Matthew Shiplett (mss0033)
Comp-7500
Project 3: AU Batch

Project 3: AU Batch Report

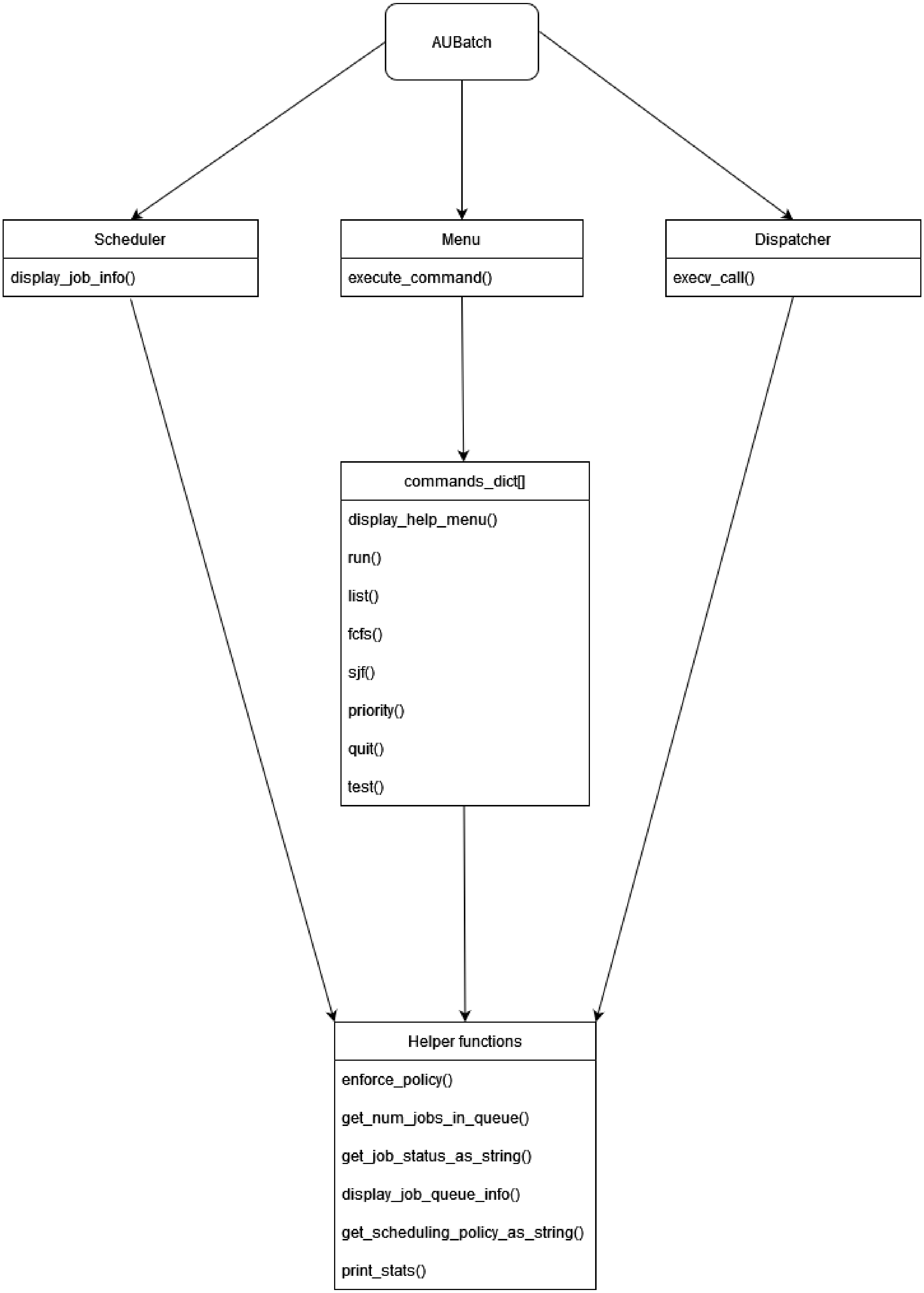
Design and Implementation:

For this project, I designed a multi-threaded program which primarily consists of running 3 threads. The primary 3 threads are the menu thread, the scheduling thread, and the dispatching thread. The menu thread, as the name suggest, is responsible for displaying the program's menu, as well as taking in user input. This thread makes calls to methods which handle the user input in the form of running commands or printing error messages. The scheduling thread is primarily responsible for taking the job information provided by the 'run' command and scheduling the job by placing the job into the job queue. The dispatcher thread is primarily responsible for take items placed into the job queue and making the call to 'execv' to run the supplied job.

