

Assignment 1

Qianlang Chen (u1172983)

CS 5140 Spring 2021

Problem 1

```
# The size of the domain in question.  
n = 3000
```

Part A

```
import random  
  
# Runs the experiment of generating random numbers in the range [0, n)  
# until two numbers generated equal. Returns the value `k` where the  
# k-th number equals to one of the numbers generated before.  
def experiment(n):  
    generated = set()  
    for k in range(1, n + 1):  
        x = random.randint(0, n - 1)  
        if x in generated: return k  
        generated.add(x)  
  
# Run the experiment once and report the k-value.  
k = experiment(n)  
print(f'Running the experiment once took {k} random trials')
```

Running the experiment once took 82 random trials

Part B

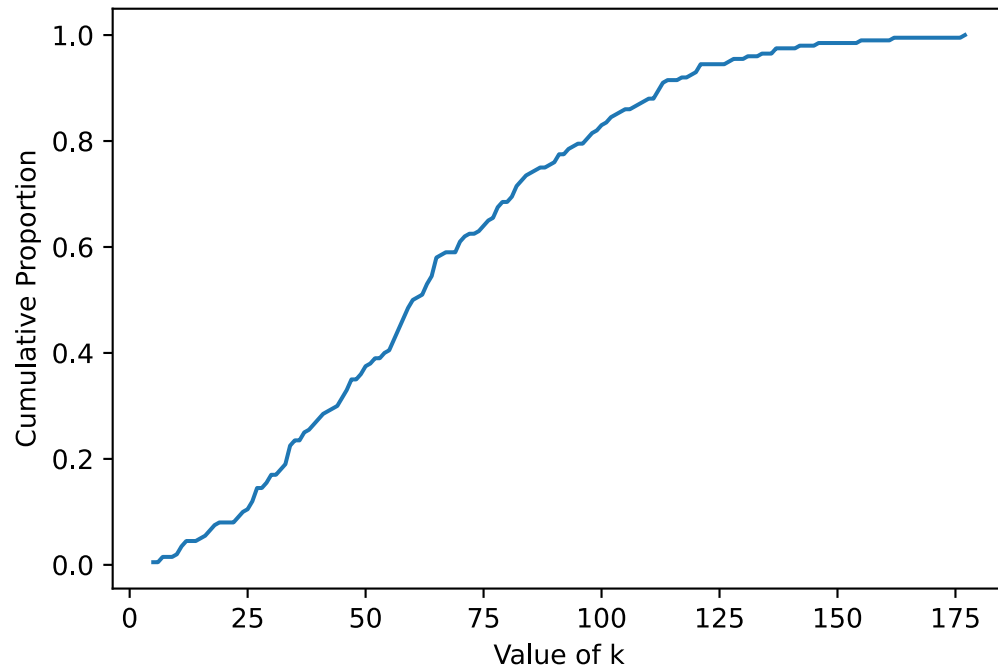
```
import itertools
from matplotlib import pyplot

# The number of experiments to run.
m = 200

# Run the experiment `m` times and record the frequency for each value
# of `k`.
k_freq = {}
for _ in range(m):
    k = experiment(n)
    k_freq[k] = k_freq.get(k, 0) + 1

# Plot the k-frequencies.
X = list(range(min(k_freq.keys()), max(k_freq.keys()) + 1))
Y = list(itertools.accumulate(k_freq.get(x, 0) / m for x in X))
pyplot.xlabel('Value of k')
pyplot.ylabel('Cumulative Proportion')
pyplot.plot(X, Y)
```

[<matplotlib.lines.Line2D at 0x23ee24ac088>]



Part C

```
k_bar = sum(k * x for k, x in k_freq.items()) / m
print(f'Empirical estimate of the expected value of `k`: {k_bar}')
```

Empirical estimate of the expected value of `k`: 65.37

Part D

In the implementation of the experiment:

- I used a hash-set to keep track of the numbers that had appeared (been randomly generated);
- I kept generating random numbers until the first time a number generated exists in that hash-set;
- When the above happened, I immediately terminated the experiment and reported the number of trials it took.

```

import time

# Returns the time in seconds that running an experiment with some
# values of `n` and `m` takes.
def time_experiment(n, m):
    start_time = time.time()
    while time.time() - start_time < .25: pass # warm-up loop
    start_time = time.time()
    for _ in range(m): experiment(n)
    return time.time() - start_time

t = time_experiment(n, m)
print(f'The experiment with n={n} and m={m} took {t:.3f} s')

