SIX WEEKS SUMMER TRAINING

REPORT

On

DSA Self-paced

Submitted by

Akash Tripathi

Registration No. **12101602**

Program Name: **Bachelor of Technology**

Under the Guidance of

Mr. Sandeep Jain

Geeks For Geeks

School of Computer Science & Engineering

Lovely professional University, Phagwara

(May-July, 2022)

DECLARATION

I hereby declare that I have completed my six weeks summer training at DSA Self paced from 25th May 2022 to 10th July 2022 under the guidance of Mr. Sandeep Jain. I have declare that I have worked with full dedication during these six weeks of training and my learning outcomes fulfil the requirements of training for the award of degree of Bachelor of Technology, Lovely Professional university, Phagwara.

Akash Tripathi

Name of Student: Akash Tripathi

Registration no: 12101602

Date: 23th Aug, 2022

Acknowledgement

It is with sense of gratitude; I acknowledge the efforts of entire hosts of well-wishers who have in some way or other contributed in their own special ways to the success and completion of the Summer Training.

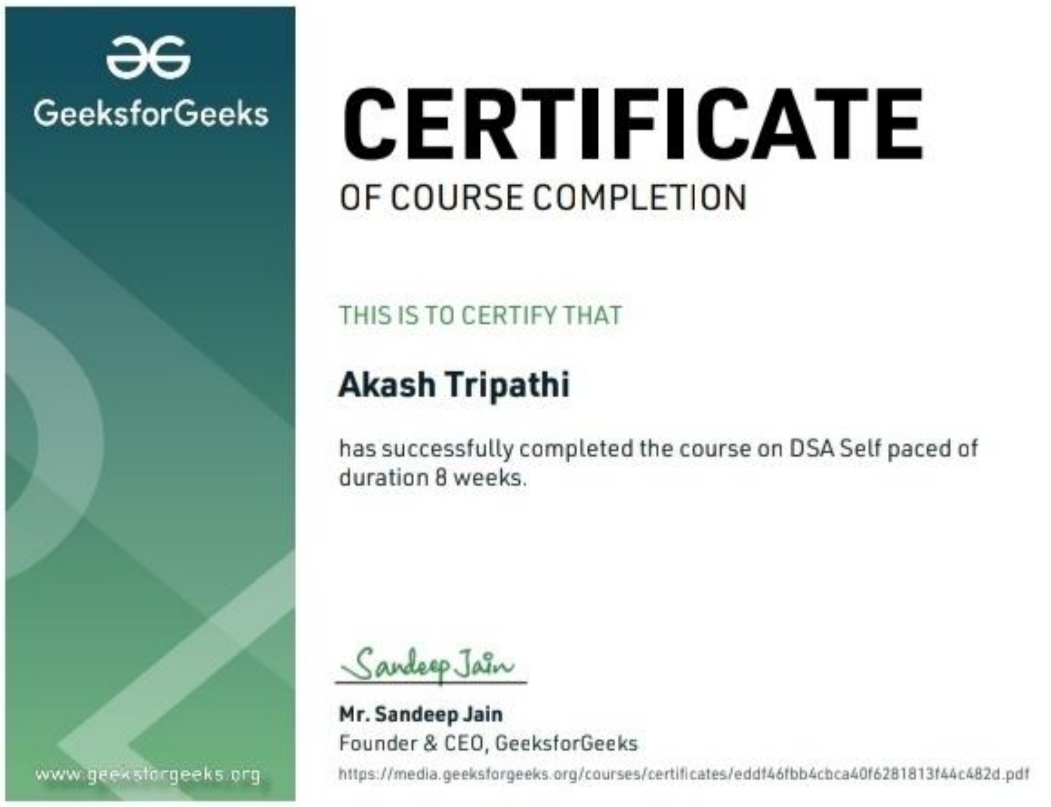
Successfully completion of any type technology requires helps from a number of people. I have also taken help from different people for the preparation of the report. Now, there is little efforts to show my deep gratitude to those helpful people.

I would like to also thank my own college Lovely Professional University for offering such a course which not only improve my programming skill but also taught me other new technology.

Then I would like to thank my parents and friends who have helped me with their valuable suggestions and guidance for choosing this course.

Last but not least I would like to thank my all classmates who have helped me a lot.

