

Pseudo code is a term which is often used in programming and algorithm based fields. It is a methodology that allows the programmer to represent the implementation of an algorithm. Simply, we can say that it’s the cooked up representation of an algorithm. Often at times, algorithms are represented with the help of pseudo codes as they can be interpreted by programmers no matter what their programming background or knowledge is. Pseudo code, as the name suggests, is a false code or a representation of code which can be understood by even a layman with some school level programming knowledge.

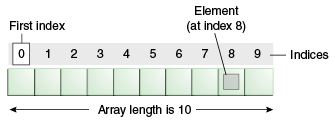
**6.2 ARRAY**

An array is a data structure that stores a collection of items of the same type. The items can be accessed through an index, which is a numerical value that corresponds to the position of the item in the array.

Arrays are useful for storing and manipulating large sets of data because they allow you to access and modify individual items quickly and easily. They are also efficient for searching and sorting data, as well as performing other operations on the data.

In most programming languages, arrays are fixed-size, meaning that once you create an array, you cannot change its size without creating a new array and copying the data over. However, some languages do have dynamic arrays, which can grow or shrink as needed.

Arrays are a fundamental data structure in computer science and are used in a wide variety of applications. They are supported by most programming languages and are an essential tool for any programmer to understand and master.



**6.3 STRING**

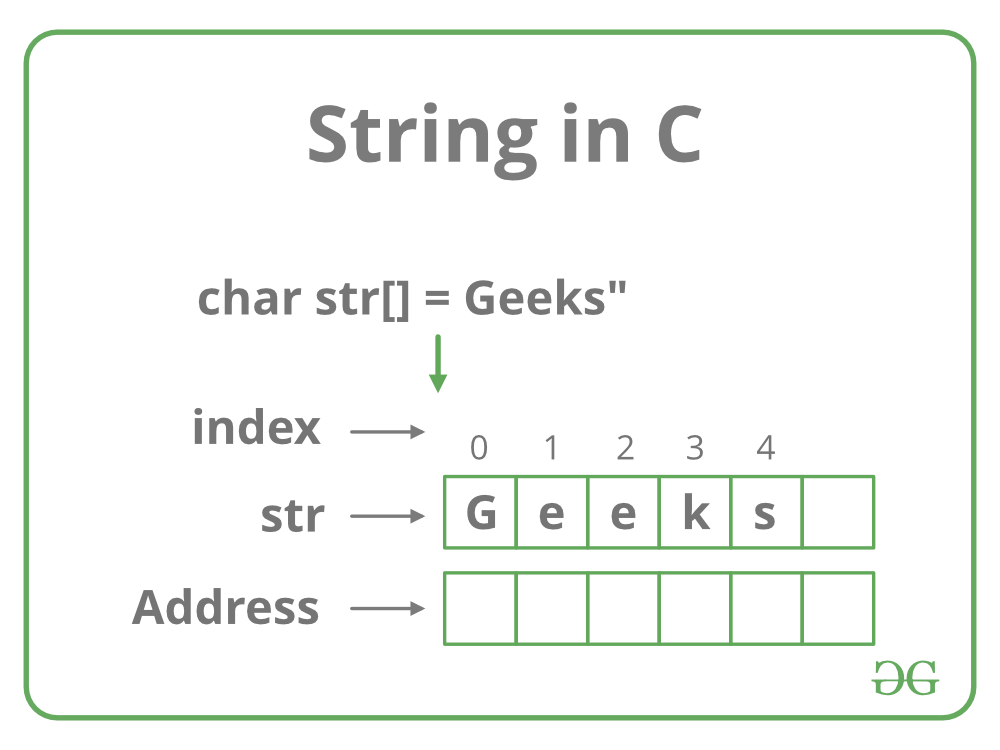
A string is a data type that represents a sequence of characters. It is often used to store and manipulate text-based data, such as words, sentences, and paragraphs.

In most programming languages, strings are implemented as an array of characters, with each character occupying a contiguous block of memory. This allows you to access and manipulate individual characters within the string using an index, similar to how you would with an array.