```

The experiment with $n=3000$ and $m=200$ took 0.015 s

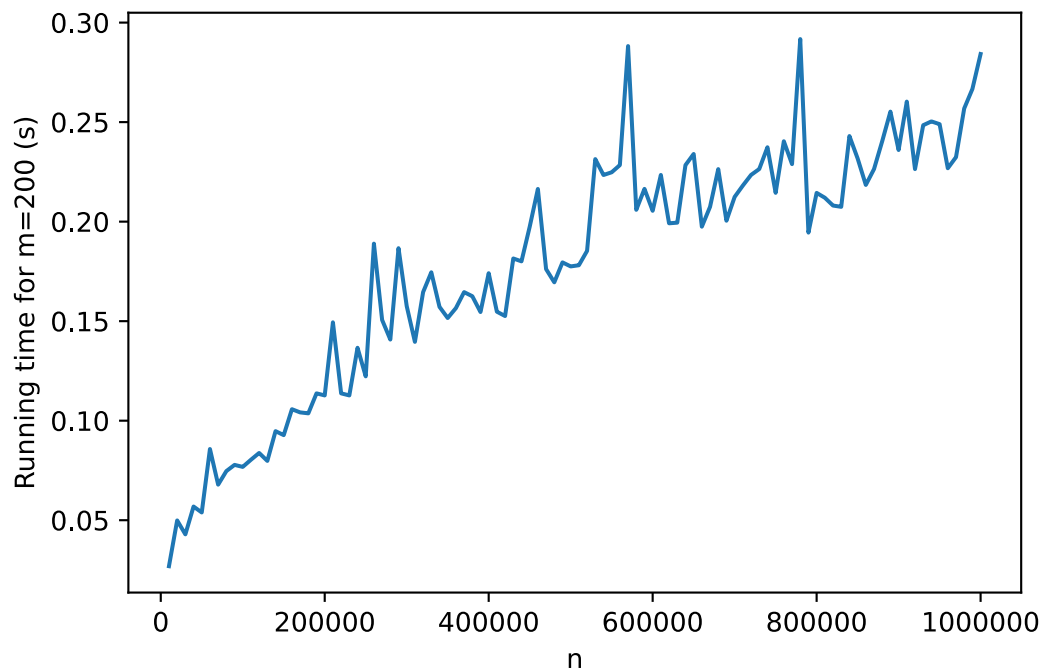
Now, for more values of n and m :

```

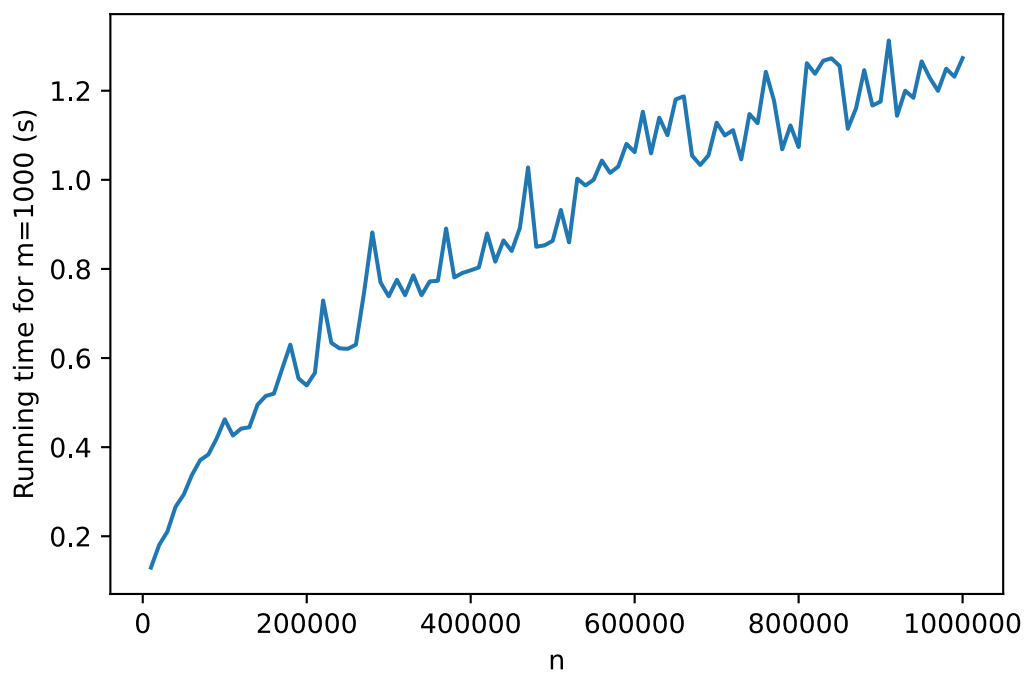
def time_m(m):
    X = list(range(10**4, 10**6 + 1, 10**4))
    Y = list(time_experiment(x, m) for x in X)