Training certificate from organization



GeeksforGeeks

CERTIFICATE

OF COURSE COMPLETION

THIS IS TO CERTIFY THAT

Akash Tripathi

has successfully completed the course on DSA Self paced of duration 8 weeks.

Sandeep Jain

Mr. Sandeep Jain

Founder & CEO, GeeksforGeeks

https://media.geeksforgeeks.org/courses/certificates/eddf46fbb4cbca40f6281813f44c482d.pdf

www.geeksforgeeks.org

Table Of Contents

Introduction

The course name DSA stands for "Data Structures and Algorithms" and Self-paced means, one can join the course anytime. All of the content will be available once one gets enrolled. One can finish it at his own decided speed.

1. **What is Data Structure?**

   A data structure is a storage that is used to store and organize data. It is a way of arranging data on a computer so that it can be accessed and updated efficiently. A data structure is not only used for organizing the data. It is also used for processing, retrieving, and storing data. There are different basic and advanced types of data structures that are used in almost every program or software system that has been developed. So we must have good knowledge about data structures.

2. **What is Algorithm?**

   An algorithm is a finite set of instructions or logic, written in order, to accomplish a certain predefined task. Algorithm is not the complete code or program, it is just the core logic(solution) of a problem, which can be expressed either as an informal high-level description as pseudocode or using a flowchart.

This course is a complete package that helped me learn Data Structures and Algorithms from basic to an advanced level. The course curriculum has been divided into 8 weeks where one can practice questions & attempt the assessment tests according to his own pace. The course offers me a wealth of programming challenges that will help me to prepare for interviews with top-notch companies like Microsoft, Amazon, Adobe etc.

Technology Learnt

- Learn Data Structures and Algorithms from basic to an advanced level like:
- Learn Topic-wise implementation of different Data Structures & Algorithms as follows

1. Introduction

- **Asymptotic Notations**

  Asymptotic notations are mathematical tools to represent the time complexity of algorithms for asymptotic analysis.

- **Worst, Average and Best-Case Time Complexities**
  It is important to analyze an algorithm after writing it to find it's efficiency in terms of time and space in order to improve it if possible.

  When it comes to analyzing algorithms, the asymptotic analysis seems to be the best way possible to do so. This is because asymptotic analysis analyzes algorithms in terms of the input size. It checks how are the time and space growing in terms of the input size.

- Analysis of Loops

2. Mathematics

- Finding number of Digits in a Number
- Arithmetic and Geometric Progressions
- Quadratic Equations
- Mean and Median
- Prime Numbers
- LCM and HCF
- Factorials
- Permutation and Combinations Basics
- Modular Arithmetic

3. Bit Magic

- Binary Representation
- Set and Unset

- Toggling
- Bitwise Operators
- Algorithms

Objective: The objective of this track is to familiarize the learners with Bitwise Algorithms which can be used to solve problems efficiently and some interesting tips and tricks using Bit Algorithms.

4. Recursion

- **Recursion Basics**

  The process in which a function calls itself directly or indirectly is called recursion and the corresponding function is called as recursive function. Using recursive algorithm, certain problems can be solved quite easily. Examples of such problems are Towers of Hanoi (TOH), Inorder/Preorder/Postorder Tree Traversals, DFS of Graph, etc.
- Basic Problems on Recursion
- **Tail Recursion**

  A recursive function is said to be following Tail Recursion if it invokes itself at the end of the function. That is, if all of the statements are executed before the function invokes itself then it is said to be following Tail Recursion.

```
void printN(int N)
{
   if(N==0)
      return;
   else
      cout<<N<<" ";

   printN(N-1);
}
```

  The above function call for N = 5 will print:

  5 4 3 2 1
- Explanation of Subset Generation Problem
- Explanation of Josephus Problem
- Explanation of permutations of a string

We iterate from first to last index. For every index i, we swap the i-th character with the first index. This is how we fix characters at the current first index, then we recursively generate all permutations beginning with fixed characters (by parent recursive calls). After we have recursively generated all permutations with the first character fixed, then we move the first character back to its original position so that we can get the original string back and fix the next character at first position.

