Homework #4

1. (10 points) **Exercise 5.2-4 (page 122): Hat-check problem.**

Ans:

Let the random variable X be the number of men that get their own hat. Then, by the definition of expectation, we have:

$$\mathrm{Ex}(X) = \sum_{k=0}^{n} \quad k \cdot \mathrm{Pr}(X = k)$$

However, it is very difficult to compute the probability Pr$(X=k)$ for $k = 1,2,...,n$, because the event for a man gets his own hat is NOT mutually independent. Even so we can solve the problem by using **linearity of expectation**.

Let X_i be an indicator for whether the i-th man gets his own hat or not. That is, $X_i=1$ means the i-th man gets his own hat, and $Xi=0$ means the i-th man gets the wrong hat. So the number of men that get their own hat is the sum of all these indicators:

$$X = \sum_{i=0}^{n} \quad X_i$$

Then take the expected value of both sides and apply linearity of expectation:

$$\mathrm{Ex}(X) = \mathrm{Ex}(\sum_{i=0}^{n} \quad X_i) = \sum_{i=0}^{n} \quad \mathrm{Ex}(X_i)$$

We know that every single man is as likely to get one hat as another, so the probability for a man to get his own hat is just $1/n$. So, we have:

$$\mathrm{Ex}(X_i) = 1 \cdot \mathrm{Pr}(X_i = 1) + 0 \cdot \mathrm{Pr}(X_i = 0) = \frac{1}{n}$$

Then, $\mathrm{Ex}(X) = \sum_{i=0}^{n} \quad \mathrm{Ex}(X_i) = n \cdot \frac{1}{n} = 1.$

2. (90 points) **Problem 5-2 (page 143-144): Searching an unsorted array. Only: (a), (b), (e), (h) only case k=1, and (i) Assume that the number of indices i such that A[i] = x is 1.**

Ans:

A is an unsorted array consisting of n elements, searching for a value x in A.

(a) Pseudocode of random search:

RANDOM-SEARCH (A, x) :

```
n = A.length
let visited[1..n] be a new bool array
for i = 1 to n
    visited[i] = FALSE
visited_counter = 0
while visited_counter < n
        r  = RANDOM(1,n)        // Choose a random number between 1 and n
        if visited[r]
            continue
        else
            if A[r] == x
                return r
            visited[r] = TRUE
            visited_counter += 1
return 0          // FAIL
```

(b) Let X be the number of indices into A that we pick before we find x. Then "X=i" means we does not find x for the first *(i-1)* random choices of index and hit x at the i-th picking. Since there is exactly one index i such that A[i] = x, the probability of hitting x for each random choice is *1/n*, where n is the length of A. Therefore, the probability of $X=i$ is $\frac{1}{n}(1-\frac{1}{n})^{i-1}$.By the definition of expectation, we have:

$$\mathsf{Ex}(X) = \sum_{i=0}^{\infty} i \cdot \mathsf{Pr}(X = i) = \sum_{i=0}^{\infty} i \cdot \frac{1}{n}(1 - \frac{1}{n})^{i-1}$$

Let $S = \sum_{i=1}^{\infty} i \cdot (1 - \frac{1}{n})^{i-1} = 1 \cdot (1 - \frac{1}{n})^0 + 2 \cdot (1 - \frac{1}{n})^1 + 3 \cdot (1 - \frac{1}{n})^2 + \cdots + (i+1) \cdot (1 - \frac{1}{n})^i + \cdots$

Then, $(1 - \frac{1}{n}) \cdot S = \sum_{i=1}^{\infty} i \cdot (1 - \frac{1}{n})^i = 1 \cdot (1 - \frac{1}{n})^1 + 2 \cdot (1 - \frac{1}{n})^2 + \cdots + i \cdot (1 - \frac{1}{n})^i + \cdots$

So, $S - (1 - \frac{1}{n}) \cdot S = \sum_{i=0}^{\infty} (1 - \frac{1}{n})^i = (1 - \frac{1}{n})^0 + (1 - \frac{1}{n})^1 + \cdots + (1 - \frac{1}{n})^i + \cdots$

That is $\frac{1}{n} \cdot S = \sum_{i=1}^{\infty} (1 - \frac{1}{n})^i = \frac{1 \cdot (1 - (1 - \frac{1}{n})^{\infty})}{1 - (1 - \frac{1}{n})} = n$where we know that $S = n^2$.

Finally, we have:

$$\mathsf{Ex}(X) = \sum_{i=0}^{\infty} i \cdot (1 - \frac{1}{n})^i \cdot \frac{1}{n} = \frac{1}{n} \sum_{i=0}^{\infty} i \cdot (1 - \frac{1}{n})^i = \frac{1}{n} S = n$$

(e) We know that *A[1], A[2], ..., A[i]* will be searched before the algorithm terminates if *A[i] = x*. Because there is exactly one index *i* such that *A[i]=x* and all possible permutations of the input array are equally likely, we know that the probability that *A[i]=x* for *i=1,2,...,n,* is 1/n. Let N be the number of searched elements before we find *x*, then we know that *N=i* if *A[i]=x*.

Assume that the comparison of each search step costs a constant time *c*, the average running time will be:

$$T_{avg}(n) = c \cdot \mathsf{Ex}(N) = c \cdot \sum_{i=1}^{n} i \cdot \mathsf{Pr}(A[i] = x) = c \cdot \frac{1}{n} \frac{n(n+1)}{2} = \frac{c}{2}(n+1)$$

The worst case happens when the x appears at the end of A, and the running time is:

$$T_{worst}(n) = c \cdot N(A[n] = x) = cn$$

(h) Assume that the permutation of one element costs c_1 and comparison for search costs c_2. We only consider case k=1, where there is exactly one index i such that *A[i] = x*. The running time of SCRAMBLE-SEARCH should be the sum of both permuting and searching cost.

The worst case is that x is the last element of the resulting permuted array. According to (e), we know that:

$T_{avg}(n) = c_1 \cdot n + c_2 \cdot \mathrm{Ex}(N) = c_1 n + \frac{c_2}{2}(n + 1),$

$T_{worst}(n) = c_1 \cdot n + c_2 \cdot n$

(i) By the assumption that the index i = 1 for A[i] = x. Because A[1]=x, we would choose DETERMINISTIC-SEARCH so that the algorithm will find x and terminate by picking only one element.