EIGHT WEEKS SUMMER TRAINING REPORT

on

DATA STRUCTURE AND ALGORITHM (SELF PACED)

Submitted by

ABHISHEK SHARMA

Registration No : 12016768

Programme Name : B.tech (CSE – Data Science (ML and AI))

Under the Guidance of

Mr. Sandeep Jain

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

Lovely Professional University, Phagwara

(May-July,2022)

DECLARATION

I hereby declare that I have completed my Eight weeks summer training at Geeks for Geeks platform from May 17 ,2022 to July 5,2022 under the guidance of MR. Sandeep Jain. I have declare that I have worked full dedication during there 8 weeks of training and my learning outcomes fulfill the requirements of training for the award of degree of B.tech.(CSE), Lovely Professional University, Phagwara.

Date – 08 JULY 2022,

Name of Student – Abhishek Sharma

Registration no : 12016768

ACKNOWLEDGEMENT

I would like to express my gratitude towards my University as well as Geeks for Geeks for providing me the golden opportunity to do this wonderful summer training regarding DSA, which also helped me in doing a lot of homework and learning. As a result, I came to know about so many new things. So, I am really thank full to them.

Moreover I would like to thank my friends who helped me a lot whenever I got stuck in some problem related to my course. I am really thankfull to have such a good support of them as they always have my back whenever I need.

Also,I would like to mention the support system and consideration of my parents who have always been there in my life to make me choose right thing and oppose the wrong. Without them I could never had learned and became a person who I am now.

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

Summer Training Certificate By Geeks for Geeks

INTRODUCTION

DSA self paced course is a complete package that helped me to learn Data Structures and Algorithms from Basic to an Advance level. The course curriculm has been divided into 8 weeks, where I practiced questions and I have attempted the assesment tests accordingly. The course offers a wealth of programming challenges that helped me to learn all about DSA and making of an algorithm and how to solve problems and the logic behind the Algorithm.

The course was Self placed means I could join the course anytime and all the content will be avilable to me once I get enrolled. There was video lectures to learn form and multiple choice questions to practice.

I learned Algorithmic techniques for solving various problems with full flexibility of time as I was not time bounded.

This course does not require any prior knowledge of Data Structure and Algorithms, but a basic knowledge of any programming language (C++ / Java) will be helpful.

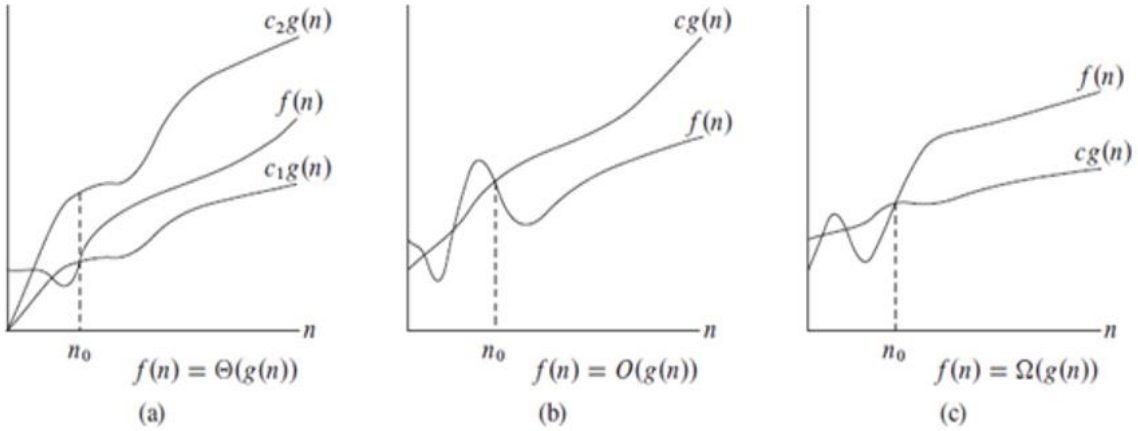
And as we all know Data Structure and Algorithm is a must skill in terms of Placement in any company because it helps us to increase our problem solving skill.