6. Arrays

- Introduction to Arrays
- Insertion and Deletion in Arrays
- Array Rotation
- Reversing an Array
- Sliding Window Technique
- Prefix Sum Array
- Implementing Arrays in C++ using STL
- Iterators in C++ STL
- Implementing Arrays in Java
- Sample Problems on Array
- XOR Linked List - A Memory Efficient Doubly Linked List | Set 1

5. Searching

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

6. Sorting

- Implementation of C++ STL sort() function in Arrays and Vectors
- Sorting in Java
- Arrays.sort() in Java
- Collection.sort() in Java
- Stability in Sorting Algorithms
- Insertion Sort
- Merge Sort
- Quick Sort
- Overview of Sorting Algorithms

7. Matrix

- 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

8. 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

9. Strings

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

10. Linked List

- Representation And Implementation
- Doubly Linked List
- Circular Linked List

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

11.Stack

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

12.Queue

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

13.Deque

- Introduction and Application
- Implementation
    - In C++ STL 14
    - In Java
- Problems (With Video Solutions)
    - Maximums of all subarrays of size k
    - Array Deque in Java
    - Design a DS with min max operations

14.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

15.Binary Search Tree

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

16.Heap

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

17.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

- Count Distinct Rows in a Binary Matrix

23.Segment Tree

- Introduction
- Construction
- Range Query
- Update Query

24.Disjoint Set

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

1. Improved my problem-solving skills by practicing problems to become a stronger developer
2. Developed my analytical skills on Data Structures to use them efficiently
3. Solved problems asked in product-based companies’ interviews
4. Solved problems in contests similar to coding round for SDE role

Reason for choosing this technology

With advancement and innovation in technology, programming is becoming a highly in-demand skill for Software Developers. Everything you see around yourself from Smart TVs, ACs, Lights, Traffic Signals uses some kind of programming for executing user commands.

Data Structures and Algorithms are the identity of a good Software Developer. The interviews for technical roles in some of the tech giants like Google, Facebook, Amazon, Flipkart is more focused on measuring the knowledge of Data Structures and Algorithms of the candidates. The main reason behind this is Data Structures and Algorithms improves the problem-solving ability of a candidate to a great extent.

1. This course has 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.
2. It has 200+ algorithmic coding problems with video explained solutions.
3. It has track based learning and weekly assessment to test my skills.
4. 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 pandemic.
5. This was a life time accessible course which I can use to learn even after my training whenever I want to revise.

Profile of the Problem

The old manual system was suffering from a series of drawbacks. Since whole of the system was to be maintained with hands the process of keeping, maintaining and retrieving the information was very tedious and lengthy. The records were never used to be in a systematic order. there used to be lots of difficulties in associating any particular transaction with a particular context. If any information was to be found it was required to go through the different registers, documents there would never exist anything like report generation. There would always be unnecessary consumption of time while entering records and retrieving records. One more problem was that it was very difficult to find errors while entering the records. Once the records were entered it was very difficult to update these records.

The reason behind it is that there is lot of information to be maintained and have to be kept in mind while running the business .For this reason we have provided features Present system is partially automated (computerized), actually existing system is quite laborious as one has to enter same information at three different places.

Problem Analysis

1. Product definition

The old manual system was suffering from a series of drawbacks. Since whole of the system was to be maintained with hands the process of keeping, maintaining and retrieving the information was very tedious and lengthy. The records were never used to be in a systematic order. there used to be lots of difficulties in associating any particular transaction with a particular context. If any information was to be found it was required to go through the different registers, documents there would never exist anything like report generation. There would always be unnecessary consumption of time while entering records and retrieving records. One more problem was that it was very difficult to find errors while entering the records. Once the records were entered it was very difficult to update these records.

