Algorithm 1 init()

```
N = 0
S = new Set()
```

Algorithm 2 get()

```
return S.getRandom(); // returns a random element of S
```

Suppose S is a set of 1000 queries and N contains the count of the number of queries seen prior to each call to add()

Proof of Correctness:

For the first 1000 elements, they are inserted and returned at random uniformly.

Suppose there are N elements in S. We want to show that get() returns any query previously passed to add() with uniform probability. Let's consider the kth query Q where $k \leq N$. The probability that get() will return the kth query is the following:

$$\begin{aligned} P(\text{Q chosen from N queries}) &= P(\text{Q inserted} \wedge \text{Q not removed} \wedge \text{Q chosen}) \\ &= P(\text{Q was inserted})P(\text{Q not removed})P(\text{Q chosen}) \\ &= \frac{1000}{k+1} \times (\frac{k+1}{k+2}\frac{k+2}{k+3}...\frac{N-1}{N}) \times \frac{1}{1000} \\ &= \frac{1}{N} \end{aligned}$$

Since k and N were chosen arbitrarily, this proof holds for all $k \leq N$.

Algorithm 3 set(Q)

```
if N < 1000:
    S.insert(Q) // insert Q into S
else if random() < 1000/(N+1): // random() is a function that returns a random number [0, 1)
    S.popRandom() // get rid of a random element from S
    S.insert(Q) // insert Q into S
N = N+1 // increment N
```