Binary Search

Feb. 21, 2014, 9:28 a.m. by Rosalind Team **Topics**: Divide-and-conquer

Binary search is the ultimate **divide-and-conquer** algorithm. To find a key k in a large file containing keys $A[1..n]$ in sorted order, we first compare k with $A[n/2]$, and depending on the result we recurse either on the first half of the file, $A[1..n/2]$, or on the second half, $A[n/2+1..n]$. The recurrence now is $T(n) = T(n/2) + O(1)$. Plugging into the **master theorem** (with $a = 1, b = 2, d = 0$) we get the familiar solution: a running time of just $O(\log n)$.

Source: Algorithms by Dasgupta, Papadimitriou, Vazirani. McGraw-Hill. 2006.

Problem

The problem is to find a given set of keys in a given array.

Given: Two positive integers $n \leq 10^5$ and $m \leq 10^5$, a sorted **array** $A[1..n]$ of integers from -10^5 to 10^5 and a list of m integers $-10^5 \leq k_1, k_2, \ldots, k_m \leq 10^5$.

Return: For each k_i, output an index $1 \leq j \leq n$ s.t. $A[j] = k_i$ or "-1" if there is no such index.

Sample Dataset

```
5
6
10 20 30 40 50
40 10 35 15 40 20
```

Sample Output

```
4 1 -1 -1 4 2
```