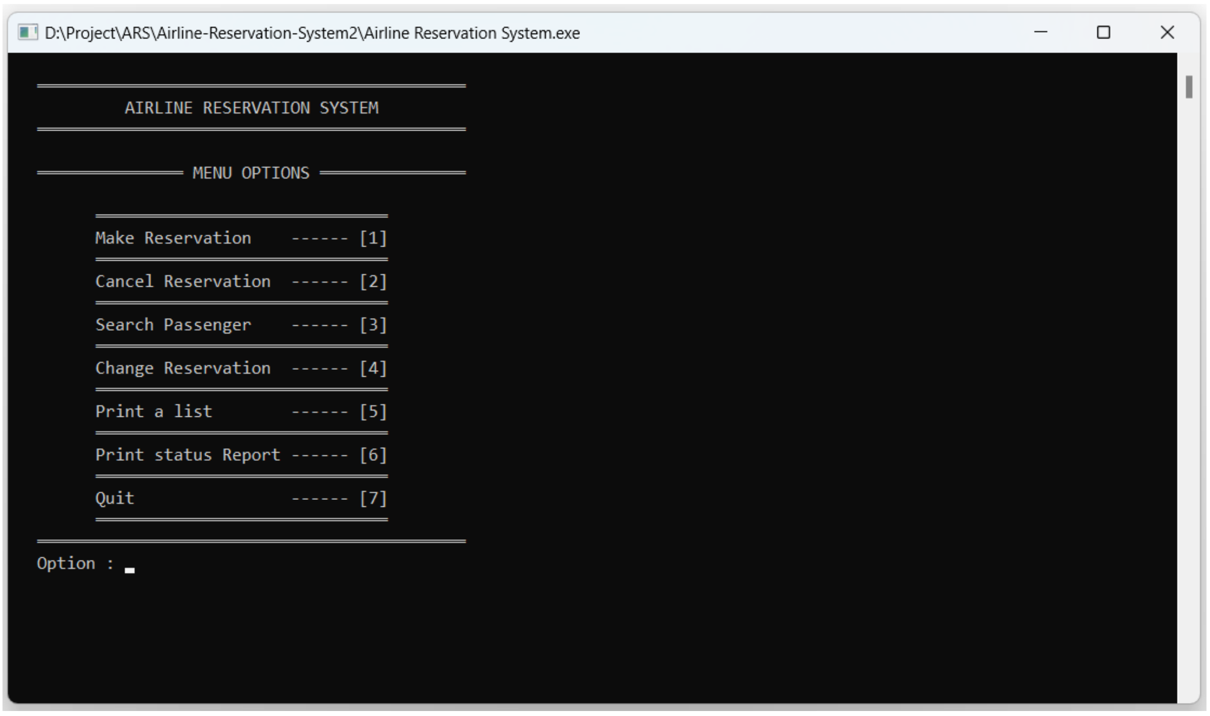
2. Feasibility Analysis

After doing the project Flight Ticket Booking Sytem, study and analyzing all the existing or required functionalities of the system, the next task is to do the feasibility study for the project. All projects are feasible - given unlimited resources and infinite time. Feasibility study includes consideration of all the possible ways to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements.

