Question 3 by Dan Nguyen (z5206032)

An array, A, of size n has distinct positive integers smaller than m i.e. $0 < A[i] < m$.

Suppose i and j are valid indices of some array. The smallest absolute difference between any two integers of the array, $|A[i] - A[j]|$, is the *separation* of that array.

Merge-sort A for a time complexity of $O(nlog(n))$ so that $A[0] < A[1] < ... < A[n]$.

Let there be an array, D, with size $n-1$, which stores adjacent *separations* of **sorted** A i.e.:

$$D[i] = |A[i] - A[i+1]| \tag{1}$$

Pre-processing A to fill D will have an expected time complexity of $O(n-1) \leq O(n)$.

Let there be a *separation* threshold, $s \in \mathbb{Z}[1, m]$, with an initial value of 1 which acts as a minimum bound for searching for adjacent *separations* of A.

Let there be a sum, S, which summates the differences encountered in D and has an initial value of 0.

Let there be a set, L, with capacity n, which is a subset of A with the largest possible *separation*. Let there be an L size counter, $l \in \mathbb{Z}[2, n]$.

There is a given integer, $k \in \mathbb{Z}[2, n]$, which is the desired length of L.

For each iterator, i, in D, add $D[i]$ to S, and if S is at least s then insert $A[i+1]$ into L, increment l, then reset S. This will have an expected time complexity of $O(n-1) \leq O(n)$.

If L has a size $l > k$, then there is a valid set of k integers which has a *separation* of at least s. s is increased using the bisection method i.e. s is increased by $s + (m - s)/2$ until $l \leq k$.

If L has a size $l < k$, then there is no valid set of k integers which has a *separation* of at least s. s is decreased using the bisection method i.e. s is decreased by $s/2$ until $l \geq k$.

D is iterated over again with the new *separation* threshold until s cannot be increased or decreased anymore which by then L will have exactly k integers with the largest *separation*. This has an expected time complexity of $O(log(m))$.

If L has a size $l = k$, then there is exactly k integers which has a *separation* of at least s. L can be immediately returned as increasing s will only remove elements from L and there is no need to iterate over D again.

Therefore, the final expected time complexity is $O(nlog(m))$ as required.