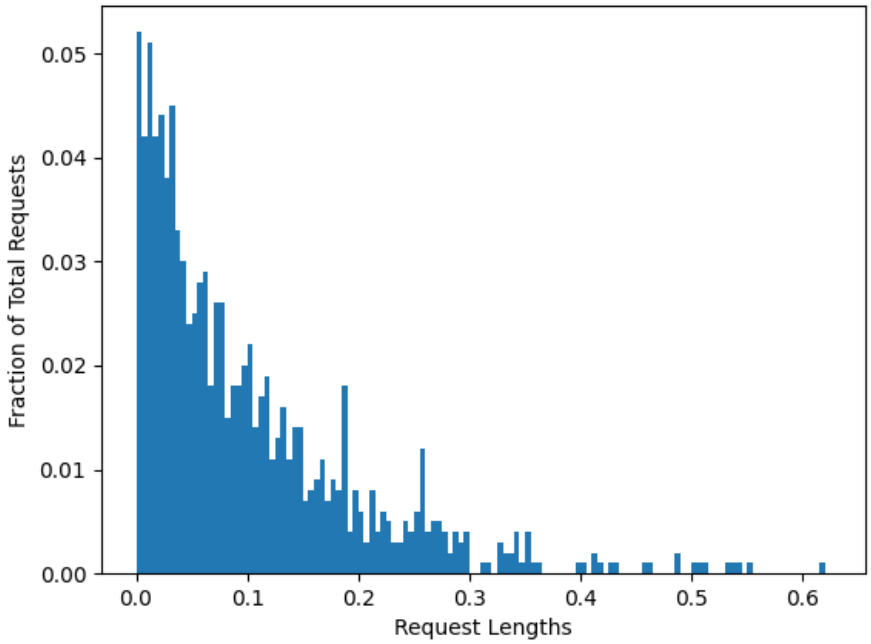
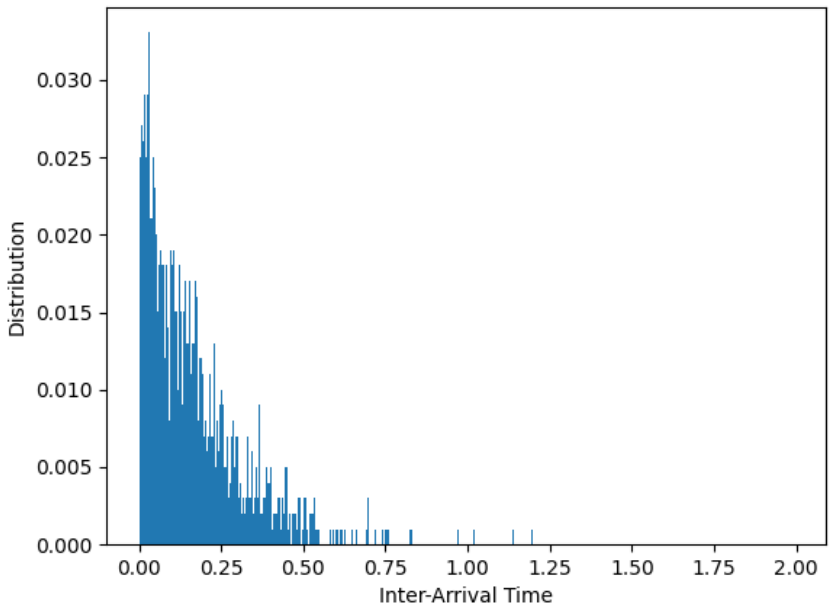
1.