Strings are often used to store and display text on the screen, as well as to read and write text from files. They are also used to represent and manipulate data that is in the form of text, such as URLs, file paths, and XML documents.

In addition to the basic operations of storing, accessing, and manipulating characters within a string, most programming languages also provide a wide range of functions and methods for working with strings, such as searching for substrings, concatenating strings, and formatting strings for display.

Overall, strings are a fundamental data type that are essential for working with text-based data in almost any programming language.



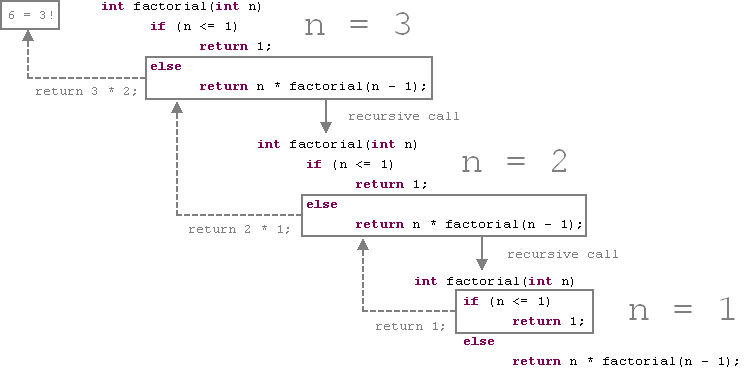
**6.4 Recursion**

Recursion is a programming technique in which a function calls itself with a modified version of its input. It is a powerful tool that allows you to solve complex problems by breaking them down into smaller, more manageable pieces.

Recursion is often used to solve problems that can be divided into smaller subproblems that are similar to the original problem. For example, the process of sorting a list of numbers can be broken down into sorting the first half of the list and the second half of the list, and then merging the two sorted halves back together.

In order to use recursion effectively, it is important to define a base case, which is a simple input for which the function can return a result directly, without calling itself. The base case serves as an "anchor" for the recursive process, allowing it to eventually terminate.

Recursion can be a challenging concept for beginners to understand, as it involves thinking about a problem in a different way than most people are used to. However, it is a powerful tool that is used extensively in computer science and is worth learning.



**6.5 Linked List**

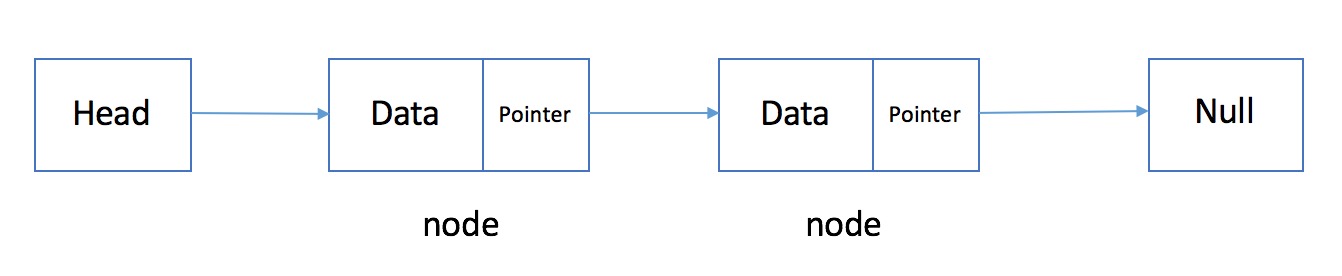
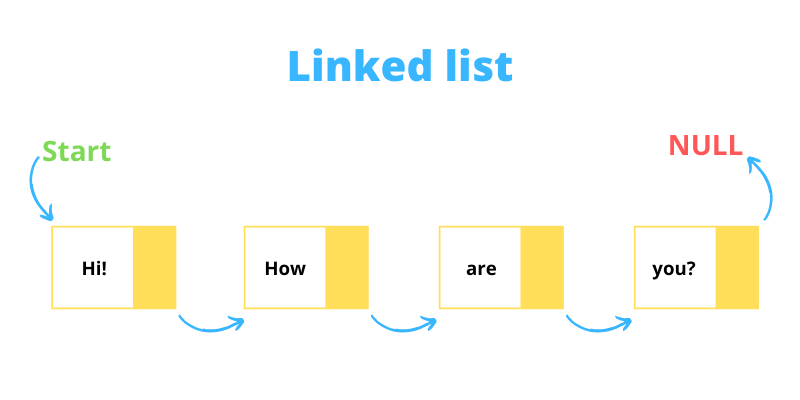
A linked list is a linear data structure that stores a sequence of elements. Each element, or node, in a linked list is composed of two parts: a value and a reference to the next element in the list.