TECHNOLOGY LEARNT

It had 24 units which was further divided into chapters and then topics so during my whole 8 week course I learned the following :

INTRODUCTION TO DSA

- **Analysis of Algorithm**
    - In this I learned about background analysis through a Program and its functions.
- **Order of Growth**
    - A mathematical explanation of the growth analysis through limits and functions.
    - A direct way of calculating the order of growth
- **Asymptotic Notations**
    - Best, Average and Worst case explanation through a program.
- **Big O Notation**
    - Graphical and mathematical explanation.
    - Calculation
    - Applications at Linear Search
- **Omega Notation**
    - Graphical and mathematical explanation.
    - Calculation.
- **Theta Notation**
    - Graphical and mathematical explanation.
    - Calculation.



- **Analysis of common loops**
    - Single, multiple and nested loops
- **Analysis of Recursion**
    - Various calculations through Recursion Tree method
- **Space Complexity**
    - Basic Programs
    - Auxiliary Space
    - Space Analysis of Recursion
    - Space Analysis of Fibonacci number

MATHEMATICS

- **Finding the number of digits in a number.**
- **Arithmetic and Geometric Progressions.**
- **Quadratic Equations.**
- **Mean and Median.**
- **Prime Numbers.**
- **LCM and HCF**

- **Factorials**
- **Permutations and Combinations**
- **Modular Arithmetic**

BITMAGIC

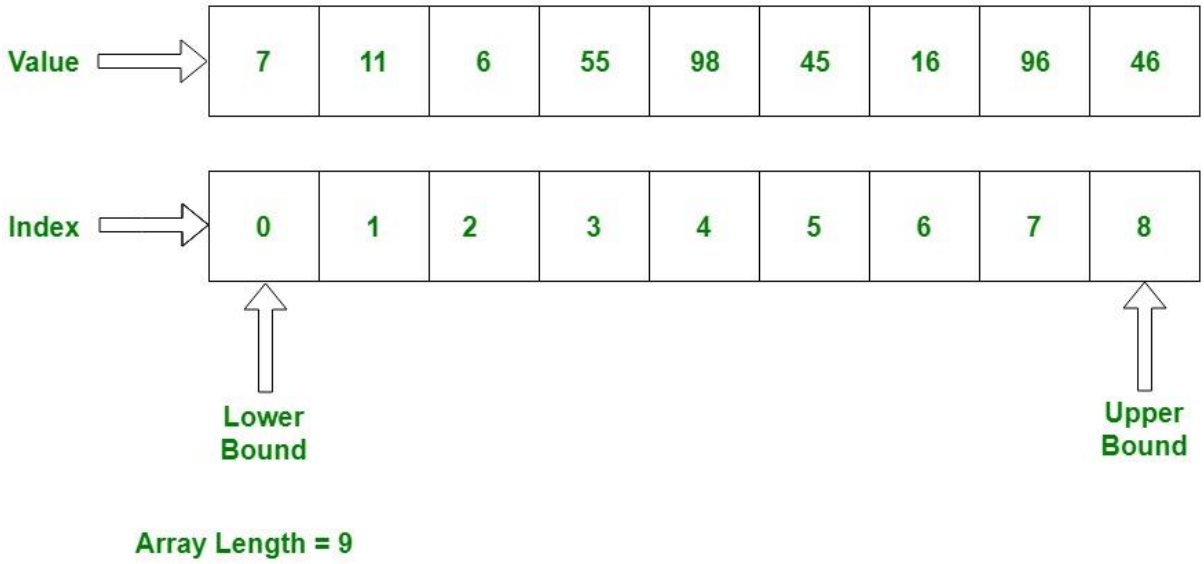
- **Bitwise Operators in C++**
    - Operation of AND, OR, XOR operators
    - Operation of Left Shift, Right Shift and Bitwise Not
- **Bitwise Operators in Java**
    - Operation of AND, OR
    - Operation of Bitwise Not, Left Shift
    - Operation of Right Shift and unsigned Right Shift
- **Problem(With Video Solutions): Check Kth bit is set or not**
    - Method 1: Using the left Shift.
    - Method 2: Using the right shift

