Project Report

Design and Analysis of Algorithms CSE-5311

Implementation and comparison of various search algorithms

Prof. Negin Fraidouni, PH.D.

By,
Name: Dannasri Srinivasan

UTA ID: 1001698730

Project Abstract :

The crux of this project is to implement various search algorithms.
An algorithm when given an input of a list of numbers gives a search element or index of the search element as output.

Efficiency is one of the main evaluation features of an algorithm, this includes efficiency in memory and runtime. This project is designed to display the important properties of different search techniques including their complexity, stability, and memory constraints. We have articulated Time complexity analysis on various search techniques.

Implementation:

I have implemented the following search algorithms:

1. Linear Search
2. Binary Search
3. Binary Search Tree
4. Red Black Tree

The data structures used in the project are Array List, Set (LinkedHashSet) and Tree.

Specifically,

Algorithm	Data Structure
Linear Search	ArrayList
Binary Search	ArrayList
Binary Search Tree	ArrayList, Tree
Red Black Tree	ArrayList, Tree

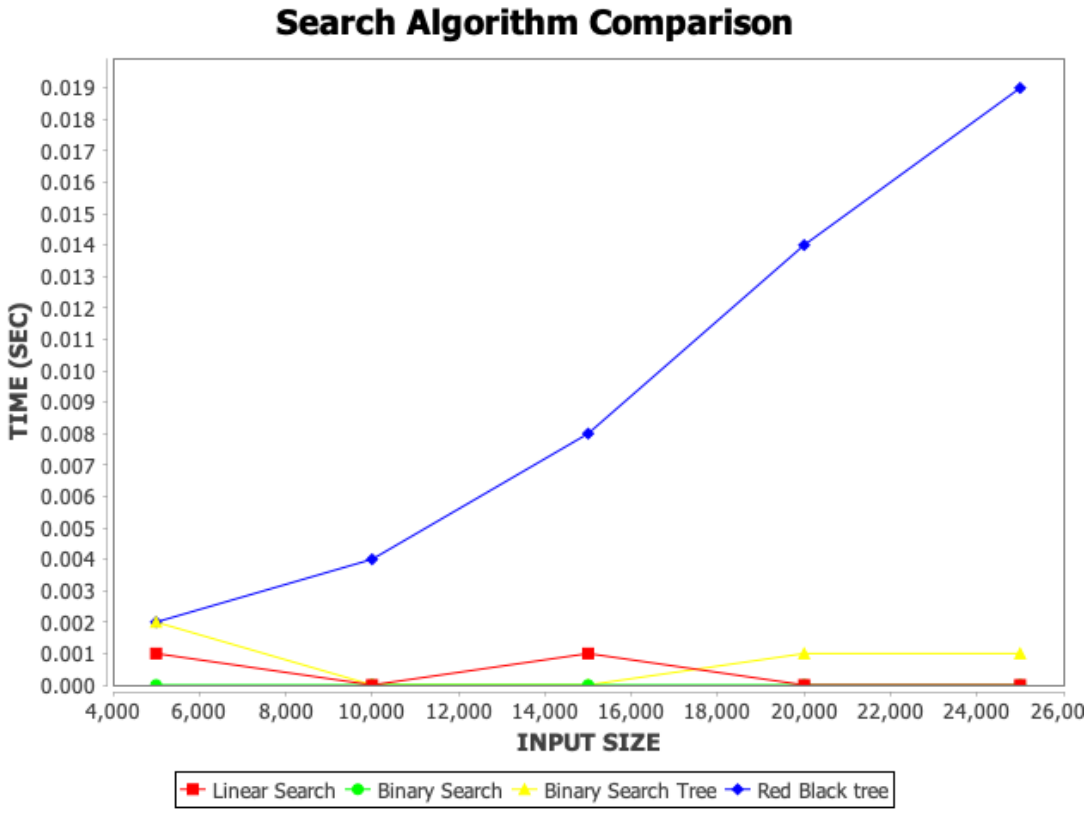
Components:

The main components of this project are Search Algorithms that are implemented from scratch and compared upon, Jfree chart library is focuses on Time-Complexity plotting and finally, using system time for finding the time complexity.

Inputs are fed through random generators to the algorithms. The random generator generates input based on the file size. I used Set data structure to avoid duplicates in my inputs. The inputs will create in such a way to analyze the behavior of the algorithms in various situations.

I implemented the project using Java programming language. I have used Terminal, and Eclipse IDE for coding and to run the search algorithms. For Graphical User Interface, I used HTML, PHP and XAMPP server. For generating graph, Jfree graph package is used to plot Run-Time Vs Input-Size. System time is used to find the time the algorithm takes to implement.