    pyplot.ticklabel_format(useOffset=False, style='plain')
    pyplot.xlabel('n')
    pyplot.ylabel(f'Running time for m={m} (s)')
    pyplot.plot(X, Y)

time_m(200)

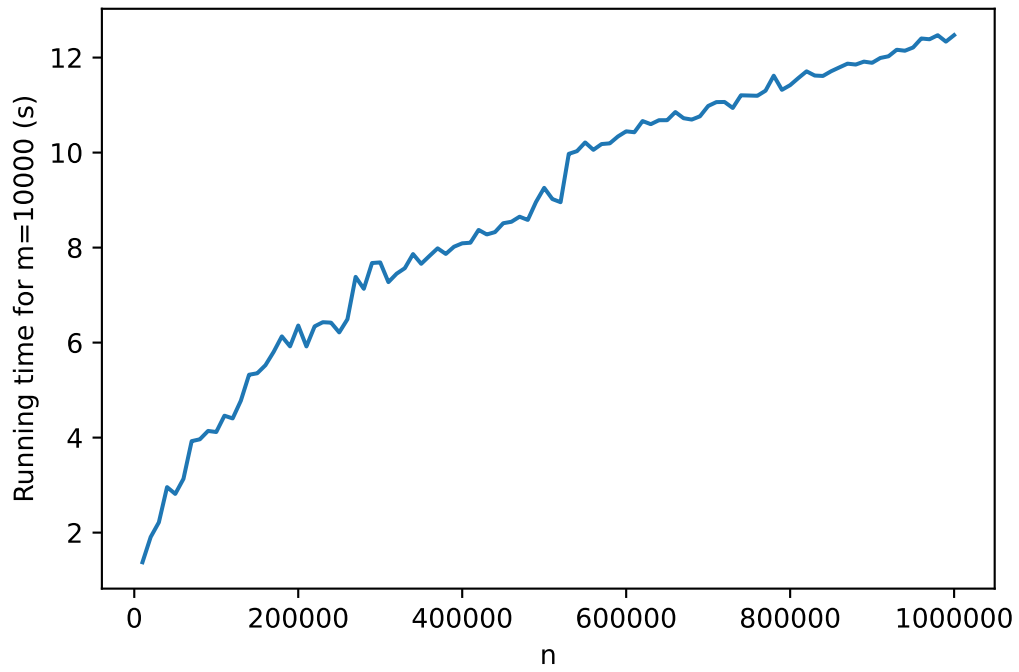
```



```
time_m(10**3)
```



```
time_m(10**4)
```