RECURSION

- **Introduction to Recursion**
- **Applications of Recursion**
- **Writing base cases in Recursion**

- Factorial
- N-th Fibonacci number

ARRAYS



- **Introduction and Advantages**
- **Types of Arrays**
  - Fixed-sized array
  - Dynamic-sized array
- **Operations on Arrays**
  - Searching
  - Insertions
  - Deletion
  - Arrays vs other DS
  - Reversing - Explanation with complexity

SEARCHING

- **Binary Search Iterative and Recursive**
- **Binary Search and various associated problems**
- **Two Pointer Approach Problems**

SORTING

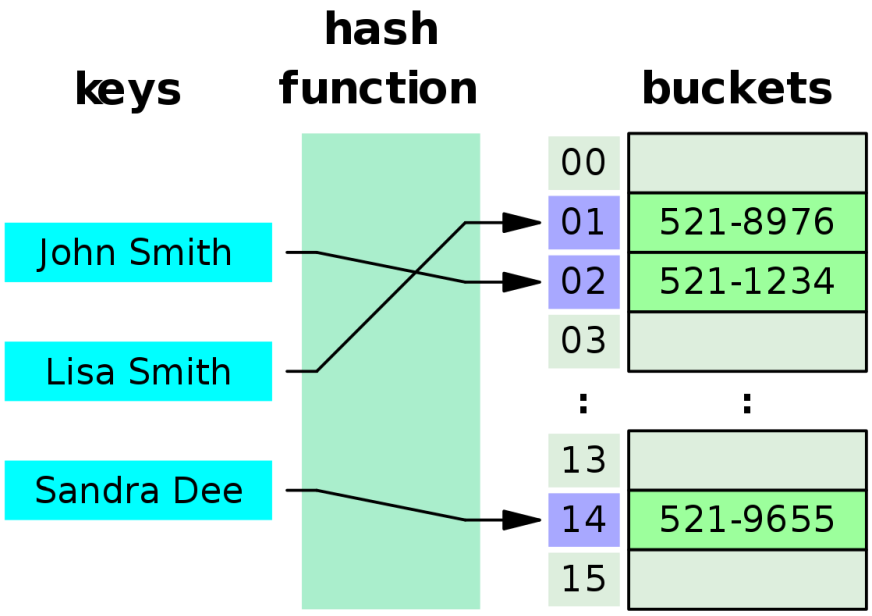
- **Implementation of C++ STL sort() function in Arrays and Vectors**
    - Time Complexities
- **Sorting in Java**
- **Arrays.sort() in Java**
- **Collection.sort() in Java**
- **Stability in Sorting Algorithms**
    - Examples of Stable and Unstable Algos
- **Insertion Sort**
- **Merge Sort**
- **Quick Sort**
    - Using Lomuto and Hoare
    - Time and Space analysis
    - Choice of Pivot and Worst case
- **Overview of Sorting Algorithms**

MATRIX

	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Row 2	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Row 3	x[2][0]	x[2][1]	x[2][2]	x[2][3]

- **Introduction to Matrix in C++ and Java**
- **Multidimensional Matrix**
- **Pass Matrix as Argument**
- **Printing matrix in a snake pattern**
- **Transposing a matrix**
- **Rotating a Matrix**
- **Check if the element is present in a row and column-wise sorted matrix.**
- **Boundary Traversal**
- **Spiral Traversal**
- **Matrix Multiplication**
- **Search in row-wise and column-wise Sorted Matrix**