Software Requirements –

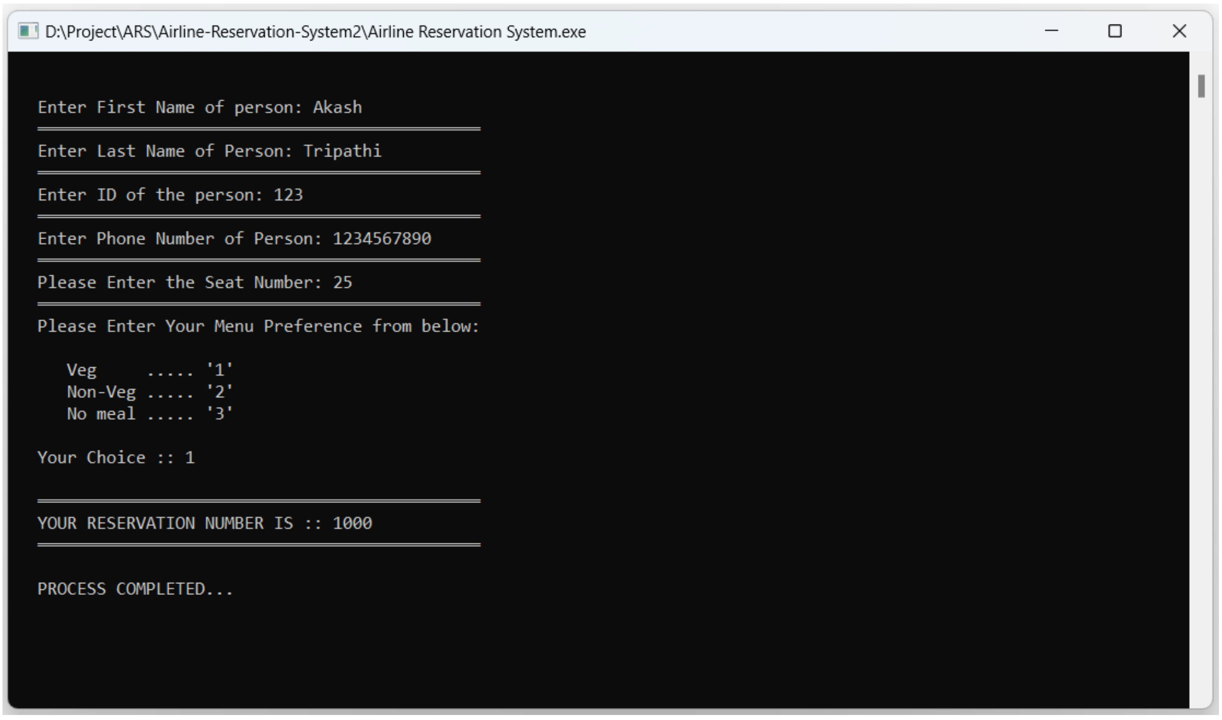
- Visual Studio
- G++ extension

- **Implementation**



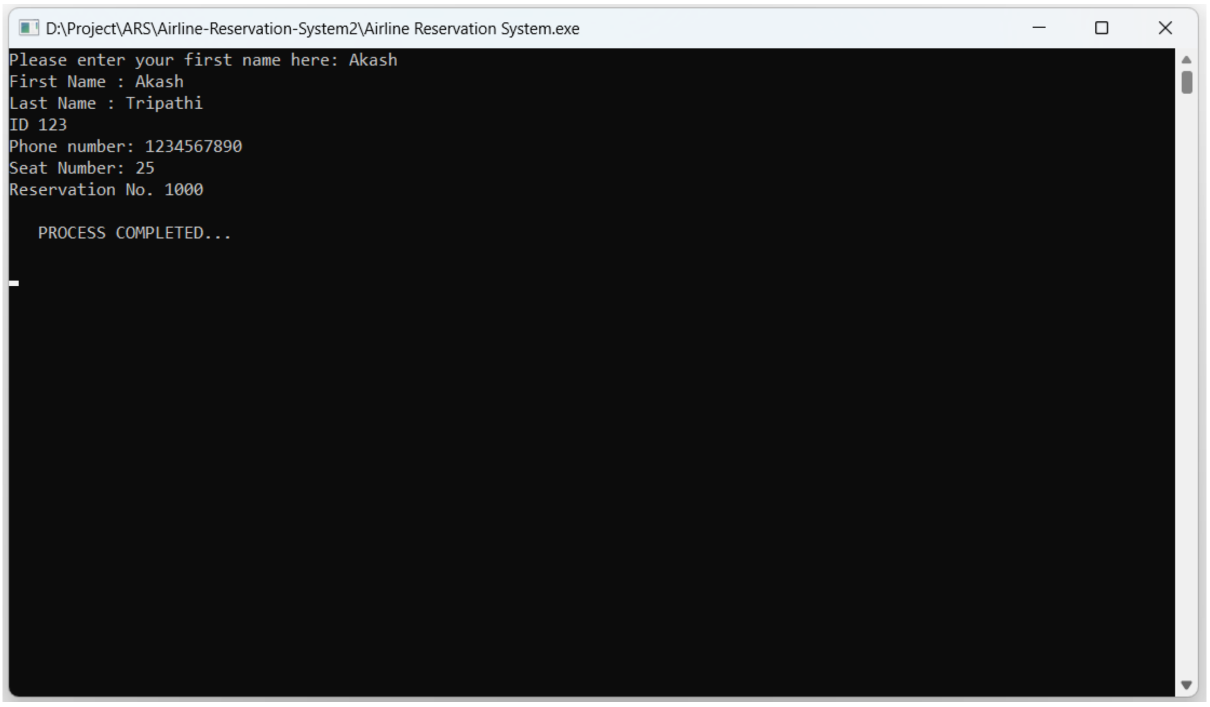
D:\Project\ARS\Airline-Reservation-System2\Airline Reservation System.exe

```
═══════════════════════════════════════════
         AIRLINE RESERVATION SYSTEM
═══════════════════════════════════════════

══════════════ MENU OPTIONS ══════════════

      ═══════════════════════════════
      Make Reservation    ------ [1]
      ═══════════════════════════════
      Cancel Reservation  ------ [2]
      ═══════════════════════════════
      Search Passenger    ------ [3]
      ═══════════════════════════════
      Change Reservation  ------ [4]
      ═══════════════════════════════
      Print a list        ------ [5]
      ═══════════════════════════════
      Print status Report ------ [6]
      ═══════════════════════════════
      Quit                ------ [7]
      ═══════════════════════════════

═══════════════════════════════════════════
Option : _
```

