SNS College of Engineering Coimbatore - 641107

Binary search

Introduction

- Very efficient algorithm for searching in sorted array
- A[0…n-1] is the key element.

A[0]….A[m-1] A[m] A[m+1]….A[n-1]

key<A[m] Key Key >A[m]

Conditions:

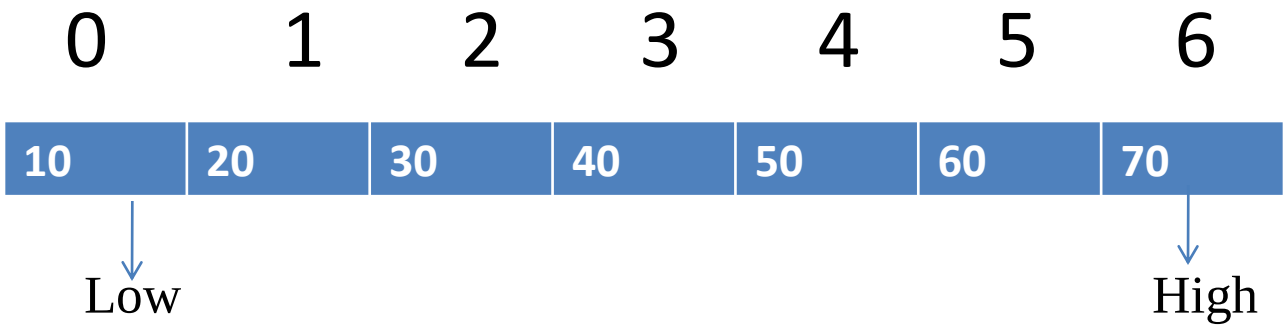
- If key=A[m] then element is present in the list.
- Otherwise if key < a[m] then search left sub tree
- Otherwise if key > a[m] then search right sub tree

Algorithms

Binary search(A[0…n-1],k)
// implements non recursive binary search
//Input: An array A[0…n-1] sorted in ascending order and a search key K
//output: An index of an arrays element that is equal to k or -1 if there is no such element

low← 0;
high ← *n*-1
while (low ≤ high) do
    m ← ⌊(*low*+high)/2⌋
    if *K* = A[*m*]
return *m*
    else if *K* < A[*m*]
high← *m*-1
    else
low ← *m*+1
return -1

Problem



M=(low+high)/2

M=(0+6)/2

M=3

A[m]=key A[3]=60 A[3]=40 & 40<60

Now search in right sub list

Do the same process

Break

- **Assemble the Quotes**
- Preparation: For a group of 30 people, print 5 or 6 quotes or phrases on a paper (i.e. Face that launched a thousand ships; Fools rush in where angels fear to tread; Picture is worth a thousand words; Power corrupts; absolute power corrupts absolutely; etc.) and cut that printed paper so that each word of each phrase is a separate piece of paper. Fold up each of these 30 or so bits of paper and give one to each participant.
- Activity: When you say "go," have everyone simulatneously open their folded paper, then move around the room and find other words related to a possible phrase, from people in the room and try to complete the phrases. When they have feel they created a phrase, they can check in with the facilitator. This involves people to suddenly get energized, both in mind and body.

Analysis of Binary search

- Best case:— $\Theta(1)$
- Worst case: — $\Theta(n \log n)$
- Average case:— $\Theta(n \log n)$