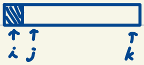
題目: 給定一 A[1,...,n] 為 n 個 element 的 array,
問是否存在三數 a, b, c in A
使得 a+b+c = C, $k \in \mathbb{Z}$

idea:

bruteforce: 用 $1 \le i \le j \le k \le n$ 可知方法數為 $C^n_3 = O(n^3)$
∴ 窮舉所有方法數看是否 A[i]+A[j]+A[k] = C 即可得

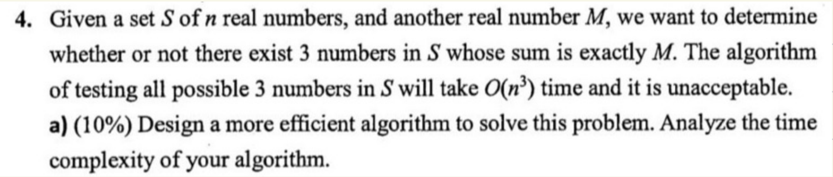
Reduce to two-sum 共做 n 輪, 等同每一輪做 C-A[i] 之 two sum problem
two sum 為 $O(n \lg n)$ 可解
∴ 為 $O(n^2 \lg n)$

$O(n^2)$ 方法: ①. 先將 A[1,...,n] 排序
②. 共做 n 輪, 設為第 i 輪



在剩下 A[i+1,...,n] 之 subarray 用兩指標: j, k
若 A[j]+A[k] == C-A[i] return True
若 A[j]+A[k] < C-A[i] ⇒ i 指標向前
若 A[j]+A[k] > C-A[i] ⇒ j 指標向後
直至 j>k 進到下一輪

n 輪做完後, return False
會為 $O(n^2)$ #.



4. Given a set *S* of *n* real numbers, and another real number *M*, we want to determine whether or not there exist 3 numbers in *S* whose sum is exactly *M*. The algorithm of testing all possible 3 numbers in *S* will take $O(n^3)$ time and it is unacceptable.

a) (10%) Design a more efficient algorithm to solve this problem. Analyze the time complexity of your algorithm.