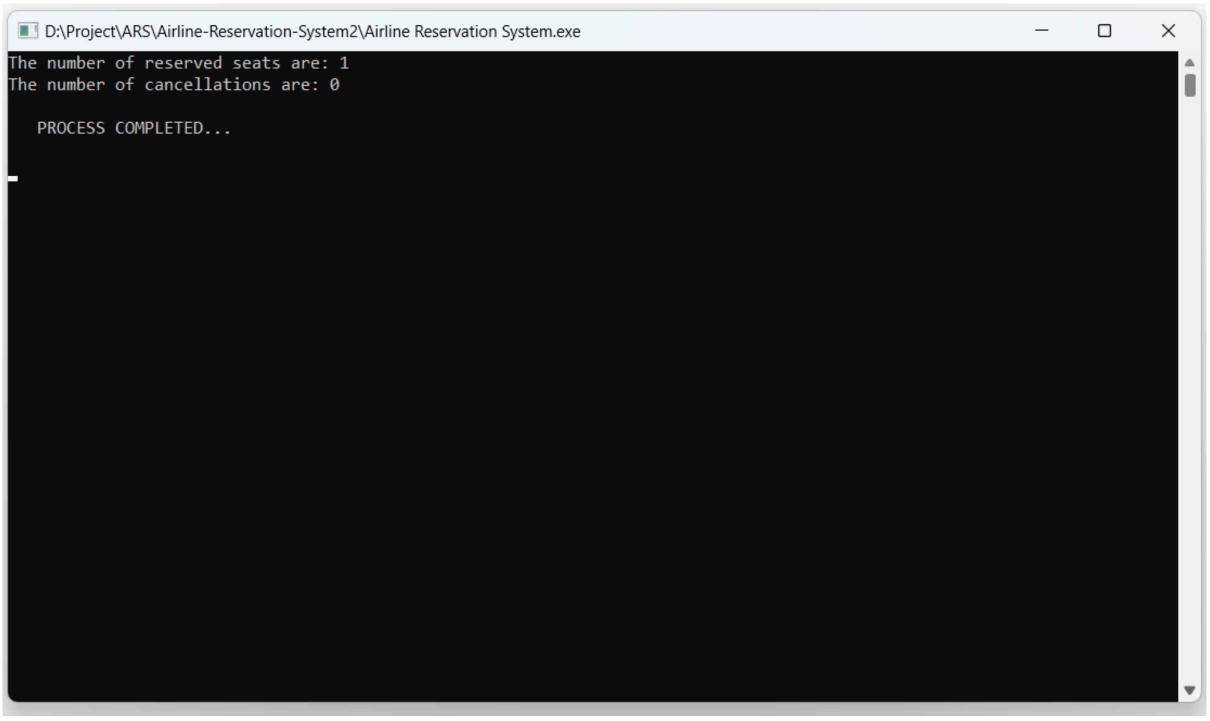


D:\Project\ARS\Airline-Reservation-System2\Airline Reservation System.exe

```
Enter First Name of person: Akash
═════════════════════════════════════════════
Enter Last Name of Person: Tripathi
═════════════════════════════════════════════
Enter ID of the person: 123
═════════════════════════════════════════════
Enter Phone Number of Person: 1234567890
═════════════════════════════════════════════
Please Enter the Seat Number: 25
═════════════════════════════════════════════
Please Enter Your Menu Preference from below:

   Veg     ..... '1'
   Non-Veg ..... '2'
   No meal ..... '3'

Your Choice :: 1

═════════════════════════════════════════════
YOUR RESERVATION NUMBER IS :: 1000
═════════════════════════════════════════════

PROCESS COMPLETED...
```


D:\Project\ARS\Airline-Reservation-System2\Airline Reservation System.exe

```
Please enter your first name here: Akash
First Name : Akash
Last Name : Tripathi
ID 123
Phone number: 1234567890
Seat Number: 25
Reservation No. 1000

   PROCESS COMPLETED...
```


D:\Project\ARS\Airline-Reservation-System2\Airline Reservation System.exe

```
The number of reserved seats are: 1
The number of cancellations are: 0

   PROCESS COMPLETED...
```

Learning Outcomes

Programming is all about data structures and algorithms. Data structures are used to hold data while algorithms are used to solve the problem using that data.