Run Time vs Input size:



Observation :

The run times of different algorithms with varying data sizes are tabulated as follows

	Input with1500 numbers	Input with 15000 numbers	Input with 1,50,000 numbers	Input file 15,00,000 numbers
Linear Search	0.001	0.001	0.002	0.006
Binary Search	0.001	0.001	0.001	0.001
Binary Search Tree	0.0	0.002	0.016	0.093
Red black Tree	0.001	0.005	0.039	0.677

The best, average and worst case scenarios are as follows

Algorithm	Best	Average	Worst
Linear Search	Omega (1)	Theta (N)	O(N)
Binary Search	Omega (1)	Theta (N)	O(N)
Binary Search Tree	Omega (1)	Theta (N)	O(N)
Red black Tree	Omega (1)	Theta (N)	O(N)

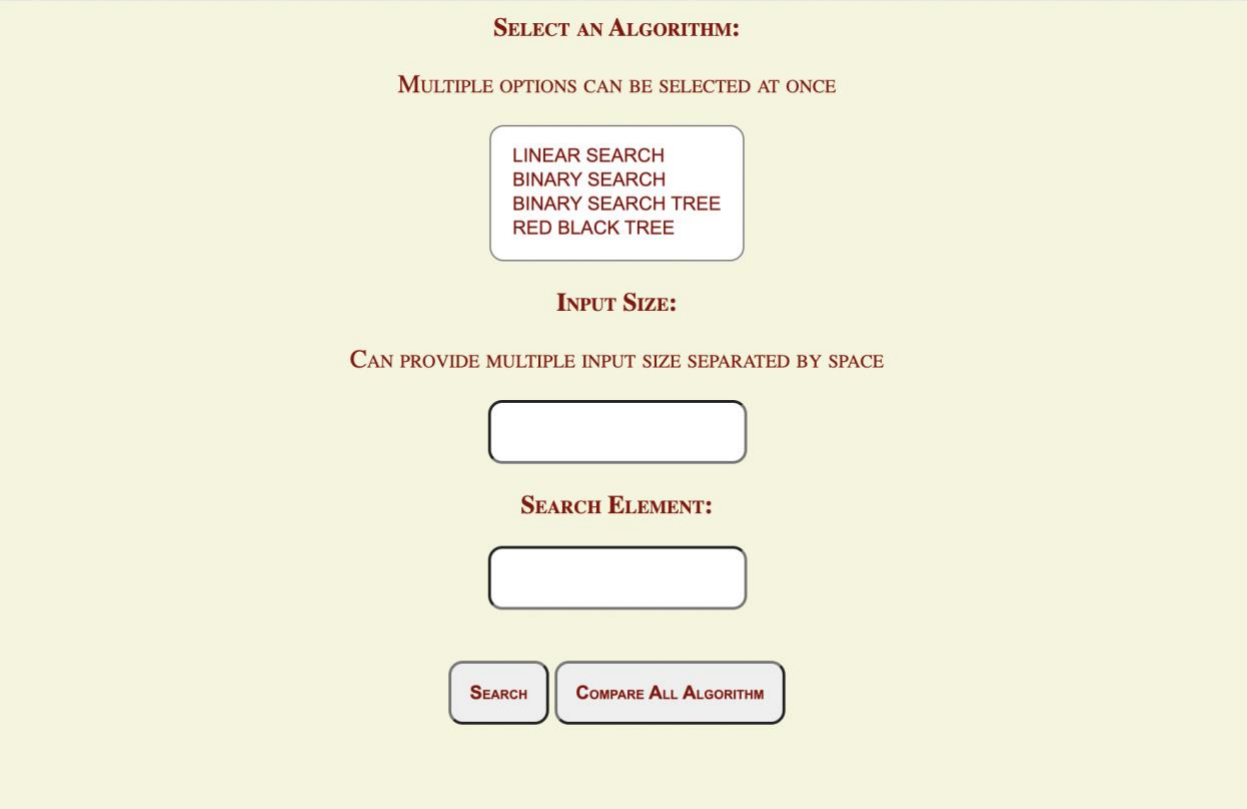
On observing the above obtained tables, we come to know that binary search is the best search algorithm in terms of its time complexity. Binary search tree and red black will work fast for small inputs. Mostly Binary search tree will fail when it comes to large inputs. Red black tree will take more time as it forms tree and perform rotation in order to maintain balanced tree.

Tools and Language used:

