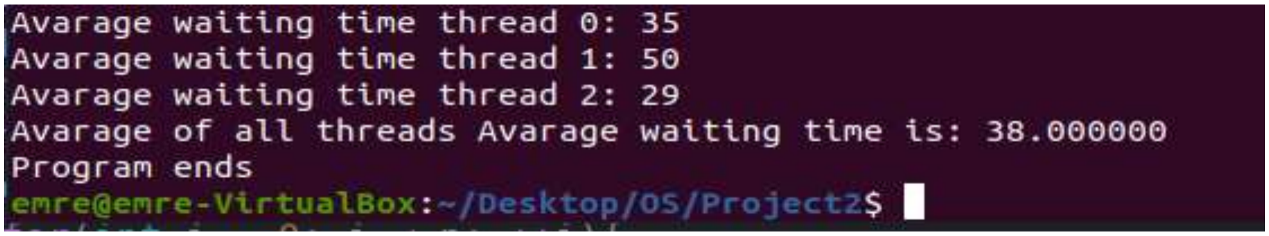
Muhammed Emre YILDIZ

21702025

CS342 Project-2

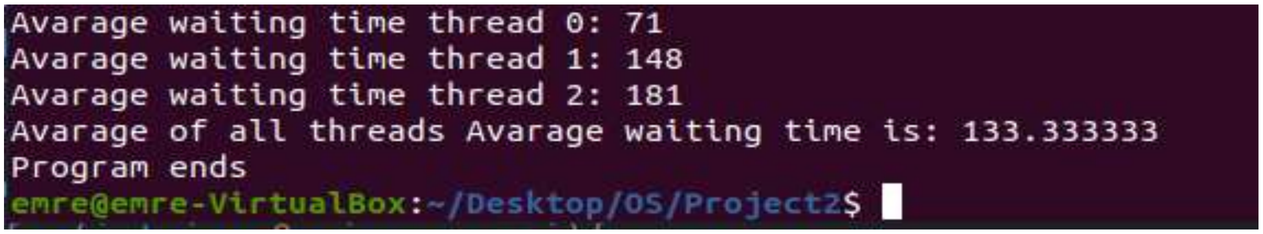
./schedule 3 30 100 200 1000 1500 FCFS



```
Avarage waiting time thread 0: 35
Avarage waiting time thread 1: 50
Avarage waiting time thread 2: 29
Avarage of all threads Avarage waiting time is: 38.000000
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
```

Fig.1

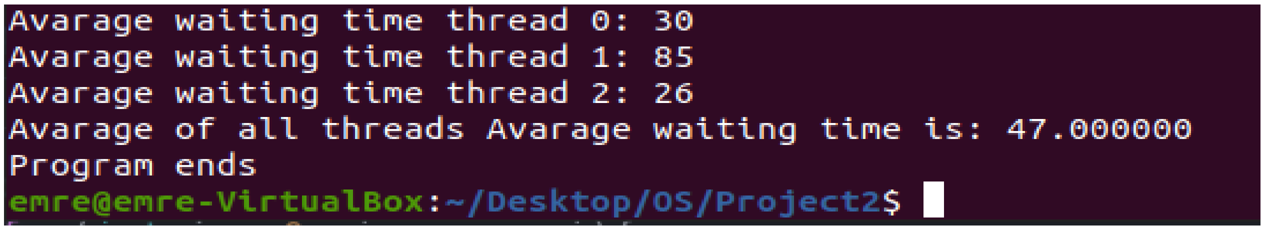
./schedule 3 30 100 200 1000 1500 SJF



```
Avarage waiting time thread 0: 71
Avarage waiting time thread 1: 148
Avarage waiting time thread 2: 181
Avarage of all threads Avarage waiting time is: 133.333333
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
```