Problem 2

```
# The size of the domain in question.  
n = 200
```

Part A

```
import random  
import itertools  
  
# Runs the experiment of generating random numbers in the range [0, n)  
# until every possible number has been generated. Returns the value `k`  
# where as of the k-th generation, all possible numbers have been  
# generated.  
def experiment(n):  
    not_generated = {x for x in range(n)}  
    for k in itertools.count(1):  
        x = random.randint(0, n - 1)  
        not_generated.discard(x)  
        if not not_generated: return k  
  
# Run the experiment once and report the k-value.  
k = experiment(n)  
print(f'Running the experiment once took {k} random trials')
```

Running the experiment once took 1126 random trials

Part B

```
from matplotlib import pyplot  
  
# The number of experiments to run.  
m = 300  
  
# Run the experiment `m` times and record the frequency for each value  
# of `k`.
```

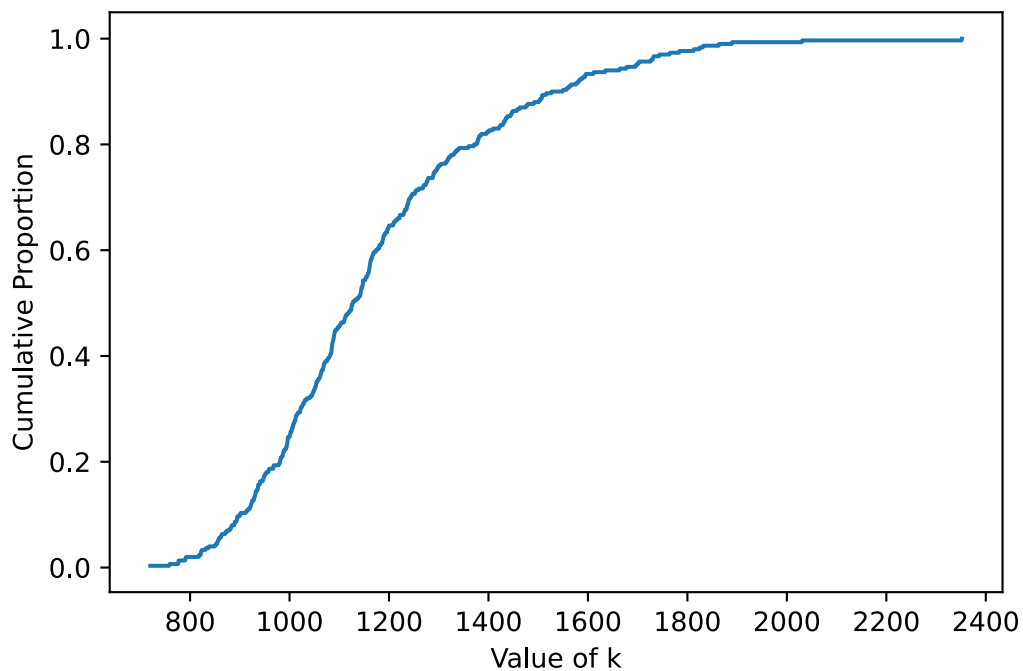
```

k_freq = {}
for _ in range(m):
    k = experiment(n)
    k_freq[k] = k_freq.get(k, 0) + 1

# Plot the k-frequencies.
X = list(range(min(k_freq.keys()), max(k_freq.keys()) + 1))
Y = list(itertools.accumulate(k_freq.get(x, 0) / m for x in X))
pyplot.xlabel('Value of k')
pyplot.ylabel('Cumulative Proportion')
pyplot.plot(X, Y)

```

[<matplotlib.lines.Line2D at 0x23ee2728d88>]



Part C

```
k_bar = sum(k * x for k, x in k_freq.items()) / m
print(f'Empirical estimate of the expected value of `k`: {k_bar}')
```

Empirical estimate of the expected value of `k`: 1176.1733333333334

Part D

In the implementation of the experiment:

- I used a hash-set to keep track of the numbers that had *not yet* appeared (*not* been randomly generated);
- I kept generating random numbers until that hash-set became empty, meaning that all possible numbers had been generated;
- When the above happened, I immediately terminated the experiment and reported the number of trails it took.

```
import time

# Returns the time in seconds that running an experiment with some
# values of `n` and `m` takes.
def time_experiment(n, m):
    start_time = time.time()
    while time.time() - start_time < .25: pass # warm-up loop
    start_time = time.time()
    for _ in range(m): experiment(n)
    return time.time() - start_time

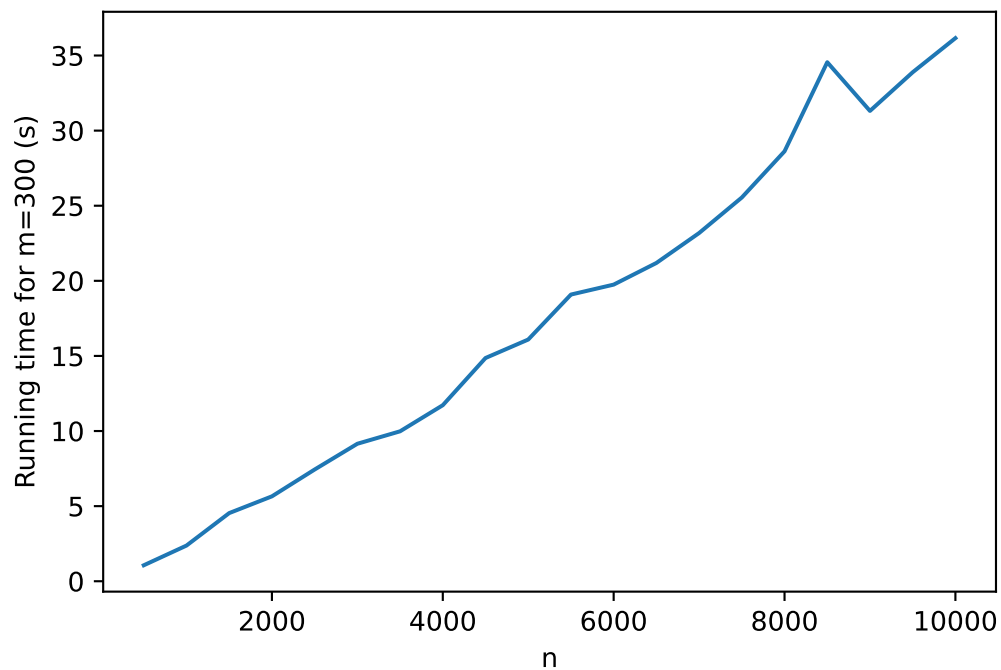
t = time_experiment(n, m)
print(f'The experiment with n={n} and m={m} took {t:.3f} s')
```

The experiment with $n=200$ and $m=300$ took 0.319 s

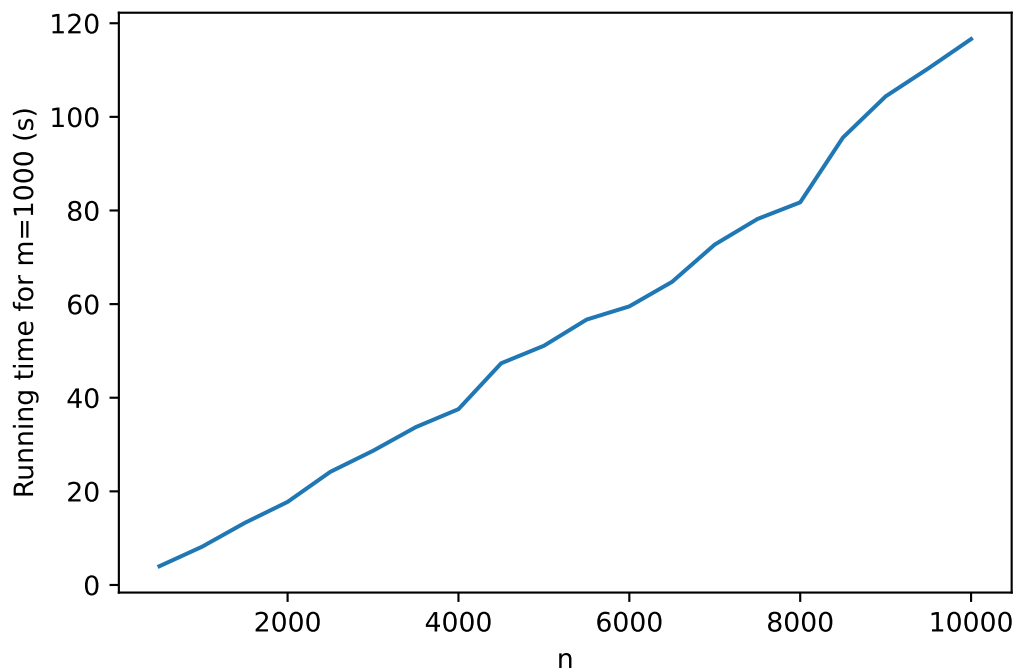
Now, for more values of n and m :

```
def time_m(m):
    X = list(range(500, 10**4 + 1, 500))
    Y = list(time_experiment(x, m) for x in X)
    pyplot.ticklabel_format(useOffset=False, style='plain')
    pyplot.xlabel('n')
    pyplot.ylabel(f'Running time for m={m} (s)')
    pyplot.plot(X, Y)
```

