

Due in class: Oct 3.

- (1) Given a sorted array $A[1 \dots n]$ of n distinct integers (positive or negative), give an algorithm that finds an index i such that $A[i] = i$, if such an index exists. Your algorithm should take $O(\log n)$ time in the worst case.
- (2) Use the master theorem to solve the following recurrence. You may assume that n is a power of 2.

$$\begin{aligned}T(1) &= 1 \\T(n) &= 4T\left(\frac{n}{2}\right) + n^c\end{aligned}$$

Find a function $g(n)$ such that $T(n) = \Theta(g(n))$. (The function g may involve several cases depending on what c is. The cases should be clearly and easily specified.)

- (3) Solve the following recurrence by iteration.

$$T(1) = 1$$

$$T(n) = 2T(n/2) + n \log n \quad \text{if } n > 1$$

- (4) Use the iteration method to solve the recurrence $T(n) = T(n - a) + T(a) + n$ where $a \geq 1$ is a positive integer. You may assume that n is a multiple of a .