Linked lists are often used to store and manipulate large amounts of data because they can grow or shrink dynamically as needed. They are also useful for implementing data structures that need to be able to insert and delete elements efficiently, such as stacks and queues.

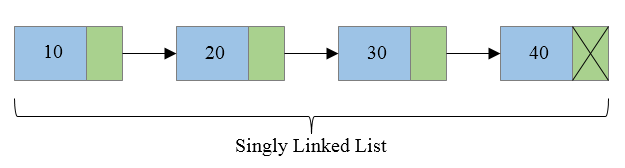
Linked lists are different from arrays in that the elements of a linked list are not stored contiguously in memory. Instead, each element is stored at a different location in memory, and the references between elements are used to link them together in a logical sequence.

There are several types of linked lists, including singly linked lists, in which each element has a reference to the next element but not the previous one, and doubly linked lists, in which each element has references to both the next and previous elements.

Overall, linked lists are a useful data structure that are widely used in computer science and are an important tool for any programmer to understand and master.



Singly linked list

****

**6.6 Stack**

A stack is a linear data structure that stores a sequence of elements in a last-in, first-out (LIFO) order. It is called a stack because it behaves like a physical stack of objects, with the most recently added element being the one that is removed first.

Stacks are often used to store and manipulate data that needs to be accessed in a specific order, such as the history of a web browser or the evaluation of arithmetic expressions. They are also used to implement algorithms that require the ability to access the most recently added element quickly, such as depth-first search and undo/redo functions.

In most programming languages, stacks are implemented using arrays or linked lists. They usually support the following basic operations:

* push: adds an element to the top of the stack
* pop: removes and returns the element at the top of the stack
* peek: returns the element at the top of the stack without removing it
* isEmpty: returns true if the stack is empty, false otherwise

Stacks are a fundamental data structure in computer science and are used in a wide variety of applications. They are supported by most programming languages and are an essential tool for any programmer to understand and master.

****

**6.7 Queue**

A queue is a linear data structure that stores a sequence of elements in a first-in, first-out (FIFO) order. It is called a queue because it behaves like a line of people waiting their turn to be served, with the person at the front of the line being the first to be served.

Queues are often used to store and manipulate data that needs to be processed in a specific order, such as tasks in a task queue or events in an event loop. They are also used to implement algorithms that require the ability to access the oldest element quickly, such as breadth-first search.

In most programming languages, queues are implemented using arrays or linked lists. They usually support the following basic operations:

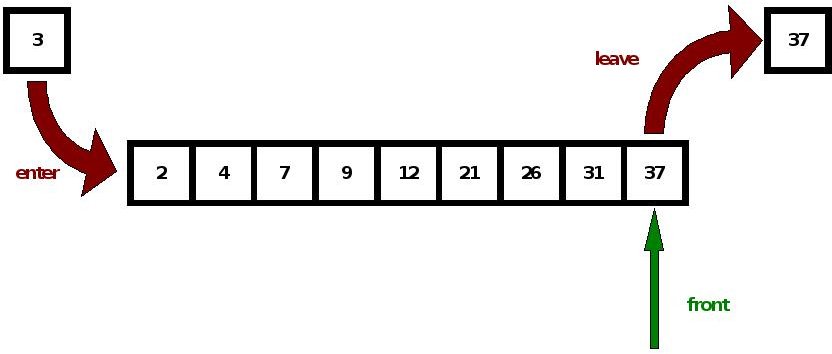
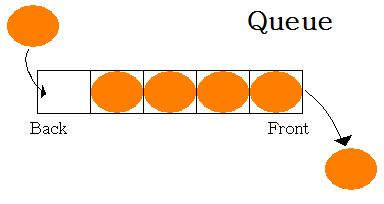
enqueue: adds an element to the back of the queue

