CS 473: Algorithms, Fall 2016
HW 6 (due Wednesday, March 14th at 8pm)

This homework contains three problems. **Read the instructions for submitting homework on the course webpage**.

Collaboration Policy: For this home work, each student can work in a group with up to three members. Only one solution for each group needs to be submitted. Follow the submission instructions carefully.

1. **Reservoir sampling** is a method for choosing an item uniformly at random from an arbitrarily long stream of data whose length is not known apriori.

```
UniformSample:
s ← null
m ← 0
While (stream is not done)
   m ← m + 1
   a_m is current item
   Toss a biased coin that is heads with probability 1/m
   If (coin turns up heads)
        s ← a_m

Output s as the sample
```

   (a) **Not to submit but useful to solve:** Prove that the above algorithm outputs a uniformly random sample from the stream.

   (b) To obtain k samples *with* replacement, the procedure for $k = 1$ can be done in parallel with independent randomness. Now we consider obtaining k samples from the stream *without* replacement. The output will be stored in an array of S of size k.

```
Sample-without-Replacement(k):

S[1..k] ← null
m ← 0
While (stream is not done)
   m ← m + 1
   a_m is current item
   If (m ≤ k)
        S[m] ← a_m
   Else
        r ← uniform random number in range [1..m]
        If (r ≤ k)
             S[r] ← a_m

Output S
```

Given a stream $a_1, a_2, \ldots, a_t$ prove that the preceding algorithm generates a uniform sample of size k without replacement from the stream. Assume that $t \geq k$.

2. We briefly discussed in the class how finding median of a stream of numbers is not easy without using too much space. However, it is possible to find an *approximate* median of a stream. Consider a stream of elements $a_1, a_2, \ldots$ where element a_i arrives at time i. Given a constant $\epsilon > 0$, an ϵ-approximate median of the stream at time t is an element whose rank is between $\lfloor (1/2 - \epsilon)t \rfloor$ and $\lceil (1/2 + \epsilon)t \rceil$ among elements of set $\{a_1, \ldots, a_t\}$. For simplicity let us assume that all elements in the stream are distinct, and are at most n (note that this also implies $t \leq n$).

   (a) Let S be a set of c elements sampled uniformly at random from set $\{a_1, \ldots, a_t\}$, and $0 \leq \delta \leq 1$. Let X be a random variable that captures the number of elements in S with rank strictly less than $\lfloor \delta t \rfloor$. Show that $\delta c - \frac{2c}{t} \leq E[X] \leq \delta c$ in case of both *sampling with replacement* and *sampling without replacement.*

   (b) Given a constant $k > 0$ we want to find an ϵ-approximate median of the stream with probability at least $(1 - \frac{1}{k})$ at any time t. Design a randomized algorithm to do the same. At any time t, your algorithm should be able to return an ϵ-approximate median of set $\{a_1, \ldots, a_t\}$ with probability $(1 - \frac{1}{k})$. The goal is to do this using as less space as possible.

   [*Hint*: Use part (a) and Q1.]

3. Recall the CountMin sketch algorithm to estimate the frequencies of the items in a stream. Suppose the algorithm uses exactly one hash function that maps elements of the stream to $\{0, \ldots, (m-1)\}$ where $m = 10$. Give an example of an input stream σ, say of length t, such that the probability is very high that for at least one of the items $j \in \sigma$, the estimate of its frequency is much larger than its actual frequency. More precisely, give an example such that (for t large enough) the probability that there is an item j with $f'_j - f_j \geq t/2$ is at least 0.99, where t is the stream length. Here f'_j is the estimated frequency of j from the sketch and f_j is the true frequency. Note that element j has to be part of the stream, and therefore has to appear at least once.

   Recall that, assuming elements of the stream are from set $[1..n]$ the hash function $h : [1..n] \rightarrow [0..(m-1)]$ is chosen uniformly at random from a 2-universal hash family $\mathcal{H}$. That is for any $x, y \in [1..n]$, if $x \neq y$ then $Pr_{h \sim \mathcal{H}}[h(x) = h(y)] = \frac{1}{m}$. Additionally, (if needed) assume that $\mathcal{H}$ is 3-uniform as well.

The remaining problems are for self study. Do *NOT* submit for grading.

- An important and fundamental problem in streaming is the following. Suppose the stream consists of m elements each of which is an integer between 1 and n. Here we assume that n is known but the stream can be arbitrarily long. We would like to estimate the number of *distinct* numbers in the stream. For instance if the stream is $1, 1, 10, 2, 2, 2, 1, 1, 10$ the answer should be 3. Of course we can do this by maintaining n counters but this would require a huge amount of space. Efficient randomized algorithms are known that output a $(1+\epsilon)$-approximate estimate for the number of distinct numbers in the stream by using $O(\log n/\epsilon^2)$ space. Here we describe a simple seed idea for this problem. Let the stream of numbers be $a_1, a_2, \ldots, a_m$. We want to estimate d, the number of distinct numbers in the stream.

To estimate d to within a constant factor[1] consider a balls and bins experiment of throwing d identical balls into n bins. Let Z be the smallest index among the indices of the non-empty bins. Suppose $d \in [2^i, 2^{i+1})$. Prove that $\Pr[Z \in [n/2^{i+2}, n/2^{i-1})] \geq c$ for some fixed constant c. Thus, n/Z gives a constant factor estimate for d with probability at least c.

In order to make this into an algorithm we use a random hash function $h : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ and keep track of $Z = \min\{h(a_1), h(a_2), \ldots, h(a_m)\}$ which is only one number to store. Hashing collapses all copies of the same number into one "ball" and and also mimics the process of throwing a ball uniformly into a bin. Of course $h(a_1), h(a_2), \ldots, h(a_m)$ don't behave independently as in the balls and bins experiment unless we choose h from the set of all hash functions. However, one can show that even if h is chosen from a 2-universal family the analysis goes through. More on this can be found in the following lecture notes `https://courses.engr.illinois.edu/cs598csc/fa2014/Lectures/lecture_2.pdf`.

- Jeff's Spring 16 Homework 4 and 5 available at links below. `https://courses.engr.illinois.edu/cs473/sp2016/hw/hw4.pdf`, `https://courses.engr.illinois.edu/cs473/sp2016/hw/hw5.pdf`
- There is another very useful sketch called the Count sketch. You can read about it, if you are interested. `https://courses.engr.illinois.edu/cs598csc/fa2014/Lectures/lecture_6.pdf`

[1] d' is a constant factor estimate for d if there are fixed constants $c_1, c_2 \geq 1$ such that $d/c_1 \leq d' \leq c_2 d$.