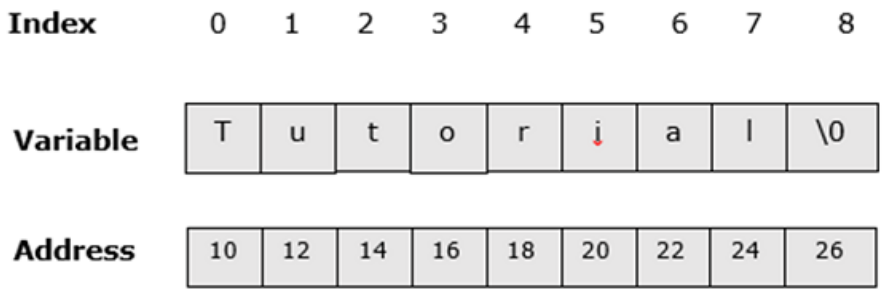
HASHING



- **Introduction and Time complexity analysis**
- **Application of Hashing**
- **Discussion on Direct Address Table**
- **Working and examples on various Hash Functions**
- **Introduction and Various techniques on Collision Handling**
- **Chaining and its implementation**
- **Open Addressing and its Implementation**
- **Chaining V/S Open Addressing**
- **Double Hashing**
- **C++**
  - Unordered Set
  - Unordered Map
- **Java**
  - HashSet

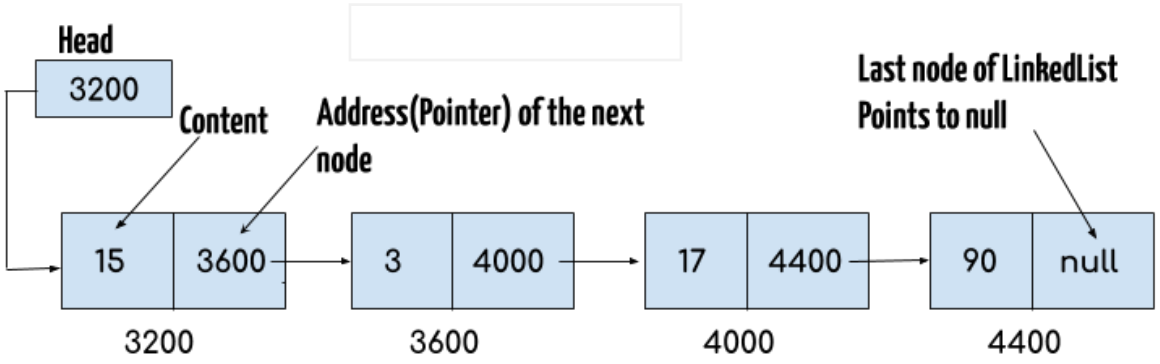
- HashMap

STRINGS



- **Discussion of String DS**
- **Strings in CPP**
- **Strings in Java**
- **Rabin Karp Algorithm**
- **KMP Algorithm**

LINKED LIST

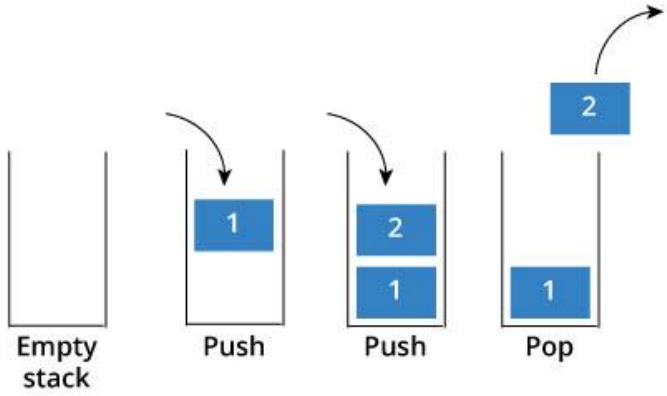


- **Introduction**
  - Implementation in CPP
  - Implementation in Java

- Comparison with Array DS

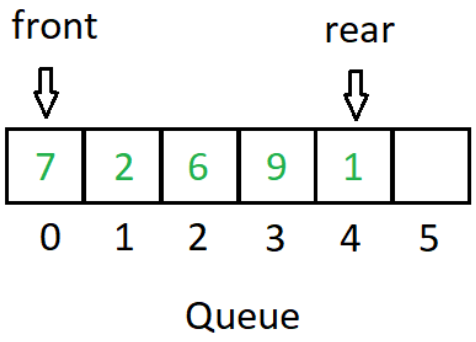
- **Doubly Linked List**
- **Circular Linked List**
- **Loop Problems**
  - Detecting Loops
  - Detecting loops using Floyd cycle detection
  - Detecting and Removing Loops in Linked List

