Fundamentals of Computing Systems Homework Assignment #5 - EVAL

1. EVAL Problem 1

   (a) **Understand the Overhead of Using SJN**

   Each time I execute the time -v command, parse its output as "Elapsed (wall clock) time (h:mm:ss or m:ss): 2:36.89" as 2 minutes, 36 seconds, and 89 milliseconds.

   i. **Low Utilization**

	FIFO (sec)	SJN (sec)
1st Trial	156.57999999999998	154.66
2nd Trial	156.76	158.0
3rd Trial	158.76	154.52
4th Trial	153.06	154.89
5th Trial	154.19	154.72
6th Trial	161.82999999999998	153.6
7th Trial	160.47	153.18
8th Trial	156.89	157.21
9th Trial	156.3	153.48
10th Trial	156.88	154.19
Average	157.17	154.84

Table 1: Low Utilization Each Policy average run time

The SJN policy is 2.33 seconds faster than FIFO method.

   ii. **High Utilization**

	FIFO (sec)	SJN (sec)
1st Trial	40.23	41.65
2nd Trial	41.26	40.66
3rd Trial	42.15	40.82
4th Trial	40.23	40.2
5th Trial	39.9	41.1
6th Trial	40.34	40.15
7th Trial	40.58	40.55
8th Trial	50.74	41.38
9th Trial	40.88	40.93
10th Trial	41.12	40.57
Average	41.74	40.80

Table 2: High Utilization Each Policy average run time

The SJN policy is 0.94 seconds faster than FIFO method.

Low utilization:

At low utilization, the average execution time under FIFO is 157.17 seconds, while under SJN it is 154.84 seconds. The SJN policy shows a notable advantage, being 2.33 seconds faster per request on average. Under low load, the extra per-request time spent by the server using FIFO is relatively higher because it processes requests in the order of arrival without considering their length. This often leads to longer requests blocking the queue, increasing the waiting time for subsequent requests. In contrast, SJN minimizes this by always handling the shortest available request first, leading to a more efficient use of server resources and less accumulated waiting time per request, thereby reducing the overall average runtime.

High utilization:

Under high utilization, the difference between FIFO and SJN diminishes. With average execution times of 41.74 seconds for FIFO and 40.80 seconds for SJN, the SJN policy is only 0.94 seconds faster per request on average. When the server is heavily utilized, all requests compete for resources, and the benefit of processing shorter requests first is less impact. The extra per-request time under FIFO is still present, but contention for server resources

and the need to keep all requests moving make it more difficult for SJN to provide significant gains. The reduced gap indicates that, under high load, the server spends a similar amount of extra time on each request regardless of scheduling policy, as resource contention becomes the dominant factor affecting execution time.

In summary, the extra per-request time spent by the server is significantly lower under SJN at low utilization due to more efficient scheduling, while at high utilization, the difference between FIFO and SJN is less impact as server resources are highly contended, reducing the effect of optimal scheduling on individual request times.

(b) **Request Response time as a function of the server utilization**

Just like the previous Eval Problems, I measured each policy Utilizations.

	FIFO		SJN	
-a	Avg Response (sec)	Utilization	Avg Response (sec)	Utilization
22	0.0745	0.54	0.0683	0.54
24	0.0803	0.59	0.0711	0.59
26	0.0870	0.64	0.0750	0.64
28	0.0992	0.69	0.0815	0.69
30	0.1144	0.74	0.0873	0.74
32	0.1358	0.79	0.0953	0.79
34	0.1640	0.83	0.1050	0.84
36	0.2159	0.88	0.1202	0.88
38	0.3274	0.93	0.1495	0.93
40	0.5225	0.98	0.1978	0.98

Table 3: Avg Response VS Utilization

Figure 1: Response time vs Utilization

The plot and accompanying data provide a comprehensive comparison of the FIFO (First-In-First-Out) and SJN (Shortest Job Next) scheduling policies in a server system across varying levels of utilization, ranging from 54% to 98%. Both policies exhibit an increase in average response time as server utilization rises, but SJN consistently outperforms FIFO across all utilization levels. At lower utilization rates (54-64%), the difference in response times between the two policies is relatively small. However, as utilization increases, particularly above 80%, the performance gap becomes more pronounced. FIFO demonstrates a much steeper increase in response times at high utilization than SJN, which maintains more stable response times even under heavy load. On average, SJN proves to be 0.0770 seconds faster than FIFO, representing a significant 42.28% improvement in response time. This superior performance of SJN is attributed to its ability to prioritize shorter jobs, leading to better overall system efficiency and more consistent response times, especially under high load conditions. The results clearly illustrate that SJN scales better with increasing server load. This makes it a

more effective choice for managing varying job sizes and maintaining system performance under pressure.

