Question 1 by Dan Nguyen (z5206032)

An array, A, of size n has only positive integers i.e. $A[i] > 0$ for $i \in \mathbb{Z}$. Let there be a pair of indices i and j where $i < j$. These indices are *consistent* if $A[j] - A[i] = j - i$.

Rearrange the *consistency* rule as:

$$A[i] - i = A[j] - j \tag{1}$$

Consider the index, i, which has the range from 1 to n inclusive. This index will be used to iterate over A.

Part A

Let there be a hash table, H, of an appropriate size larger than or equal to n. H is initially empty.

To count *consistent* indices, $A[i] - i$ is looked-up in H for each i in A. If the look-up was successful, then the counter for discovered pairs of *consistent* indices is incremented. Otherwise, the value, $A[i] - i$, is inserted into H. Iteration over A has an expected time-complexity of $O(n)$, and hash table look-ups and insertions have an expected time-complexity of $O(1)$ - giving a final expected time-complexity of $O(n)$.

Part B

Let there be an AVL tree, B, of size n which stores the value, $A[i] - i$, for each i in A. Iteration through A has time-complexity $O(n)$ and an AVL tree insertion has a worst time-complexity of $O(log(n))$. This gives a final worst time-complexity of $O(log(n))$.

After iterating through A, merge-sort B so that the values of B are ordered. Merge-sort has a worst time-complexity of $O(nlog(n))$.

To count *consistent* indices, a binary search in B is done to find a value equal to $A[i] - i$ for each i in A. A successful binary search increments the counter for discovered pairs of *consistent* indices. Otherwise, iteration over A is continued. Iteration over A has a time-complexity of $O(n)$ and a binary search through an AVL tree has a worst time-complexity of $O(log(n))$ - giving a final worst time-complexity of $O(log(n))$.