Department of Computer Science, IBA

Data Structures (3+1) Instructor: Quratulain

Algorithm Complexity

Iterative and recursive

Objective

The objective of this lab is to understand how we can analyze the time complexity of an iterative and recursive algorithms.

Searching Algorithms:

Given a sorted array of n integers and target value, determine if the target exits in the array or not using following algorithms.

1. Linear Search algorithm – iterative and recursive implementations
2. Binary Search algorithm - iterative and recursive implementations

Further Analysis test

1. Use system time in main method before and after calling each search algorithm to understand the exact time taken by these algorithms. To take system time use System.nanoTime();

   For example in main method do as follows :

   ```
   Long  startTime= System.nanoTime();
         LinearSearch(arr, value);
   Long  endTime= System.nanoTime();
   System.out.Print(endTime-startTime);
   ```

   Note: Since our algorithms are dependent on system resources that are manage by operating system therefore the time taken by same algorithm with same value is not consistent and thus rather than using system time, count the number of operations in each algorithm to estimate the performance.

2. To count the number of operations in search algorithms create an integer variable and increase its value after each comparison. The total number of comparisons are used to analyze the growth rate of each algorithm as the data size gets larger. For recursive solution count number of function calls.
   Create a table as shown below in excel sheet and draw a graph to show a growth rate of each algorithm. Your algorithm must validate the growth rate of linear in linear search and logarithmic in binary search algorithm as discussed in theory.

N	Binary Search Total comparisons	Linear Search Total comparisons
10	3	10
100	7	100
1000	--	--
5000	--	--

********** Good Luck **********