(c) **CDF plots of -a 40 FIFO and SJN**

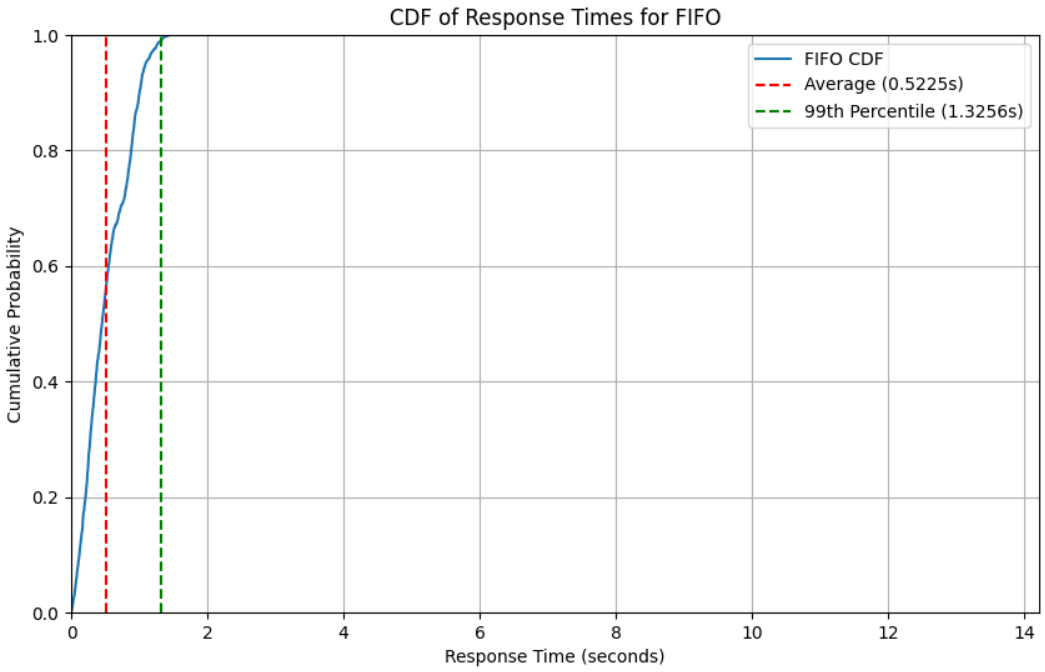

Figure 2: CDF of FIFO

FIFO - Average: 0.5225s, 99th Percentile: 1.3256s

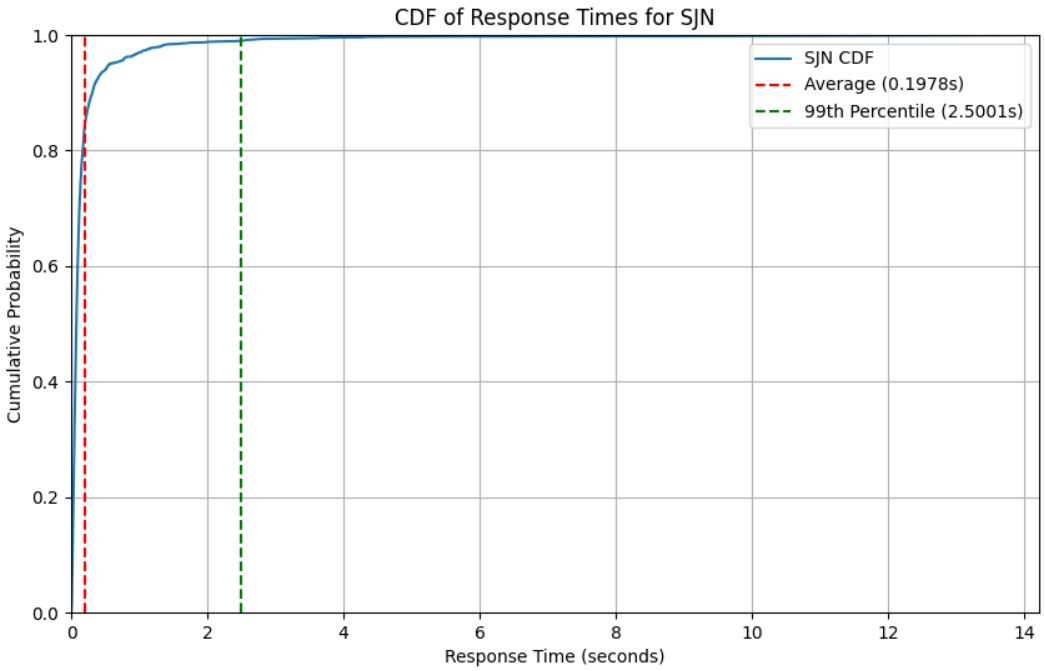

Figure 3: CDF of SJN

SJN - Average: 0.1978s, 99th Percentile: 2.5001s

(d) **Differences between the two CDF plots**

The CDF of FIFO shows a narrower response time distribution, indicating greater predictability. Tasks are processed in the order they arrive, ensuring a consistent and small difference between average response time (0.5225s) and the 99th percentile (1.3256s). This makes FIFO easier to predict and plan around, particularly in real-time systems.

On the other hand, SJN's CDF reveals a wider response time distribution. While it achieves a lower average response time (0.1978s), the large gap to the 99th percentile (2.5001s) shows more variability, especially for longer tasks. This unpredictability can lead to delayed job completion under heavy loads, which is a disadvantage in systems that prioritize meeting worst-case deadlines.

Based on the textbook's definition of predictability (minimizing the gap between BCET and WCET), FIFO is more predictable. It offers stable, consistent performance, while SJN trades predictability for better average performance at the risk of delaying longer jobs.

In conclusion, FIFO provides more predictable system behavior, making it better suited to real-time systems where deadlines are critical, while SJN offers higher performance for shorter tasks but with increased variability.