Problem 1 - Heap Insert and Delete

Insert 17, then 8 on the heap below. Then perform delete-max on the original heap.

(Same as Problem 4 on Tutorial 1)

Problem 2 - Heapsort

Perform heapsort on the following array:


Problem 3 - Quickselect

Perform QuickSelect to find the 3rd smallest element in this unsorted array:

\[ A = [8, 17, 10, 1, 6, 20, 2, 9, 7, 13] \]
Problem 4 - Merging Lists

Given \( k \) sorted lists, where the combination of the \( k \) lists has \( n \) elements in total, give an \( O(n \times \log(k)) \) algorithm to combine the \( k \) sorted lists into a single sorted list.

Problem 5 - Linear Time Range Matching

Consider a list of integers \((a_1, a_2, a_3, ..., a_n)\) such that \( a_i < a_j \) if \( i < j \) for \( 1 \leq i \leq j \leq n \). Also, given a string \( S \) of length \( k \), let \( S_i \) be the \( i-th \) suffix of \( S \) starting from the last character (so \( S_1 \) would be the last character of \( S \) and \( S_k \) would be the entire string). Let \( \text{weight}(S_i) = \) the sum of the numerical ascii values of all the characters in \( S_i \).

a) Write pseudocode for a \( \Theta(n + k) \) algorithm that takes as input the list of integers \((a_1, a_2, a_3, ..., a_n)\), the string \( S \), and a value \( \delta \in \mathbb{Z}^+ \) and finds all \( a_i \) that lie within \( \delta \) of some \( \text{weight}(S_j) \) for all \( 1 \leq j \leq k \). The algorithm must use only constant additional space. Assume that reporting each \( a_i \) does not take additional space.

b) Suppose instead of just reporting each \( a_i \), we want to write an algorithm that constructs a list of tuples \((a_i, S_j)\) which are all pairs of \( a_i \) within \( \delta \in \mathbb{Z}^+ \) of \( \text{weight}(S_j) \) for all \( 1 \leq j \leq k \). Relaxing the constant space constraint, does such an algorithm run in \( \Theta(n + k) \) time in the worst case? If not, derive the worst case run time of the algorithm.