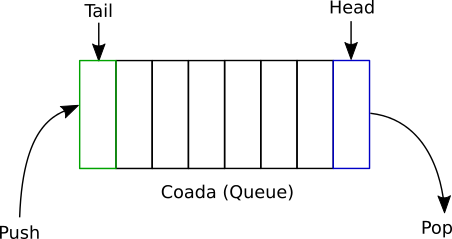
dequeue: removes and returns the element at the front of the queue

peek: returns the element at the front of the queue without removing it

isEmpty: returns true if the queue is empty, false otherwise

Queues are a fundamental data structure in computer science and are used in a wide variety of applications. They are supported by most programming languages and are an essential tool for any programmer to understand and master.



****

**6.8 Hashing**

Hashing is a technique used to map data of any size to a fixed-size value, called a hash code or hash value. Hashing is used for a variety of purposes, including data storage, data security, and data integrity.

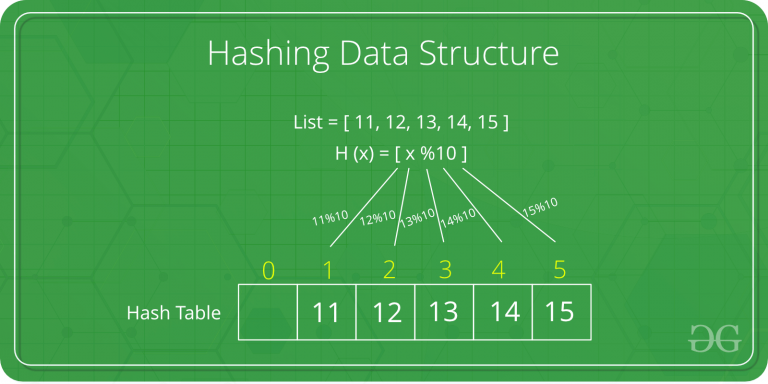
One common use of hashing is to store data in a data structure called a hash table. A hash table is a collection of key-value pairs, where each key is hashed to a unique hash code, and the corresponding value is stored in the table at the index corresponding to the hash code. This allows for fast look-up of values based on their keys, as the hash code can be used to locate the value in the table directly, without having to search through the entire table.

Hashing is also used in cryptography to create secure hashes of passwords and other sensitive data. These hashes are used to verify the integrity of the data, as any change to the original data will result in a different hash value.

Overall, hashing is a widely used technique in computer science that plays a role in many important applications. It is an essential tool for any programmer to understand and master.

Hashing is a technique or process of mapping keys, and values into the hash table by using a hash function. It is done for faster access to elements.

Let a hash function H(x) maps the value x at the index x%10 in an Array. For example if the list of values is [11,12,13,14,15] it will be stored at positions {1,2,3,4,5} in the array or Hash table respectively.

