Exercise 1

Priority queues with DeleteRandom. Add a method `deleteRandom()` to the `MaxPQ` class that deletes and returns a random key. Your implementation should use logarithmic time in the worst case and constant space. (Hint: Exchange the item to be deleted with the item at the end of the heap, delete it, and then restore the heap invariant by swimming or sinking as necessary.)

Exercise 2

Dynamic-median finding. Recall that the median of a set of keys is the "middle" key when the keys are sorted. If there is an even number of keys, then there are two middle keys; in this case, we define the median to be the smaller of the two middle keys. In this exercise, you will design a data type that supports the following API:

- `insert(key)`: insert a new `key` into the data structure.
- `median()`: return the median key.
- `removeMedian()`: remove and return the median key.

Your implementation should support `insert()` in logarithmic time, `median()` in constant time, and `removeMedian()` in logarithmic time.

Hint: Keep the median key in `v`; use a max-oriented heap for keys less than the key of `v`; use a min-oriented heap for keys greater than the key of `v`. To insert, add the new key into the appropriate heap, replace `v` with the key extracted from that heap.

Exercise 3

Computational number theory. Write a program that prints out all integers of the form $a^3 + b^3$ where a and b are integers between 0 and n in sorted order, without using excessive space.

The following algorithm achieves this in time proportional to $n^2 \log n$ and space proportional to n.

- Create a priority queue that can hold n items, where each item is a tuple of the form $(i^3 + j^3, i, j)$.
- Initialize the priority queue by inserting the n items $(0^3 + 0^3, 0, 0)$, $(1^3 + 1^3, 1, 1)$, $(2^3 + 2^3, 2, 2)$, …, $(n^3 + n^3, n, n)$.
- While the priority queue is not empty, do the following:
  - Remove the smallest item $(i^3 + j^3, i, j)$ from the priority queue and print it.
  - If $j < n$, insert the item $(i^3 + (j+1)^3, i, j+1)$ into the priority queue.
    (Note: The priority queue will never contain more than n items, since for each i there is at most one item of the form $(i^3 + j^3, i, j)$ in the priority queue at any time.)

As a second part of this exercise, modify your program to find all distinct pairs of integers (a, b) and (c, d) such that $a^3 + b^3 = c^3 + d^3$. For example, $1729 = 1^3 + 12^3 = 9^3 + 10^3$ is one such number (the *Hardy–Ramanujan* number).