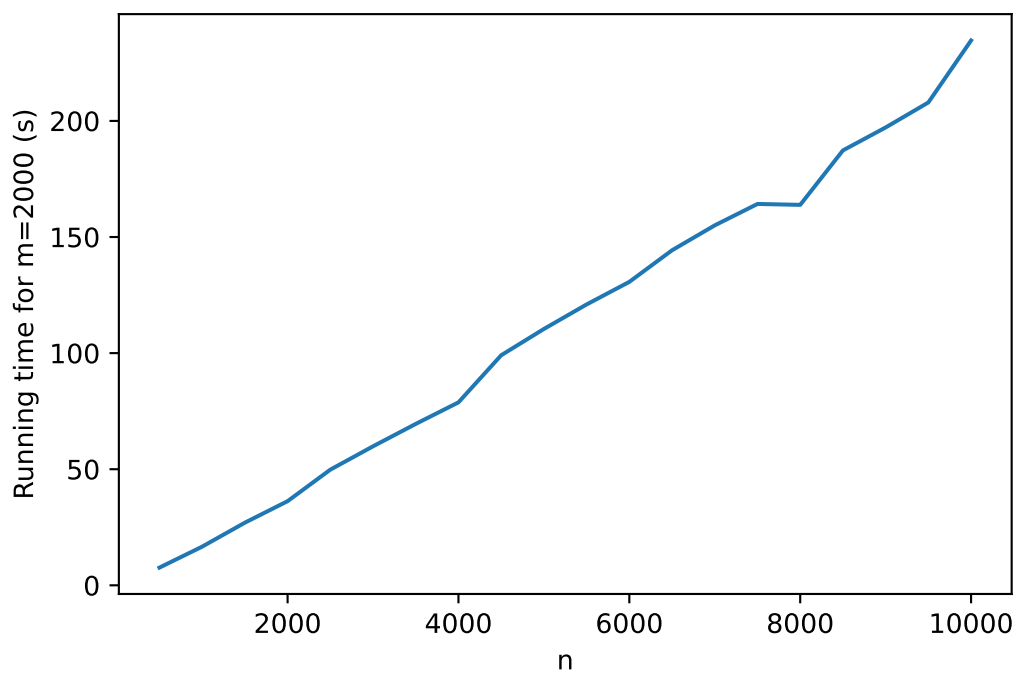
```
time_m(300)
```



```
time_m(1000)
```



```
time_m(2000)
```



Problem 3

Part A

I will use Method 1 from the lecture to estimate the actual expected value of k :

$$\begin{aligned}\Pr(\text{Collision with } k \text{ objects in a domain sized } n) &\approx 1 - \left(1 - \frac{1}{n}\right)^{k^2/2} \\ \frac{1}{2} &\approx 1 - \left(1 - \frac{1}{3000}\right)^{k^2/2} \\ \frac{k^2}{2} &\approx \log\left(\frac{1}{2}, \frac{2999}{3000}\right) \\ k &\approx \sqrt{2 \log\left(\frac{1}{2}, \frac{2999}{3000}\right)} \\ &\approx \boxed{64.48}\end{aligned}$$

Therefore, after generating about 65 random numbers, we can expect the probability that two of the random numbers equal to be more than 50%. This result is reasonably close to the value estimated in *Problem 1-C* (65.37).

Part B

I will use the (only) method from the lecture to estimate the actual expected value of k :

$$\begin{aligned}E[k] &= n \cdot \sum_{i=1}^n \frac{1}{i} \\ &= 200 \cdot \sum_{i=1}^{200} \frac{1}{i} \\ &= 200 \cdot \left(1 + \frac{1}{2} + \cdots + \frac{1}{200}\right) \\ &\approx \boxed{1176}\end{aligned}$$

Therefore, we can expect to see every possible number being generated after generating about 1176 random numbers. This result is reasonably close to the value estimated in *Problem 2-C* (1176).