****

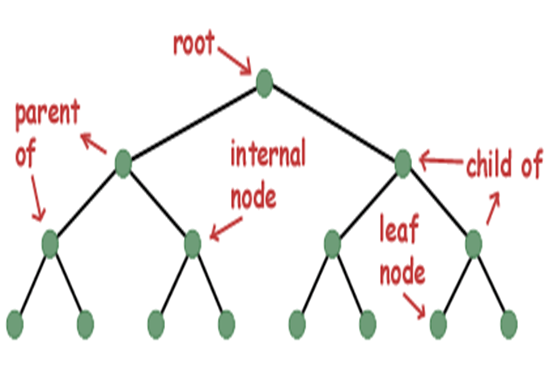
**6.9 Tree**

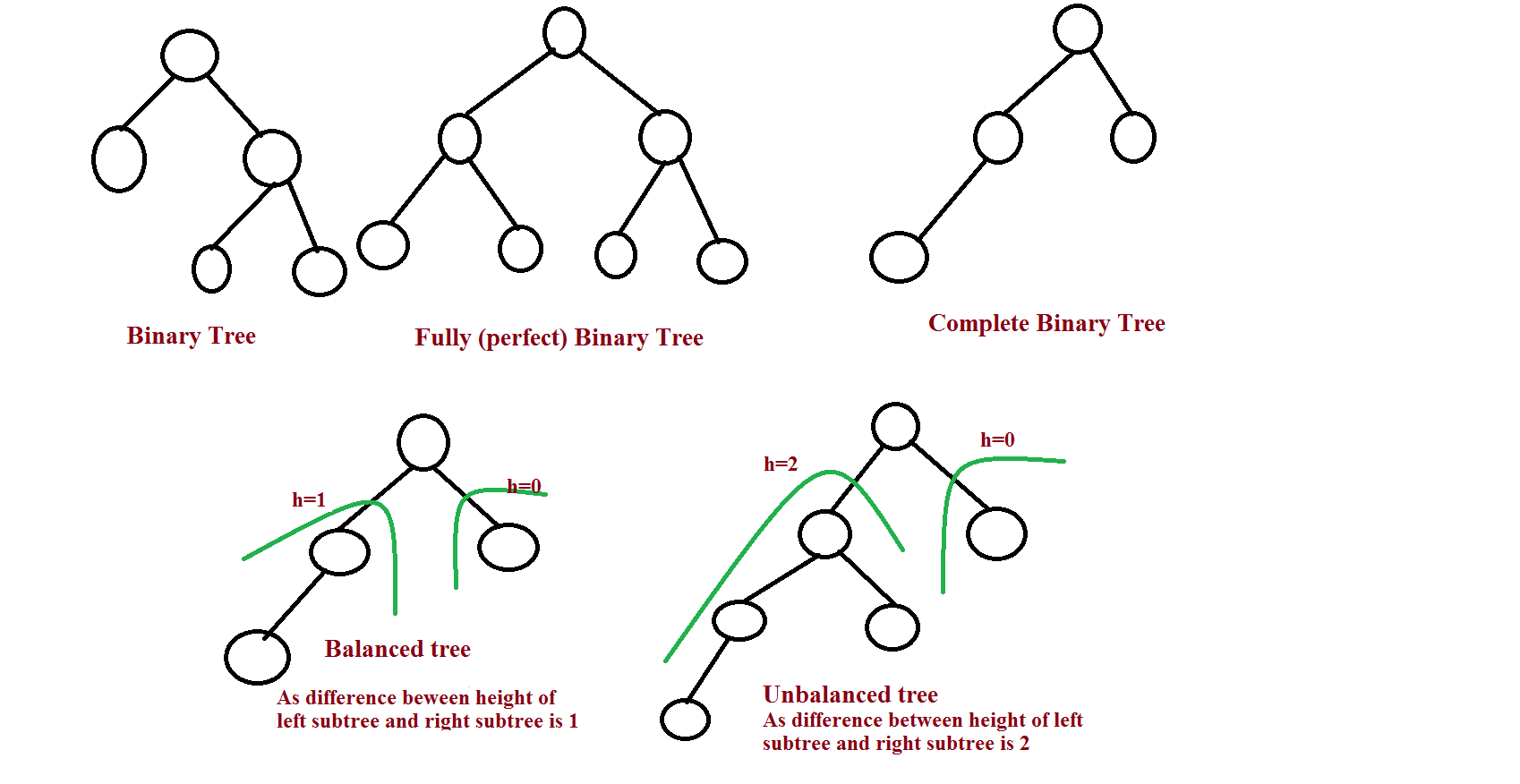
A tree is a data structure that represents a hierarchical structure. It consists of nodes, which store data, and edges, which connect the nodes in a parent-child relationship. The node at the top of the tree is called the root, and the nodes below it are called children. The children of a node are called its sub-tree.

Trees are often used to store and manipulate hierarchical data, such as the structure of a file system or the relationships between objects in an object-oriented program. They are also used to implement algorithms that require the ability to access data in a specific order, such as searching and sorting.