Fig.2

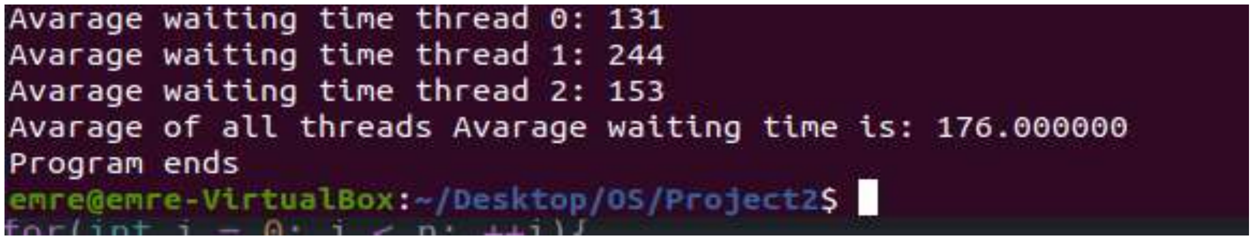
./schedule 3 30 100 200 1000 1500 PRIO



```
Avarage waiting time thread 0: 30
Avarage waiting time thread 1: 85
Avarage waiting time thread 2: 26
Avarage of all threads Avarage waiting time is: 47.000000
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
```

Fig.3

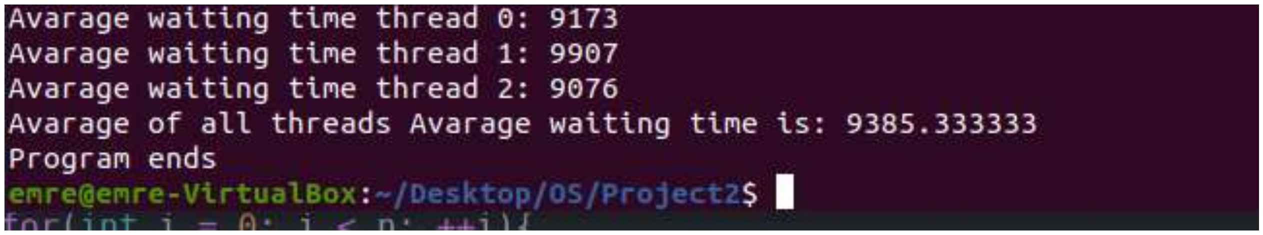
./schedule 3 30 100 200 1000 1500 VRUNTIME



```
Avarage waiting time thread 0: 131
Avarage waiting time thread 1: 244
Avarage waiting time thread 2: 153
Avarage of all threads Avarage waiting time is: 176.000000
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
```

Fig.4

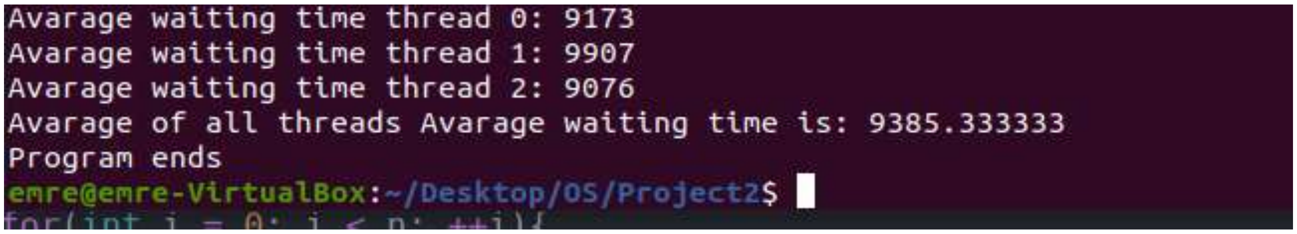
./schedule 3 30 100 200 100 200 FCFS



```
Avarage waiting time thread 0: 9173
Avarage waiting time thread 1: 9907
Avarage waiting time thread 2: 9076
Avarage of all threads Avarage waiting time is: 9385.333333
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
for(int i = 0; i < n; ++i){
```

Fig.5

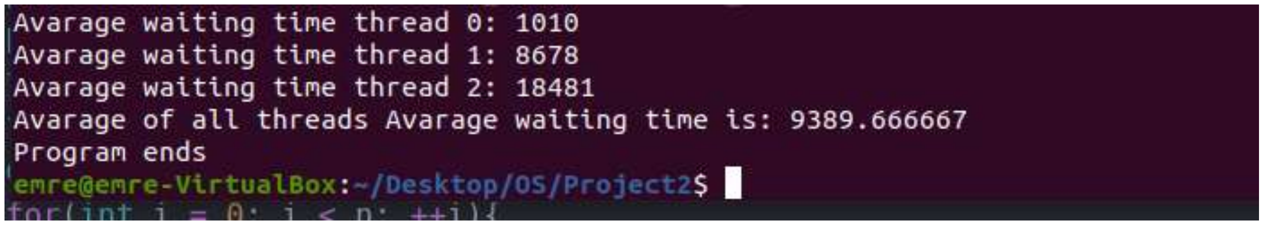
./schedule 3 30 100 200 100 200 SJF



```
Avarage waiting time thread 0: 9173
Avarage waiting time thread 1: 9907
Avarage waiting time thread 2: 9076
Avarage of all threads Avarage waiting time is: 9385.333333
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
for(int i = 0; i < n; ++i){
```

