

# Interview Questions: Priority Queues (ungraded)

3/3 points (100%)

Practice Quiz, 3 questions



1 / 1  
points

1.

**Dynamic median.** Design a data type that supports insert in logarithmic time, find-the-median in constant time, and remove-the-median in logarithmic time.

*Note: these interview questions are ungraded and purely for your own enrichment. To get a hint, submit a solution.*

binary tree

Your answer cannot be more than 10000 characters.

**Thank you for your response.**

*Hint:* maintain *two* binary heaps, one that is max-oriented and one that is min-oriented.



1 / 1  
points

2.

**Randomized priority queue.** Describe how to add the methods `sample()` and `delRandom()` to our binary heap implementation. The two methods return a key that is chosen uniformly at random among the remaining keys, with the latter

method also removing that key. The `sample()` method should take constant time; the `delRandom()` method should take logarithmic time. Do not worry about preserving the underlying array.

## Interview Questions: Priority Queues (ungraded)

Practice Quiz, 3 questions

3/3 points (100%)

priority queue

Your answer cannot be more than 10000 characters.

Thank you for your response.



1 / 1  
points

3.

**Taxicab numbers.** A *taxicab* number is an integer that can be expressed as the sum of two cubes of positive integers in two different ways:  $a^3 + b^3 = c^3 + d^3$ . For example, 1729 is the smallest taxicab number:  $9^3 + 10^3 = 1^3 + 12^3$ . Design an algorithm to find all taxicab numbers less than  $n$ .

- Version 1: Use time proportional to  $n^2 \log n$  and space proportional to  $n^2$ .
- Version 2: Use time proportional to  $n^2 \log n$  and space proportional to  $n$ .

# Interview Questions: Priority Queues (ungraded)

Practice Quiz, 3 questions

3/3 points (100%)

no idea

▲ Your answer cannot be more than 10000 characters.

**Thank you for your response.**

*Hints:*

- Version 1: Form the sums  $a^3 + b^3$  and sort.
- Version 2: Use a min-oriented priority queue with  $n$  items.

