FIT2004 S2/2016: Assessment week-04: Solution Overview

Task 1-A.

An array of size N+1 is created where each element is initialized to 0. Then, for each invoice number read from the file, the element at array[num] is set to 1 where num is the invoice number. Once all invoice numbers are read, the indices in the array where the value is still 0 are returned.

Task 1-B.

The sum of numbers from 1 to N is $\frac{N(N+1)}{2}$ which is stored in a variable say *total*. Then, for each invoice number *num* read from the file, *num* is subtracted from *total*. The value of the *total* at the end of the program is the missing invoice number.

Task 2.

The main requirement for the question is that the solution would use $O(k)$ space with an $O(N \log k)$ worst time complexity. This can be achieved by using a min heap of size k where key is the number of followers. The heap is initialized with first k users. Note that the root of the heap corresponds to the user with k^{th} smallest number of followers. Then, for each of the remaining $n - k$ users, the user is inserted in the heap and an entry from the heap is popped. This takes $O(\log k)$ for each new user because the heap contains $O(k)$ entries. The total cost for N users is $O(N \log k)$. The heap contains top-k celebrities when the program terminates. The entries in the heap can then be sorted in $O(k \log k)$ and displayed,

Task 3.

The main requirement for the question is that the solution would take $O(\log N)$ time to update median every time a number is read. The main idea is to use two heaps such that the difference in their sizes is never more than one. One heap is a min-heap and the other is a max-heap. The min-heap will store the largest half of the seen numbers and the max-heap will store the smallest half of the seen numbers. Therefore, the root of min-heap and the root of max-heap correspond to the middle elements if all numbers are assumed to be sorted. If the different between the sizes of the two heaps becomes more than 1, an element from the bigger heap is popped and inserted in the smaller heap.