STACK



- **Understanding the Stack data structure**
- **Applications of Stack**
- **Implementation of Stack in Array and Linked List**
  - In C++
  - In Java

QUEUE

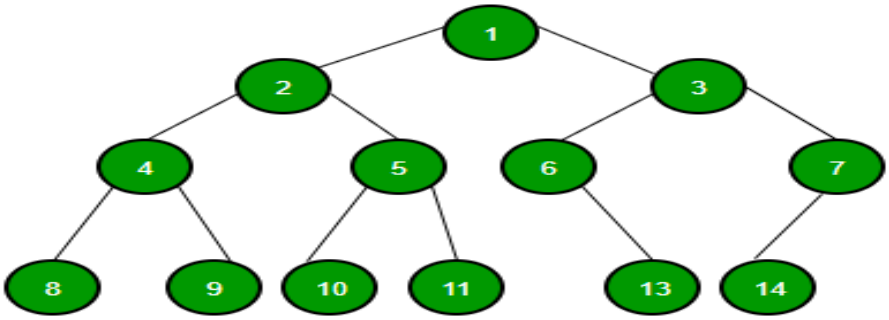


- **Introduction and Application**
- **Implementation of the queue using array and LinkedList**
  - In C++ STL
  - In Java
  - Stack using queue

DEQUE

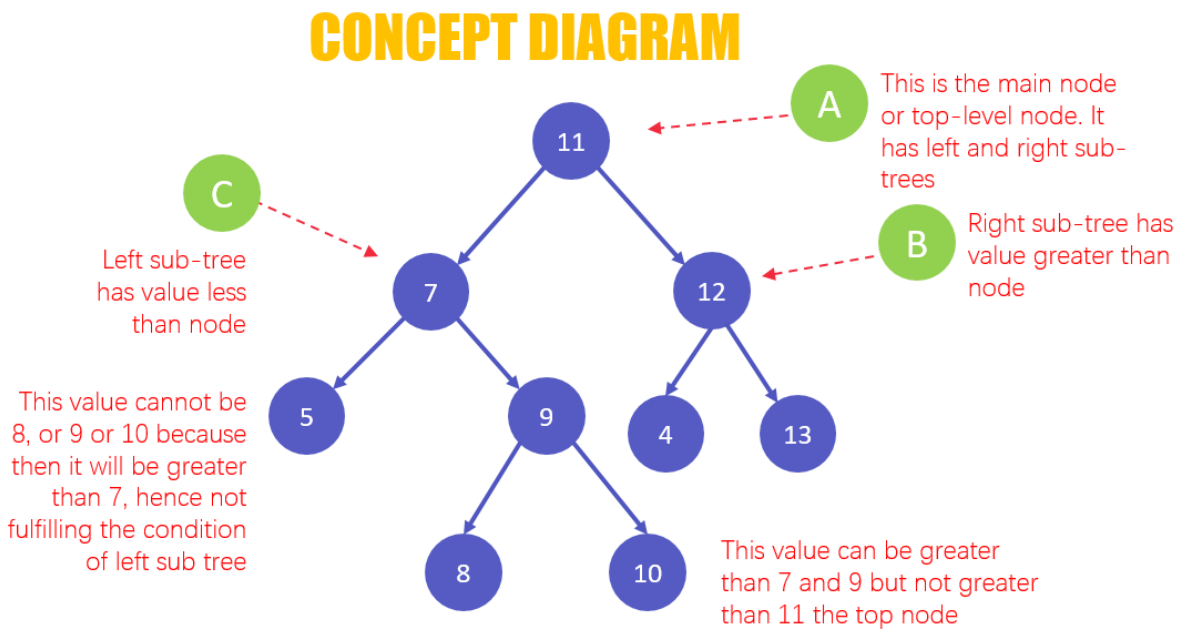
- **Introduction and Application**
- **Implementation**
  - In C++ STL
  - In Java
- **Problems(With Video Solutions)**
  - Maximums of all subarrays of size k
  - ArrayDeque in Java
  - Design a DS with min max operations

TREE



- **Introduction**
  - Tree
  - Application
  - Binary Tree
  - Tree Traversal
- **Implementation of:**
  - Inorder Traversal
  - Preorder Traversal
  - Postorder Traversal
  - Level Order Traversal (Line by Line)
  - Tree Traversal in Spiral Form

