Search algorithms

Vladimir Petkov

What is searching in programming?

What is searching in programming?

We try to find an element in a given sequence,

Expected output is one of:

- position of the element in the sequence
- information that the element cannot be found in the sequence

What to do if we have multiple elements with the given value:

- return any matching element
- return the first matching element
- return the last matching element

Naïve search

- Iterate over all the elements of the sequence.
- Complexity ?
- If we are going to perform a single search and we know nothing about the sequence this is the best we can do

Binary search

- Works on sorted sequence
- Example of a divide and conquer technique

- Maintains an interval where the element we are searching for can be:
    - Initialize this interval with the whole sequence
    - On each step, split the interval in two and only proceed with
    - part that can possibly hold the element
    - Terminates when the interval is smaller than 1

Binary search

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Binary search

- Useful not only on sorted sequences/arrays

- Can be used to search the solution on a monotonic and continuous

- Be careful when working with doubles

Binary search

Find the x for which:

x^3 + 3*sqrt(x) + log(x) = 72

For x > 0 the function is continuous and monotonically increasing

Coding time

Although the basic idea of binary search is comparatively straightforward, the details can be surprisingly tricky …

Donald Knuth

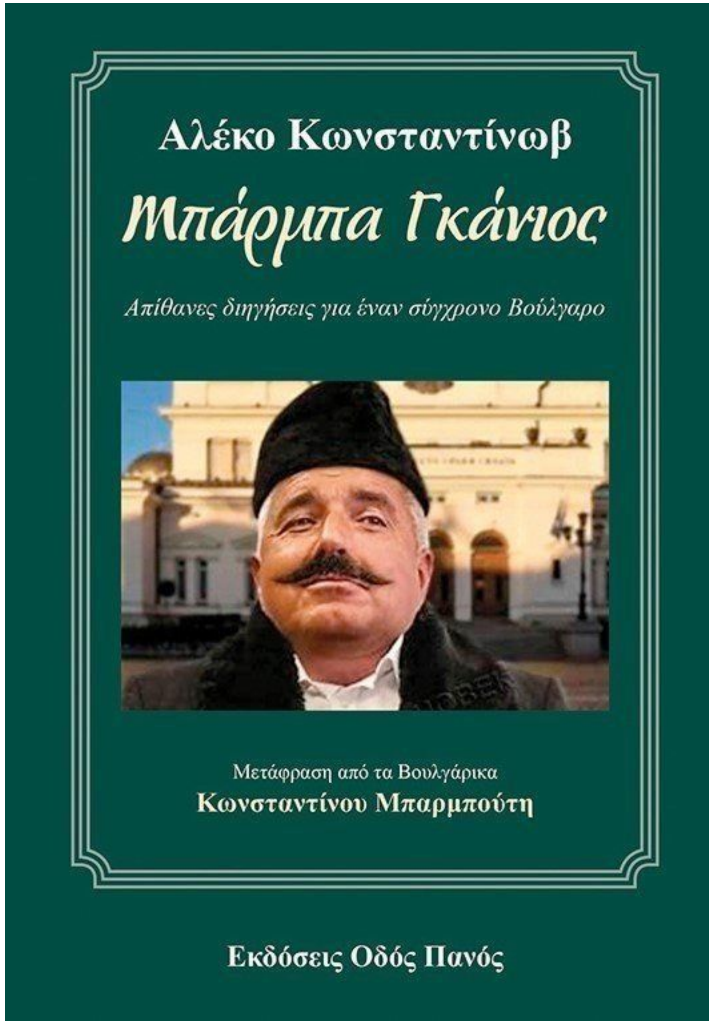
Can we do better?

Can we do better?

You are in a library.

Books are ordered alphabetically.

How would you search for the book “Бай Ганьо”?



Interpolation search

- Interpolation search uses knowledge for the distribution of the elements in the sequence.
- In the previous case we knew the distribution is linear.
- Idea is the same as in binary search
- splitting point is computed differently with the knowledge for the distribution

Interpolation search

- When distribution is close to linear the formula for the splitting index is: **leftIndex + ((target - arr[leftIndex]) * (rightIndex - leftIndex) / (arr[rightIndex] - arr[leftIndex]))**
- Apart from this difference the algorithm of binary search is the same.
- If the distribution is close enough to the one we are interpolating, the search will perform O(log(log(n))) comparisons.

Interpolation search

- What if we are wrong about the distribution?
- O(?)

In a nutshell

- For a single search in unordered sequence naïve search is the best option
- If the sequence is sorted but nothing is known about the distribution of values, binary search is the best option having complexity O(log(N))
- If we have further knowledge for the distribution of the values, we may use interpolation search that has an average case complexity of O(log(log(N)))

All clear?

… or nothing clear?