Problem 1 (10 points)

Suppose we roll a fair *k*-sided die with the numbers 1 through *k* on the die’s faces. If X is the number that appears, what is E[X]?

The expectation is the sum of each value multiply its probability, so the equation should be as follows:

Problem 2 (10 points)

A monkey types on a 26-letter keyboard that has lowercase letters only. Each letter is chosen independently and uniformly at random from the alphabet. If the monkey types 1,000,000 letters, what is the expected number of times that sequence “proof” appears?

Problem 3 (25 points)

The following approach is often called *reservoir* sampling. Suppose that we have a sequence of items, passing by one at a time. We want to maintain a sample of one item that has the property that it is uniformly distributed over all the items that we have seen at each step. Moreover, we want to accomplish this without knowing the total number of items in advance or storing all of the items that we see. Consider the following algorithm, which stores just one item in memory at all times. When the first item appears, it is stored in the memory. When the *k*-th item appears, it replaces the item in memory with probability 1/*k*. Explain why this algorithm solves the problem.

For , the probability of the first element was 1/k = 1.

For some , after the nth etimes has been replaced the item in memory with probability 1/n, so the nth item in the memory will be 1/n.

For ,the (n+1)th item has been replaced in memory with probability .

The same item as the previous turn with probability , therefore the item in memory in this case can be any item from 1st to nth item with probability   
So the probability of ith item was in memory is for

Thus the statement was right

Problem 4 (15 points)

Suppose that we roll a standard fair die 100 times. Let X be the sum of the numbers that appear over the 100 rolls. Use Chebyshev's inequality to bound .

Let be the number on the face of the die for roll .

We can compute:

And the variance of will be :

And .

Where

Thus

Therefore, by Chebyshev’s inequality we have:

Problem 5: Exercise 2.3.1 (page 58) (25 points)

Design MapReduce algorithms to take a very large file of integers and produce as output:

1. The largest integer.

map(key = file, value=contents):

for each integer in contents, emit (integer, “1”)

reduce(key = integer, values=uniq\_conts):

sum all the pairs that has the same key

emit all the pairs after the summing

max = pair[0].key

find\_max(all pairs):

for pair in pairs:

if pair.key() > max:

max = pair.key()

1. The average of all the integers.

Map(key = file, value = contents)

For each integer in contents, emit (integer, “1”)

Reduce (key = integer, values = uniq\_counts):

Sum all (key, value) pairs in the out list

Emit result pari (total\_integer\_value, total\_count)

1. The same set of integers, but with each integer appearing only once.

Map(key=file, value = contents):

For each integer in contents, emit (integer, “1”)

Reduce(key=integer, values= uniq\_counts):

Sum all the pairs that has the same key

Emit all the pairs after the summing

1. The same set of integers, but with each integer appearing only once.

Map(key=file, value=contents):

For each integer in contents, emit (integer, “1”)

Reduce (key = integer, values = uniq\_counts):

Sum al the pairs that has the same key

Emit all the pairs after the summing

Problem 6: Exercise 3.1.1 (page 95) (15 points)

Compute the Jaccard similarities of each pair of the following three sets: {1, 2, 3, 4}, {2, 3, 5, 7}, and {2, 4, 6}.

Let , ,

Then