BINARY SEARCH TREE



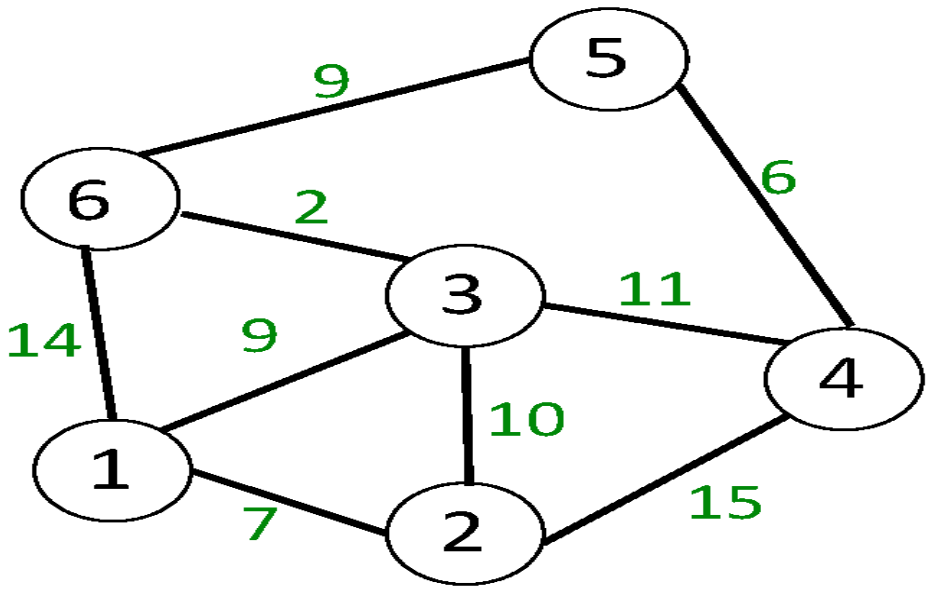
- **Background, Introduction and Application**
- **Implementation of Search in BST**
    - In CPP
    - In Java
- **Insertion in BST**
    - In CPP
    - In Java
- **Deletion in BST**
    - In CPP
    - In Java
- **Floor in BST**
    - In CPP
    - In Java
- **Self Balancing BST**
- **AVL Tree**

- **Red Black Tree**
- **Set in C++ STL**
- **Map in C++ STL**

HEAP

- **Introduction & Implementation**
- **Binary Heap**
    - Insertion
    - Heapify and Extract
    - Decrease Key, Delete and Build Heap
- **Heap Sort**
- **Priority Queue in C++**
- **PriorityQueue in Java**

GRAPH



- **Introduction to Graph**
- **Graph Representation**
    - Adjacency Matrix
    - Adjacency List in CPP and Java
    - Adjacency Matrix VS List
- **Breadth-First Search**
    - Applications
- **Depth First Search**
    - Applications
- **Shortest Path in Directed Acyclic Graph**
- **Prim's Algorithm/Minimum Spanning Tree**
    - Implementation in CPP
    - Implementation in Java
- **Dijkstra's Shortest Path Algorithm**
    - Implementation in CPP
    - Implementation in Java
- **Bellman-Ford Shortest Path Algorithm**
- **Kosaraju's Algorithm**
- **Articulation Point**
- **Bridges in Graph**
- **Tarjan’s Algorithm**

GREEDY

- **Introduction**
- **Activity Selection Problem**
- **Fractional Knapsack**
- **Job Sequencing Problem**

BACKTRACKING

- **Concepts of Backtracking**
- **Rat In a Maze**
- **N Queen Problem**

DYNAMIC PROGRAMMING

- **Introduction**
- **Dynamic Programming**
    - Memoi59zation
    - Tabulation

TREE

- **Introduction**
    - Representation
    - Search
    - Insert
    - Delete
- **Count Distinct Rows in a Binary Matrix**