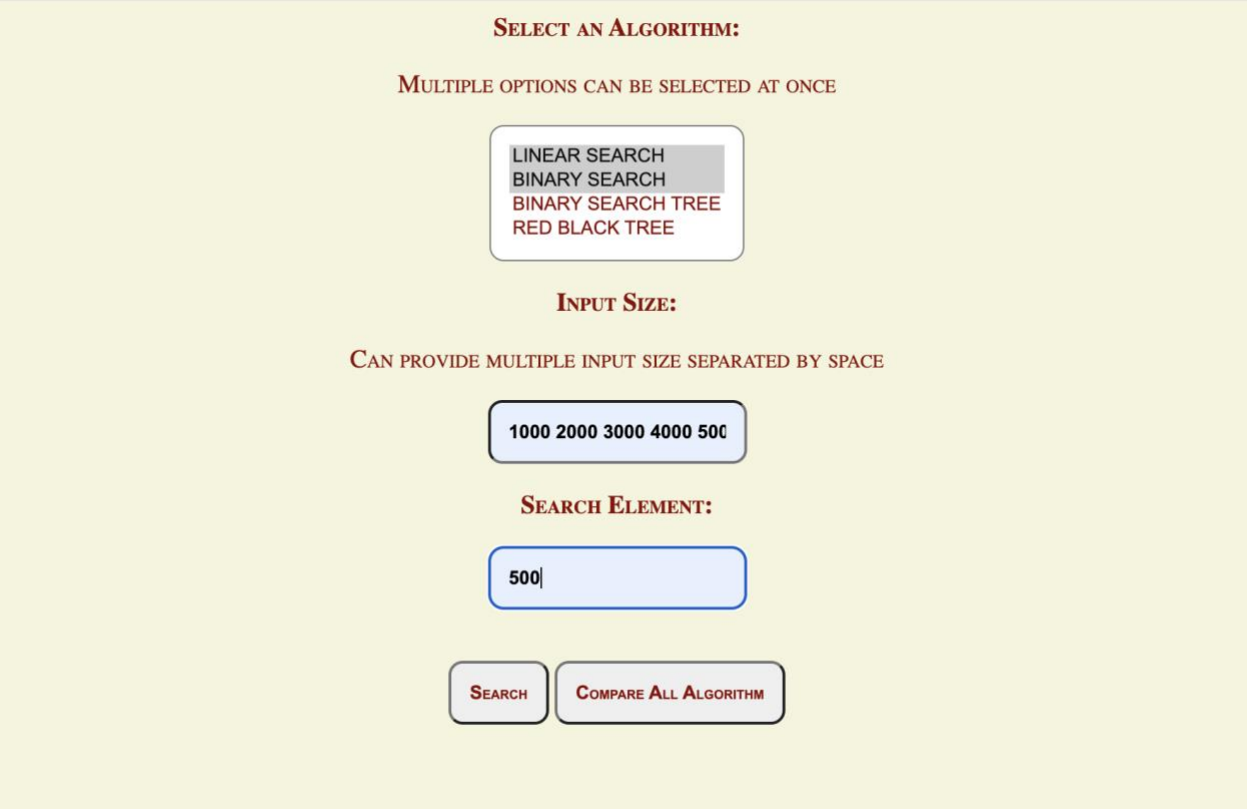
Eclipse, Java, Terminal, JFree library for plotting the complexity, Html, PHP and XAMPP for GUI.

GUI:

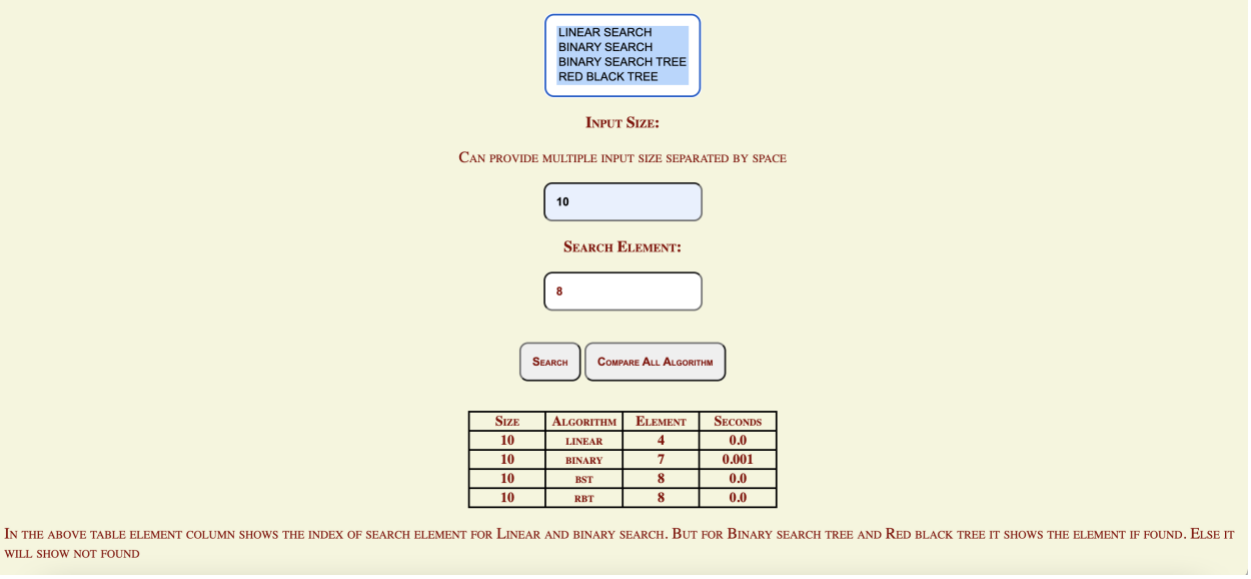
In GUI, we can provide type of algorithms, search element and multiple input size. We can perform search operation with multiple algorithms, or we can compare multiple algorithms. Compare will show difference in runtime and a graph with input size vs run time complexity.



Input can be given as below,



And the output will be,



Conclusion:

This project gives an insight on the various search algorithms and its corresponding best case, average case and worst case time complexities.

By analyzing and plotting the Run-time Vs Input-Size graph, we come to know the which algorithm to be used given a scenario, by analyzing input size and time taken.