Fig.6

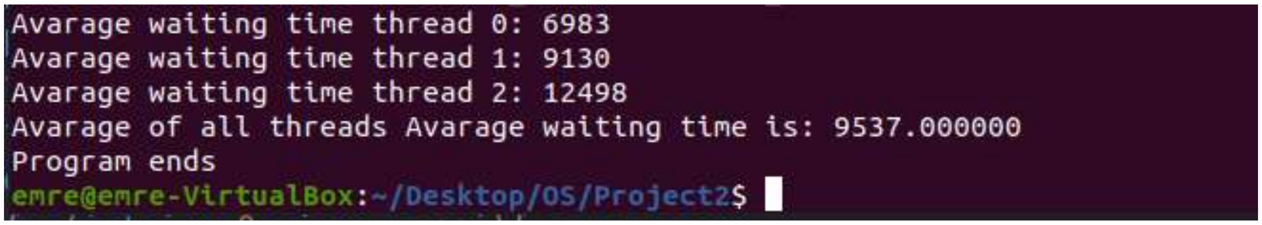
./schedule 3 30 100 200 100 200 PRIO



```
Avarage waiting time thread 0: 1010
Avarage waiting time thread 1: 8678
Avarage waiting time thread 2: 18481
Avarage of all threads Avarage waiting time is: 9389.666667
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
for(int i = 0; i < n; ++i){
```

Fig.7

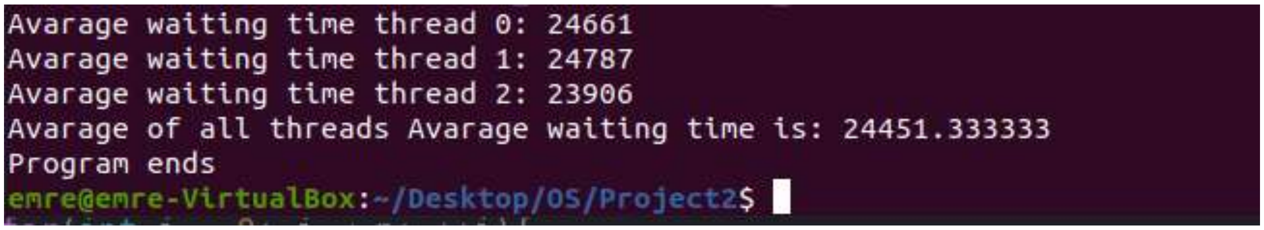
./schedule 3 30 100 200 100 200 VRUNTIME



```
Avarage waiting time thread 0: 6983
Avarage waiting time thread 1: 9130
Avarage waiting time thread 2: 12498
Avarage of all threads Avarage waiting time is: 9537.000000
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
```

Fig.8

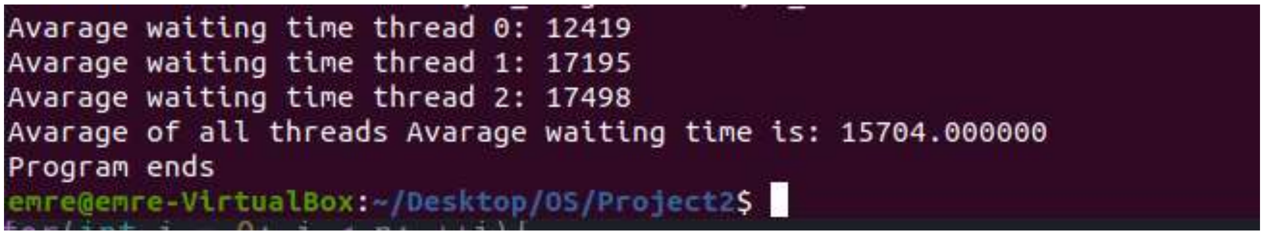
./schedule 3 30 300 350 100 200 FCFS



```
Avarage waiting time thread 0: 24661
Avarage waiting time thread 1: 24787
Avarage waiting time thread 2: 23906
Avarage of all threads Avarage waiting time is: 24451.333333
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
```