SEGMENT TREE

- **Introduction**
- **Construction**
- **Range Query**
- **Update Query**

DISJOINT SET

- **Introduction**
- **Find and Union Operations**
- **Union by Rank**
- **Path Compression**
- **Kruskal's Algorithm**

REASON FOR CHOOSING DSA

All of the above was part of my training during my summer break I specially choose the DSA by Geeks for Geeks for reasons stated below :

- I was interested in Problem Solving and Algorithms since my first semester.
- Data structure is a thing you need to know no matter in which language do you code.
- One need to learn how to make algorithm of a real life problem he/she is facing.
- It had video lectures of all the topics from which one can easily learn. I prefer learning from video rather than books and notes. I know books and notes and thesis have their own significance but still video lecture or face to face lectures make it easy to understand faster as we are involved Practically.
- It had 200+ algorithmic coding problems with video explaind solutions.
- It had track based learning and weekly assesment to test my skills.
- It was a great opportunity for me to invest my time in learning instead of wasting it here and there during my summer break in this Covid-19 panademic.
- It contained a lot of knowledge for such a resonable price.
- The course was in two programing languages C++ and JAVA.
- This was a life time accessable course which I can use to learn even after my training whenever I want to revise.
- Along with all these reasons one of the reason was the Geeks for Geeks platform which is offering the course because Geeks for Geeks is one of the best platform for Computer Science Students.

LEARNING OUTCOMES

A lot of beginners and experienced programmers avoid learning Data Structures and Algorithms because it's complicated and they think that there is no use of all the above stuff in real life but there is a lot of implementation of DSA in daily life.

For Example, If we have to search our roll number in 2000 pages of Document how would we do that?

- If we try to search it randomly or in sequence it will take too much time.
- We can try another method in which we can directly go to page no. 1000 and we can see if our roll no. is there or not if not we can move ahead and by repeating this and eliminating we can search our roll no. in no time.

And this is called Binary Search Algorithm.

Two reasons to Learn Data Structure and Algorithms -

- If you want to crack the interviews and get into the product based companies
- If you love to solve the real-world complex problems.

I have learnt a vast number of topics like Trees, Graphs, Linked Lists, Arrays, etc. I understood their basics, there working, there implementation, and their practical use in the problems we face while we solve a problem using coding.

When we work in IT sector (Software or Programing part to be specific) we need to solve the problems and make programs write tons of code which will help us with the given problem and to write a program one need to make different algorithms. Many algorithms combine to make a program. Now, algorithm are writen in some lenguages but they are not dependen ton them, one need to make a plan and algo first then write it into any language wether i tis C++ or JAVA or C or any other programing language. Algorith is based on data structure and its implementation and working. So, basiclly one need to have a good grip on DSA to work in programing sector.

When you ask someone to make a decision for something the good one will be able to tell you "*I chose to do X because it's better than A, B in these ways. I could have gone with C, but I felt this was a better choice because of this*". In our daily life, we always go with that person who can complete the task in a short amount of time with efficiency and using fewer resources. The same things happen with these companies. The problem faced by these companies is much harder and at a much larger scale. Software developers also have to make the right decisions when it comes to solving the problems of these companies.
Knowledge of data structures like Hash Tables, Trees, Tries, Graphs, and
various algorithms goes a long way in solving these problems efficiently and the interviewers are more interested in seeing how candidates use these tools to solve a problem.

I learned about how to break a problem into pieces and then find the solution then how to maket he desired algorithm which will help me to solve my respective problem.

What I Learned from the course precisely :

- I Learned Data Structures and Algorithms from basic to advanced level.
- Learned Topic-wise implementation of different Data Structures & Algorithms.
- Improved my problem-solving skills to become a stronger developer.
- Developed my analytical skills on Data Structures and use them efficiently.
- Solved problems asked in product-based companies' interviews.
- Solved problems in contests similar to coding round for SDE role.

This will help me during my career as a programmer and afterwards also whenever I need to code. We are surrounded by a lot of real-world complex problems for which no one has the solution. Observe the problems in-depth and you can help this world giving the solution which no one has given before.