Data structures and algorithms (DSA) goes through solutions to standard problems in detail and gives you an insight into how efficient it is to use each one of them. It also teaches you the science of evaluating the efficiency of an algorithm. This enables you to choose the best of various choices.

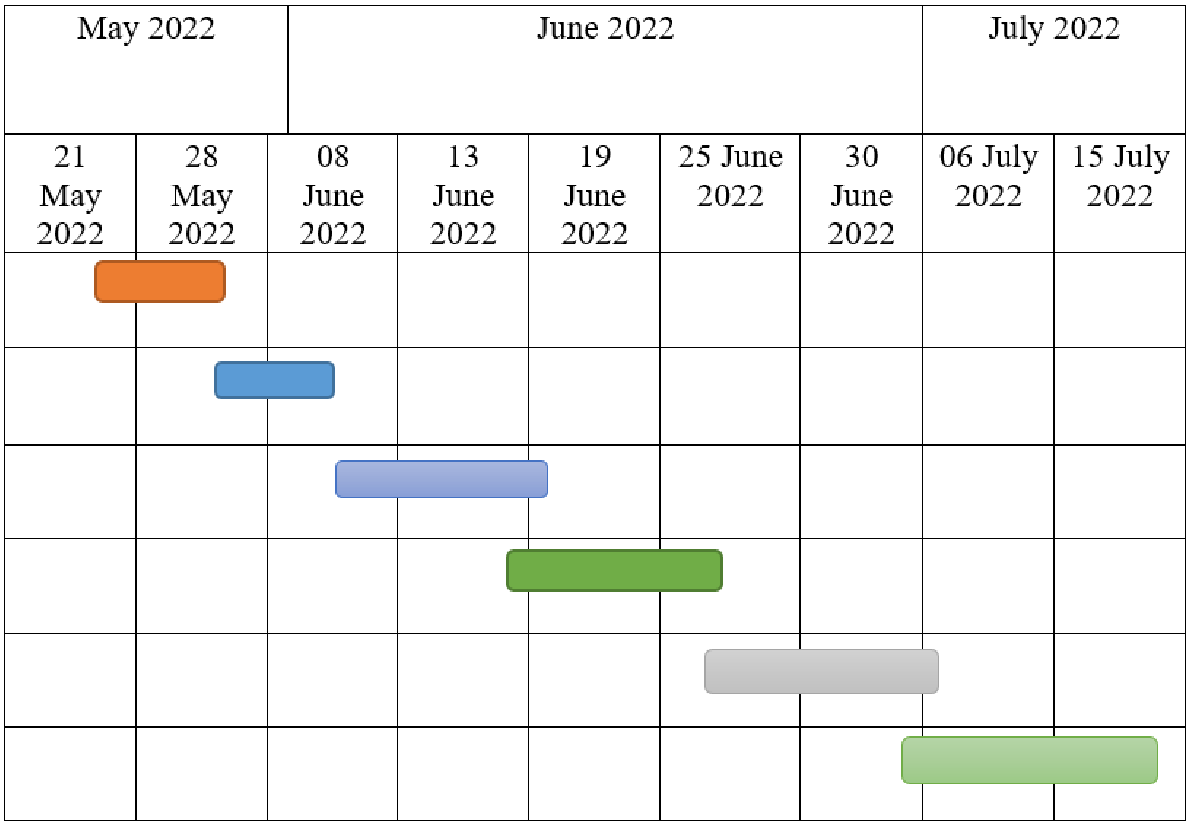
For example, you want to search your roll number in 30000 pages of documents, for that you have choices like Linear search, Binary search, etc.

So, the more efficient way will be Binary search for searching something in a huge number of data. So, if you know the DSA, you can solve any problem efficiently.

The main use of DSA is to make your code scalable because

- Time is precious
- Memory is expensive

Gantt Chart


May 2022
June 2022
July 2022
21 May 2022
28 May 2022
08 June 2022
13 June 2022
19 June 2022
25 June 2022
30 June 2022
06 July 2022
15 July 2022

Bibliography

- Google
- GeeksforGeeks
- Tutorials Point
- W3School
- Javatpoint
- Programiz
- YouTube