There are several types of trees, including binary trees, which have at most two children per node, and n-ary trees, which have any number of children per node. Trees can also be unbalanced, meaning that the nodes are not evenly distributed, or balanced, meaning that the nodes are evenly distributed.

Overall, trees are a fundamental data structure in computer science and are used in a wide variety of applications. They are supported by most programming languages and are an essential tool for any programmer to understand and master.



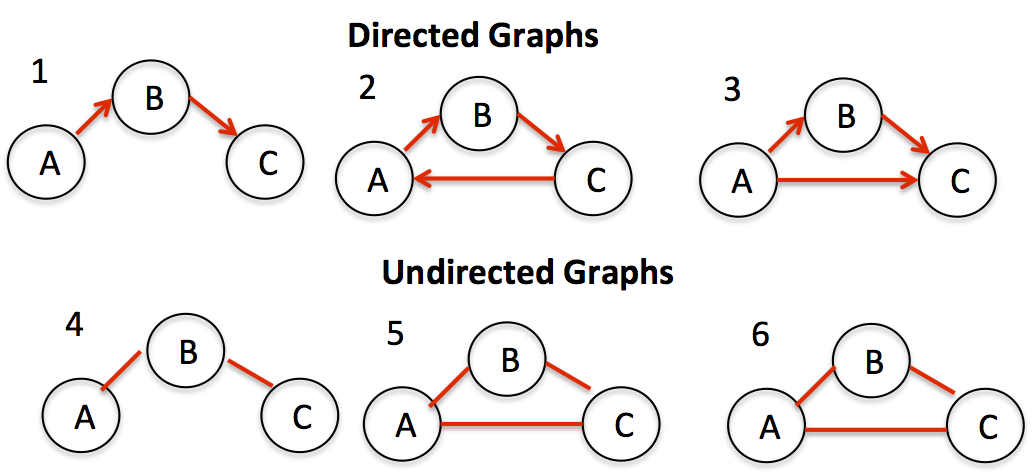
****

**6.10 Graph**

A graph is a data structure that consists of a finite set of vertices (also called nodes) and a set of edges connecting these vertices. The vertices represent entities or objects, and the edges represent relationships between these vertices.

There are two main types of graphs: directed and undirected. In a directed graph, the edges have a direction and are called arcs. In an undirected graph, the edges do not have a direction and are called edges.

Graphs can be used to represent a wide variety of real-world situations, such as social networks, biological networks, and transportation networks. They are commonly used in computer science and mathematics to represent relationships between data, and they can be analyzed using various algorithms and techniques.



**7 Conclusion and Future Scope**

**7.1 Conclusion:-**

In conclusion, I can say that this training was a great experience. Thanks to this project, I acquired deeper knowledge concerning my technical skills, but I also personally benefited.

In conclusion, DSA is an important field that plays a vital role in the development of software and the efficient manipulation of data. It involves the study of data structures, algorithms, and their performance, and it is a key subject for any computer scientist or software engineer.

**7.2 Future Scope:-**

Data structures and algorithms are fundamental concepts in computer science that are applicable in a wide range of fields. As such, there will likely always be a strong demand for professionals who are knowledgeable in these areas.

Some possible areas where data structures and algorithms may have a significant impact in the future include:

**Big data and data analytics:** As the amount of data generated continues to increase, there will be a need for more efficient data structures and algorithms to store and process this data.

**Artificial intelligence and machine learning:** Data structures and algorithms are critical for the development of AI and machine learning systems, which rely on large amounts of data and complex algorithms to perform tasks such as image and speech recognition.

**Cybersecurity:** Data structures and algorithms are also important for the development of secure systems, as they can be used to design and implement efficient encryption algorithms and data security protocols.

**Internet of Things (IoT):** The proliferation of connected devices and sensors in the IoT will require efficient data structures and algorithms to handle the large amounts of data generated and to enable real-time decision making.

Overall, the future scope of data structures and algorithms is likely to be broad and varied, as they are fundamental building blocks that are applicable in a wide range of fields and technologies.