" Data structure and algorithms help in understanding the nature of the problem at a deeper level and thereby a better understanding of the world. "

BIBLIOGRAPHY

- DSA Books
- Geeks for Geeks website
- Geeks for Geeks Course
- Oracle java documentation

Mini Project

Project Undertaken:

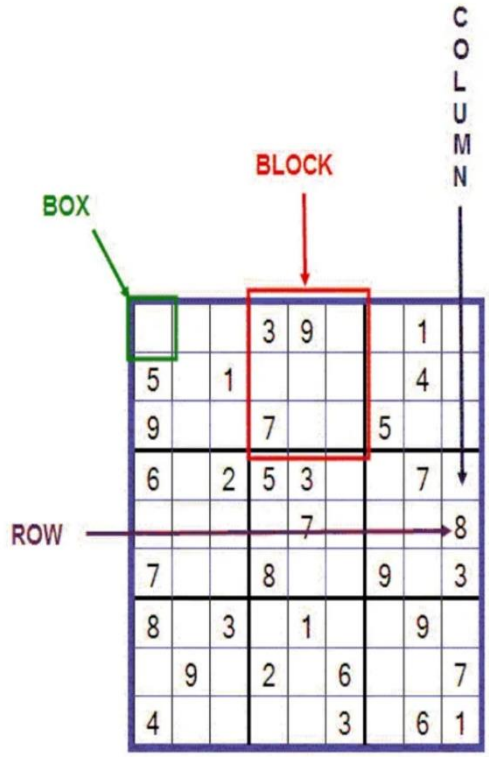
Sudoku Solver:

This project is about solving a Sudoku problem by a particular method which given a correct solution for given problem.

I work very hard to complete this project and able to build logic that can find out the solution of Sudoku problem and it generate solution for some multiple grids.

RULES:

Fill a given grid(9x9) with the number 1 to 9, So that every column, row, and 3x3 box the numbers 1 to 9, keeping in mind that the same number doesn't repeat in that particular row, column or the 3x3 box.



Abhishek Sharma (Sudoku Solver)

Solve!

Clear board

Type any numbers and Click on Solve!

Abhishek Sharma (Sudoku Solver)

1								
	2			4			3	
		3						
			4					
	6			5			7	
					6			
						7		
	1			2			8	
								9

Solve!

Clear board

Abhishek Sharma (Sudoku Solver)

1	4	5	2	3	7	6	9	8
6	2	7	8	4	9	1	3	5
8	9	3	1	6	5	2	4	7
2	3	1	4	7	8	9	5	6
4	6	8	9	5	2	3	7	1
5	7	9	3	1	6	8	2	4
3	8	4	5	9	1	7	6	2
9	1	6	7	2	4	5	8	3
7	5	2	6	8	3	4	1	9

Solve!

Clear board

Learning Outcome from Mini Project

My project uses five main modules:

1. Reading the input from file.
2. Search for patterns.
3. Provide solution.
4. Compare the run-time.
5. Solve by Backtracking.

BACKTRACKING ALGORITHM

Backtracking is an algorithmic technique for solving problems recursively by trying to build a solution incrementally, one piece at a time, removing those solutions that fail to satisfy the constraints of the problem at any point of time (by time, here, is referred to the time elapsed till reaching any level of the search tree).

Backtracking is a technique to solve problems where multiple choices are there and we don't know the correct choice and hence we solve problem with trial and error i.e. trying each option until goal is achieved.

We basically check that the same number is not present in current row, current column and current 3x3 subgrid. After checking for safety, we assign the number, and recursively check whether this assignment leads to a solution or not. If the assignment doesn't lead to a solution, then we try next number for current empty cell. And if none of the number (1 to 9) lead to solution, we return false.

Conclusion:

As you've seen, data structures are the essential building blocks that we use to organize all of our digital information. Choosing the right data structure allows us to use the algorithms we want and keeps our code running smoothly. Understanding data structures and how to use them well can play a vital role in many situations including:

1.technical interviews in which you may be asked to evaluate and determine runtime for data structures given specific algorithms

2.day-to-day work for many software engineers who manipulate data stored in structures

3.data science work where data is stored and accessed

References:

https://practice.geeksforgeeks.org/batch/dsa-4