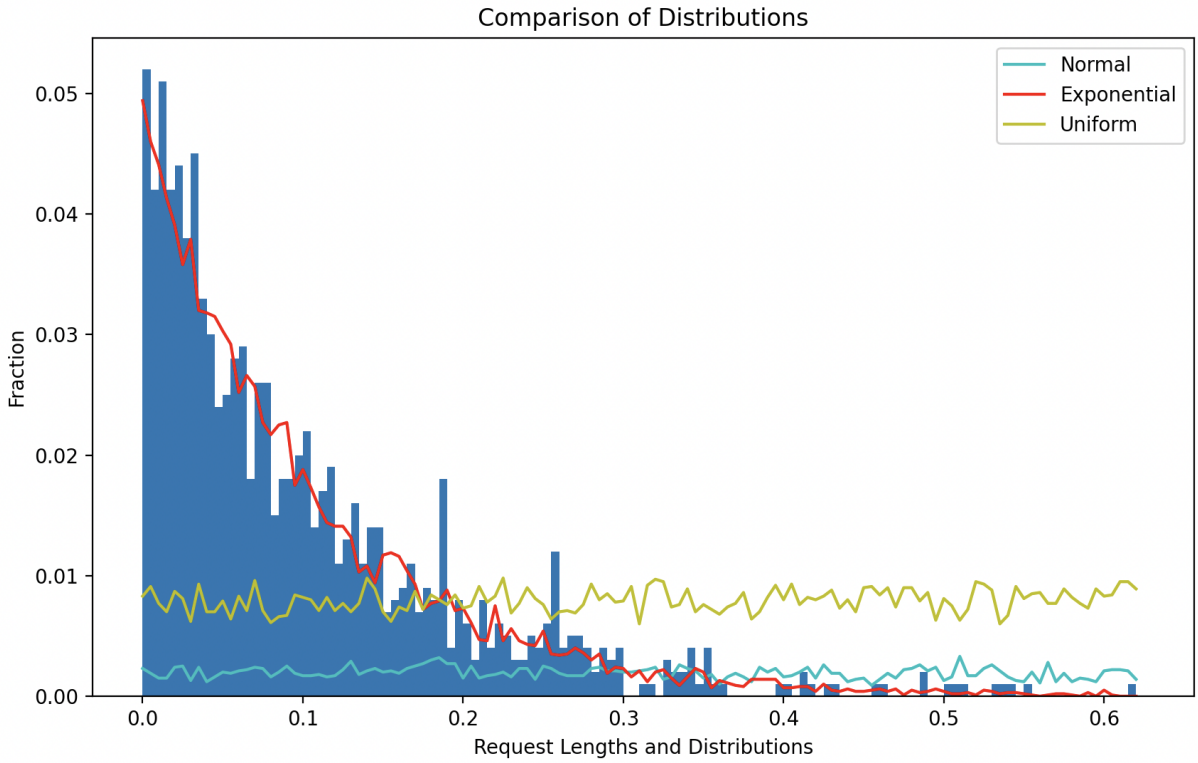
a)


Fraction of Total Requests
0.05
0.04
0.03
0.02
0.01
0.00
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Request Lengths

b)

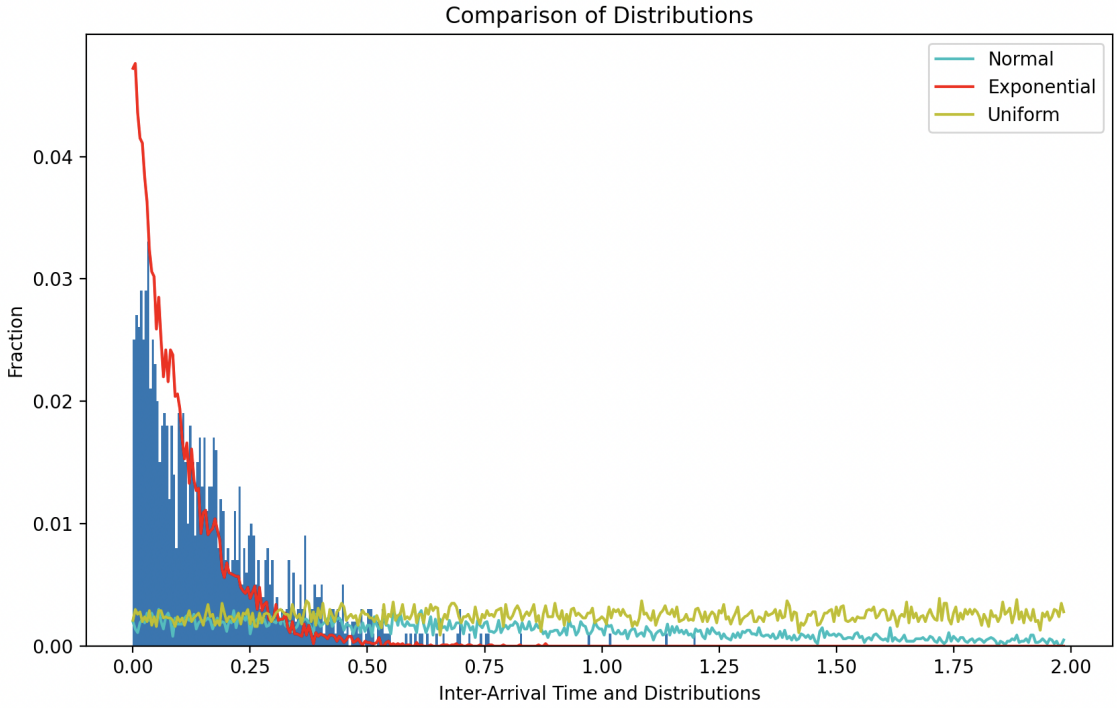

Distribution
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Inter-Arrival Time

c)



Exponential distribution is remarkably close to the distribution of request lengths.

d)



The distribution of inter-arrival time still matches the exponential distribution the most.
-a controls average arrival rate -a of 6 means average of 6 messages come in one second.
-s controls the average request length. -s of 10 indicates average request length is 1/10.

