

## Discussion 3

---

1. Let's consider a long, quiet country road with houses scattered very sparsely along it. We can picture the road as a long line segment, with an eastern endpoint and a western endpoint. Further, let's suppose that, despite the bucolic setting, the residents of all these houses are avid cell phone users. You want to place cell phone base stations at certain points along the road, so that every house is within four miles of one of the base stations. Give an efficient algorithm that achieves this goal and uses as few base stations as possible. Prove that your algorithm correctly minimizes the number of base stations.
2. Your friend is working as a camp counselor, and he is in charge of organizing activities for a set of campers. One of his plans is the following mini-triathlon exercise: each contestant must swim 20 laps of a pool, then bike 10 miles, then run 3 miles. The plan is to send the contestants out in a staggered fashion, via the following rule: the contestants must use the pool one at a time. In other words, first one contestant swims the 20 laps, gets out, and starts biking. As soon as this first person is out of the pool, a second contestant begins swimming the 20 laps; as soon as he or she is out and starts biking, a third contestant begins swimming, and so on. Each contestant has a projected *swimming time*, a projected *biking time*, and a projected *running time*. Your friend wants to decide on a *schedule* for the triathlon: an order in which to sequence the starts of the contestants. Let's say that the *completion time* of a schedule is the earliest time at which all contestants will be finished with all three legs of the triathlon, assuming the time projections are accurate. What is the best order for sending people out, if one wants the whole competition to be over as soon as possible? More precisely, give an efficient algorithm that produces a schedule whose completion time is as small as possible. Prove that your algorithm achieves this.
3. The values 1, 2, 3, . . . , 63 are all inserted (in any order) into an initially empty min-heap. What is the smallest number that could be a leaf node?
4. Given an unsorted array of size  $n$ . Devise a heap-based algorithm that finds the  $k$ -th largest element in the array. What is its runtime complexity?
5. Suppose you have two min-heaps, A and B, with a total of  $n$  elements between them. You want to discover if A and B have a key in common. Give a solution to this problem that takes time  $O(n \log n)$  and explain why it is correct. Give a brief explanation for why your algorithm has the required running time. For this problem, do not use the fact that heaps are implemented as arrays; treat them as abstract data types.