Exercise 1 Typing some traces

These exercises are simply meant to give you some first-hand experience with the algorithms we have seen today on heaps. The next exercises are more fun and creative.

1. Using Figure 6.2 as a model, illustrate the operation of MAX-HEAPIFY$(A, 3)$ on the array $A = \langle 27, 17, 3, 16, 10, 1, 5, 7, 12, 4, 8, 9, 0 \rangle$. (CLRS 6.2-1)
2. Using Figure 6.3 as a model, illustrate the operation of BUILD-MAX-HEAP on the array $A = \langle 5, 3, 17, 10, 84, 19, 6, 22, 9 \rangle$. (CLRS 6.3-1)
3. Using Figure 6.4 as a model, illustrate the operation of HEAPSORT on the array $A = \langle 5, 13, 2, 25, 7, 17, 20, 8, 4 \rangle$. (CLRS 6.4-1)
4. Suppose that the objects in a max-priority queue are just keys (ie. no need to do $A[i].key$). Illustrate the operation of MAX-HEAP-INSERT$(A, 10)$ on the heap $A = \langle 15, 13, 9, 5, 12, 8, 7, 4, 0, 6, 2, 1 \rangle$. (NB: just use the INSERT operation presented in the slides)

Exercise 2 Exam question

This is a sub-question from the 2024 exam set.

'Game of Thrones' is a pretty great show and you are a huge fan. It only has two problems: the last 2 (or 4, to be honest) seasons and the fact that not *every* episode contains dragons. You really love dragons. Therefore, you decide to watch every episode and note down how many minutes of dragon action each episode has. After each episode you want to add the episode to some data structure, that will you let easily select and rewatch the episode(s) with the most amount of dragon.

1. What data structure might fit your need — a stack, a queue, a linked list or a priority queue? Remember that you don't want to sort all the episodes each time you add a new one or want to get the one with the most dragon. Explain your answer.

Exercise 3 Loop invariant for HEAPSORT

Argue the correctness of HEAPSORT using the following loop invariant (CLRS 6.4-2):

> At the start of each iteration of the **for** loop of lines 2-5, the subarray $A[1:i]$ is a max-heap containing the i smallest elements of $A[1:n]$, and the subarray $A[i+1:n]$ contains the $n-i$ largest elements of $A[1:n]$, sorted.

Exercise 4 Priority Queues

These questions concern priority queues.

1. The largest element in a heap must appear in position 1, and the second largest must be in position 2 or position 3. Give the list of positions in a heap of size 31 where the kth largest *can* appear and *cannot* appear for $k = 2, 3, 4$ (assuming the values are distinct).
2. Explain how to use a priority queue to implement the stack and queue datastructures.
3. The operation MAX-HEAP-DELETE(A, x) deletes the object x from max-heap A. Give a pseudo-code implementation of MAX-HEAP-DELETE for an n-element max-heap that runs in $O(\log n)$ time. (CLRS 6.5-10)