2.
a)
I used regex

```
queue_pattern = re.compile(r'Q:\[([^\]]*)\]')
completion_pattern = re.compile(r'(\d+\.\d{6})$')
```

to capture queue and completion timestamp, later converting them into lists storing queue sizes and completion times.

I use R1 completion time - R0 completion time (and so on) to get time elapsed. Therefore, totally 999 elapsed time.

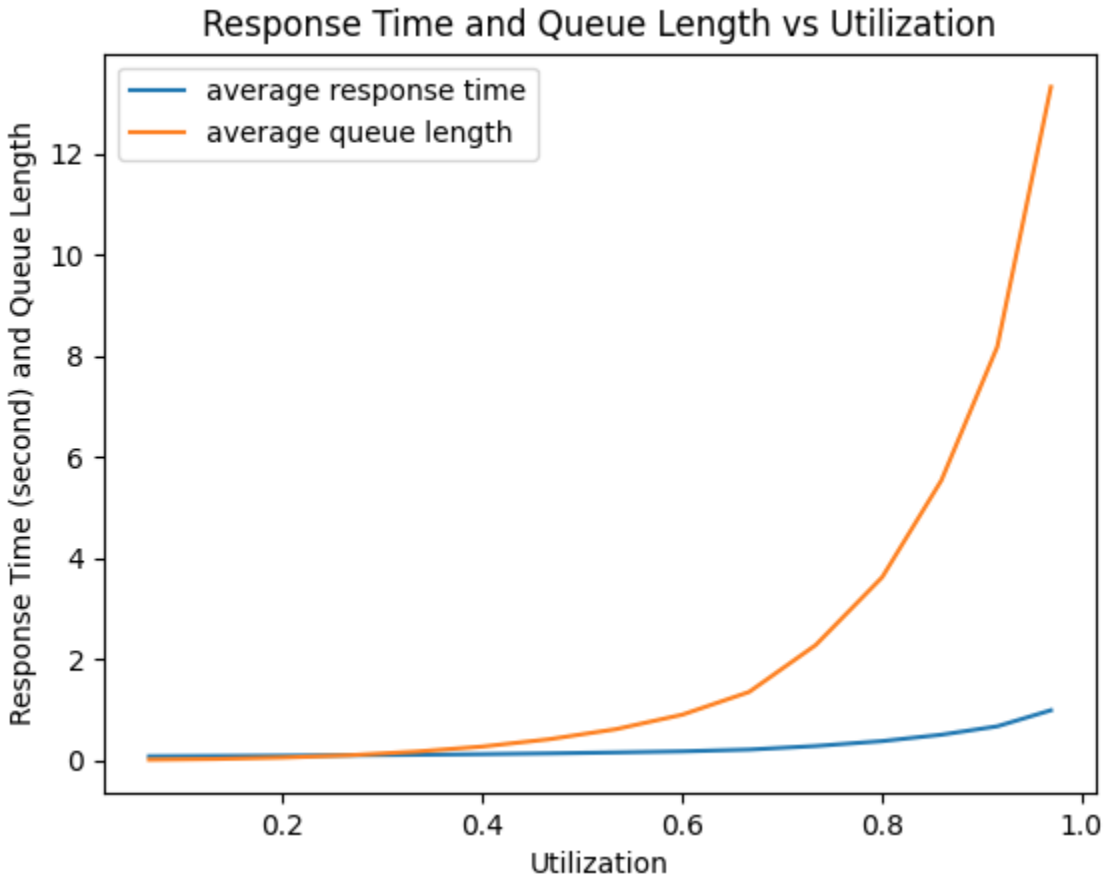
For the queue sizes, since the last snapshot is empty, and the program exits after the last completion time, there are 999 sizes.

I add up all queue size * elapsed time, and divide them by the total runtime (last completion time - first completion time) to get 8.16298.

```
sum = 0
for i in range(999):
    sum += time_elapsed[i] * queue_sizes[i]

timed_avg = sum/(141190.771469 - 141118.214427)
print(timed_avg)
```

b)



Average response time and average queue length both increase as a result of increased utilization. Specifically, when the utilization is higher, average queue length grows significantly faster than response time.

c)
I think when dividing average queue length by average response time, we get the arrival rate which was imputed from -a parameter. This is related to the Little's Law (q = arrival rate * T_q) T_q corresponds to the response time, q means items in the whole system, approximately same as the queue length in a single-server system.