Fig.9

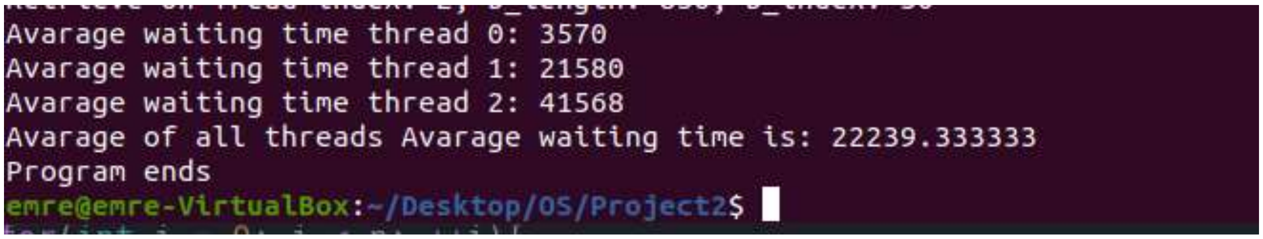
./schedule 3 30 300 350 100 200 FCFS



```
Avarage waiting time thread 0: 12419
Avarage waiting time thread 1: 17195
Avarage waiting time thread 2: 17498
Avarage of all threads Avarage waiting time is: 15704.000000
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
```

Fig.10

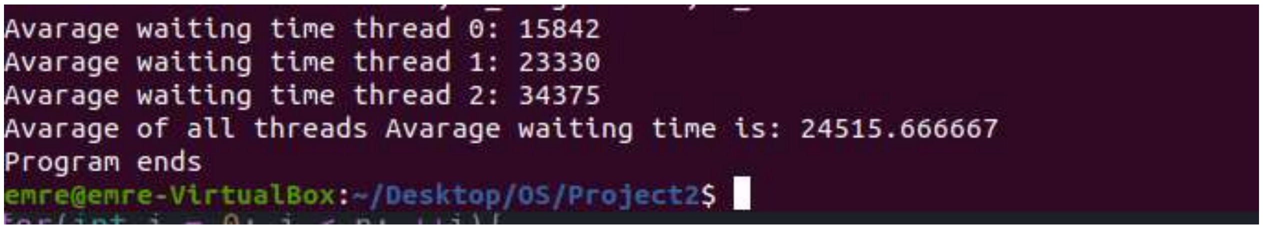
./schedule 3 30 300 350 100 200 FCFS



```
Avarage waiting time thread 0: 3570
Avarage waiting time thread 1: 21580
Avarage waiting time thread 2: 41568
Avarage of all threads Avarage waiting time is: 22239.333333
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
```

Fig.11

./schedule 3 30 300 350 100 200 FCFS



```
Avarage waiting time thread 0: 15842
Avarage waiting time thread 1: 23330
Avarage waiting time thread 2: 34375
Avarage of all threads Avarage waiting time is: 24515.666667
Program ends
emre@emre-VirtualBox:~/Desktop/OS/Project2$
```

Fig.12

FCFS	**SJF**	**PRIO**	**VRUNTIME**
38	133.33	47	176
9385.33	5101.33	9389.667	9537
24451.33	15704	22239	24515

Conclusion:

1- Firstly, avarage b and minb are smaller than avarage a and min a. It means approximately scheduler wait values are similar with threads therefore when threads do instertion, the scheduler is nearly ready to retrieve, therefore waiting times are much more smaller than other cases. When these values are used, the best algorithm is FCFS because ready queue is nearly empty. Every values can be retrieved after they are just inserted.

2- Secondly, avarageb and minb are equal to avarage a and min a. It means scheduler and threads wait equal times. Therefore worked thread number is greater than scheduler thread number which is 1. Therefıre worker threads have more opportunity to insert. Thus, waiting times get higher. In this case, SJF is the best algorithm to use. Because it can select small jobs. That is why it gets best values.

3- Thirdly, avarageb is bigger than avarage a and min a. It means, even work thread number is higher than scheduler thread, they also wait less. So, they can do much more insert than scheduler thread retrieve. That’s why, this case causes much more waiting time and similar to second case, best result is given by SJF.