

The algorithms portion of the course will require you to evaluate mathematical expressions. Recursive algorithms have a natural synergy with recurrence relations and hence for this course, you will be expected to solve simple recurrence relations.

1. $T(n) = 2T(n-1) + cn$
2. $T(n) = 2T\left(\frac{n}{2}\right) + cn$
3. $T(n) = 2T\left(\frac{n}{4}\right) + cn$

Recurrence relations are a fascinating and area of discrete mathematics. Feel free to explore more advanced equations and solving techniques!¹

Here are several problems that are easy to solve in $O(n)$ time, essentially by brute force. Your task is to design algorithms for these problems that are significantly faster using binary search related ideas.

1. Suppose we are given an array $A[1..n]$ of n *distinct integers*, which could be positive, negative, or zero, sorted in increasing order so that $A[1] < A[2] < \dots < A[n]$.
 - (a) Describe a fast algorithm that either computes an index i such that $A[i] = i$ or correctly reports that no such index exists.
 - (b) Formulate a recurrence relation that describes your algorithm.
 - (c) Suppose we know in advance that $A[1] > 0$. Describe an even faster algorithm that either computes an index i such that $A[i] = i$ or correctly reports that no such index exists. [Hint: This is **really** easy.]
2. Suppose we wanted to count the number of times some integer value x occurs in A . Describe an algorithm (as fast as possible) which returns the number of elements containing value x .
3. Suppose we are given an array $A[1..n]$ such that $A[1] \geq A[2]$ and $A[n-1] \leq A[n]$. We say that an element $A[x]$ is a **local minimum** if both $A[x-1] \geq A[x]$ and $A[x] \leq A[x+1]$. For example, there are exactly six local minima in the following array:

9	7	7	2	1	3	7	5	4	7	3	3	4	8	6	9
	▲			▲				▲		▲	▲			▲	

Describe and analyze a fast algorithm that returns the index of one local minimum. For example, given the array above, your algorithm could return the integer 9, because $A[9]$ is a local minimum. [Hint: With the given boundary conditions, any array **must** contain at least one local minimum. Why?]

¹http://discrete.openmathbooks.org/dmoi2/sec_recurrence.html

4. Suppose you are given two sorted arrays $A[1..n]$ and $B[1..n]$ containing distinct integers. Describe a fast algorithm to find the median (meaning the n th smallest element) of the union $A \cup B$. For example, given the input

$$A[1..8] = [0, 1, 6, 9, 12, 13, 18, 20] \quad B[1..8] = [2, 4, 5, 8, 17, 19, 21, 23]$$

your algorithm should return the integer 9. *[Hint: What can you learn by comparing one element of A with one element of B ?]*

To think about later:

4. Now suppose you are given two sorted arrays $A[1..m]$ and $B[1..n]$ and an integer k . Describe a fast algorithm to find the k th smallest element in the union $A \cup B$. For example, given the input

$$A[1..8] = [0, 1, 6, 9, 12, 13, 18, 20] \quad B[1..5] = [2, 5, 7, 17, 19] \quad k = 6$$

your algorithm should return the integer 7.

5. Suppose you have an algorithm that given as input a directed graph $G = (V, E)$, nodes $s, t \in V$, and an integer k , outputs whether the *number* of distinct shortest paths from s to t is at least k . Describe an algorithm that counts the number of distinct shortest s - t paths in G . Does your algorithm run in polynomial time?