STUDENT LESSON

A19 – Sequential & Binary Searches

STUDENT LESSON

A19 – Sequential & Binary Searches

INTRODUCTION: Searching for an item is a very important algorithm to a computer scientist. What makes computers tremendously valuable is their ability to store and search for information quickly and efficiently. For example, Internet search engines process billions of pages of information to help determine the most appropriate resources for users, and a word processor's spell-checking feature enables quick searching of large dictionaries. In this lesson, you will learn about a simple sequential search and the very efficient binary search.

The key topics for this lesson are:

A. Sequential Search
B. Binary Search
C. Recursive vs. Non-recursive Algorithms

VOCABULARY: BINARY SEARCH SEQUENTIAL SEARCH

DISCUSSION:

A. Sequential Search

1. Searching a linear data structure, such as an array, can be as simple and straightforward as checking through every value until you find what you are looking for. A sequential search, also known as a *linear search*, involves starting at the beginning of a list or sequence, sorted or not, and searching one-by-one through each item, in the order they exist in the list, until the value is found.

2. This unsophisticated approach is appropriate for small lists or unordered lists.

3. The order of a sequential search is linear, O(N).

B. Binary Search

1. The word binary (from the word two) refers to anything with two possible options or parts. A binary search involves binary decisions – decisions with two choices.

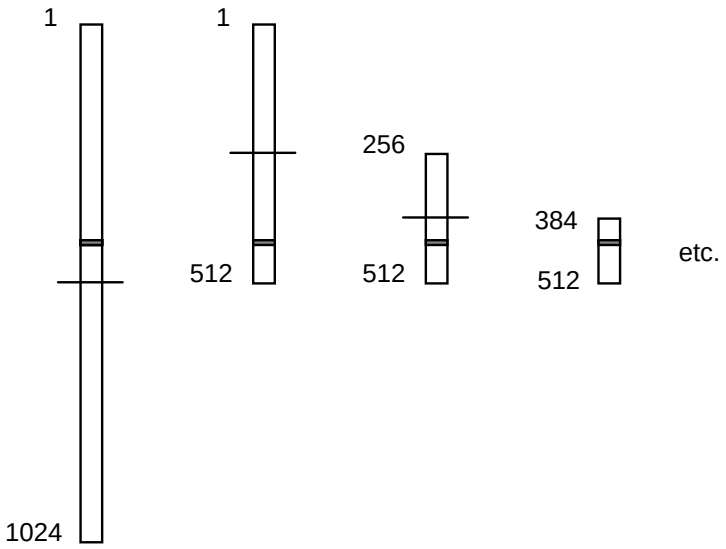
2. The underlying idea of a binary search is to divide one's data in half and to examine the data at the point of the split. If the data is sorted, it's very easy and efficient to ignore one half or the other half of the data, depending on where the value that is being searched is located.

3. Assuming that a list is already sorted, a target value is searched for by repeating the following steps:

a. Divide the list in half.

b. Examine the value in the middle of the list. Is the target value equal to the value there, or does it come *before* or *after* the center value? If the target value comes before or after, then return to step a, to repeat the process with the halved list where the target value is located.

4. For example, with a list of 1,024 sorted values, we happen to be searching for a target value that is stored in position 492. Using a binary search, the list of 1,024 is split in half; because the target value is not found in position 512, we proceed to search the first half (and discard the values in the second half). Within the sublist from 1…512, we do another binary search sequence: split; examine; and binary search again.



5. The speed of a binary search comes from the elimination of half of the data set each time. If each arrow below represents one binary search process, only ten steps are required to search a list of 1,024 numbers:

`1024 → 512 → 256 → 128 → 64 → 32 → 16 → 8 → 4 → 2 → 1`

The $\log_2 1024$ is 10. In a worst-case scenario, if the size of the list doubled to 2,048, only one more step would be required using a binary search.

The efficiency of a binary search is illustrated in this comparison of the number of entries in a list and the number of binary divisions required.

Number of Entries	Number of Binary Divisions
1,024	10
2,048	11

4,096	12
…	…
32,768	15
…	…
1,048,576	20
N	$\log_2 N$

6. The order of a binary search is O(log2N).

C. Recursive vs. Non-recursive Algorithms

1. The binary search algorithm can be coded recursively or non-recursively. Here are some arguments for each method.

2. A non-recursive version requires less memory and fewer steps by avoiding the overhead of making recursive calls.

3. However, the recursive version is somewhat easier to understand and code and is more fun! The lab assignment can be coded as either a recursive or non-recursive version of binary search.

SUMMARY/ REVIEW: Searching algorithms is widely used in programs. Binary searching is the fastest - if the list is sorted.

ASSIGNMENT:
Lab Assignment A19.1, *Store*
Lab Assignment A19.1, Data File, *file50.txt*
Lab Assignment A19.2, *Search*
Lab Assignment A19.3, *CountWords*
Lab Assignment A19.3, Data Files, *test.txt*, *dream.txt*, *Lincoln.txt*
Worksheet A19.1, *James Bond Database*
Worksheet A19.1, Data File, *bond.txt*
Worksheet A19.2, *James Bond Search*