Additionally, many helper functions where implemented ranging form gathering the number of jobs in the queue, calculating the total remaining CPU time for the waiting jobs, displaying portions of the menu, and many more.

Please see the below images for a visual representation of the implementation:

Matthew Shiplett (mss0033)
Comp-7500
Project 3: AU Batch



Performance metrics and workload conditions:

Tests performed:

1. Benchmark: **batch_job**, Policy: **SJF**, Number of Jobs: **5**, Priority Levels: **5**, Min CPU Time: **5 seconds**, Max CPU Time: **15 seconds**
    a. Average turnaround time: **19.000 seconds**
    b. Average CPU time: **7.000 seconds**
    c. Average waiting time: **19.000 seconds**
    d. Throughput: **0.053 No./second**
2. Benchmark: **batch_job**, Policy: **Priority**, Number of Jobs: **5**, Priority Levels: **5**, Min CPU Time: **5 seconds**, Max CPU Time: 15 seconds
    a. Average turnaround time: **26.000 seconds**
    b. Average CPU time: **8.000 seconds**
    c. Average waiting time: **26.000 seconds**
    d. Throughput: **0.038 No./second**
3. Benchmark: **batch_job**, Policy: **FCFS**, Number of Jobs: **5**, Priority Levels: **5**, Min CPU Time: **5 seconds**, Max CPU Time: **15 seconds**
    a. Average turnaround time: **26.000 seconds**
    b. Average CPU time: **7.000 seconds**
    c. Average waiting time: **26.000 seconds**
    d. Throughput: **0.038 No./second**

Frist of all, I would like to apologize for the limited testing, but I am completing this project very close to the deadline. From the testing above both Priority and FCFS scheduling achieve the same throughput of 26 seconds. However, Priority achieved that with an average CPU time of 8 seconds, where as FCFS had 7 seconds of average CPU time. Admittedly, to come to a solid conclusion of which method is better, more testing would be required.

Lessons Learned:

Despite my years of experience, I am not above starting these projects as early as possible. Additionally, I retained less information on the practices of C programming from my time as an undergraduate than I had anticipated. This hubris has caused me great pain during this project, and will likely also cost me a great many points on the overall grade. I need to spend some time understanding how the 'execv', 'wait', 'waitpid', and pthreads work.

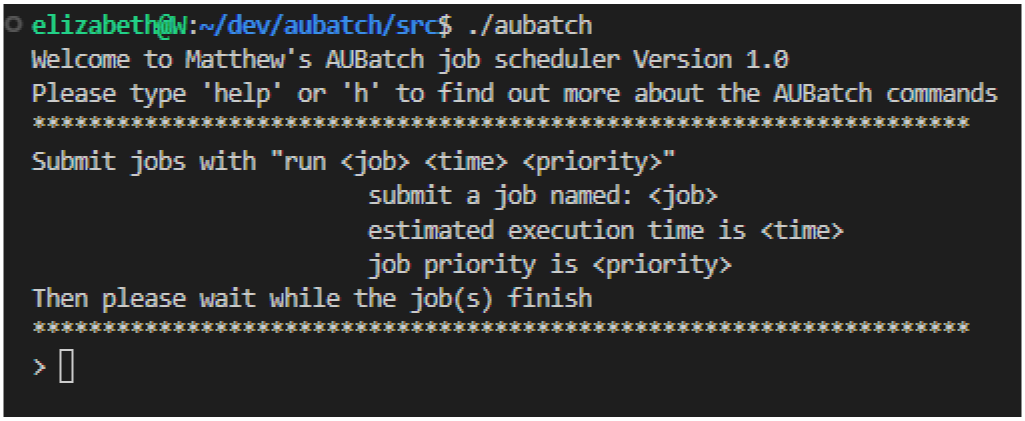
On a more positive note, I did gain a new appreciation for C, pointers, structs, and threading. I gained a new understanding of the pthread library, mutual exclusion, condition variables, signaling, as well as libraries such as time and ctype. This project also helped reinforce my understanding of separate compilation. Additionally, I believe I have a greater understanding of how scheduling works under the hood, as well as what sort of metrics can be used to evaluate it, and what better and worse performance scheduling looks like at a low level.

As for which scheduling method achieves the best performance, as I stated in the above performance section, I would need to do additional testing to come to a more solid conclusion. Based on my limited results, I would conclude SJF is likely to come out on top more often than not.

Sample I/O:

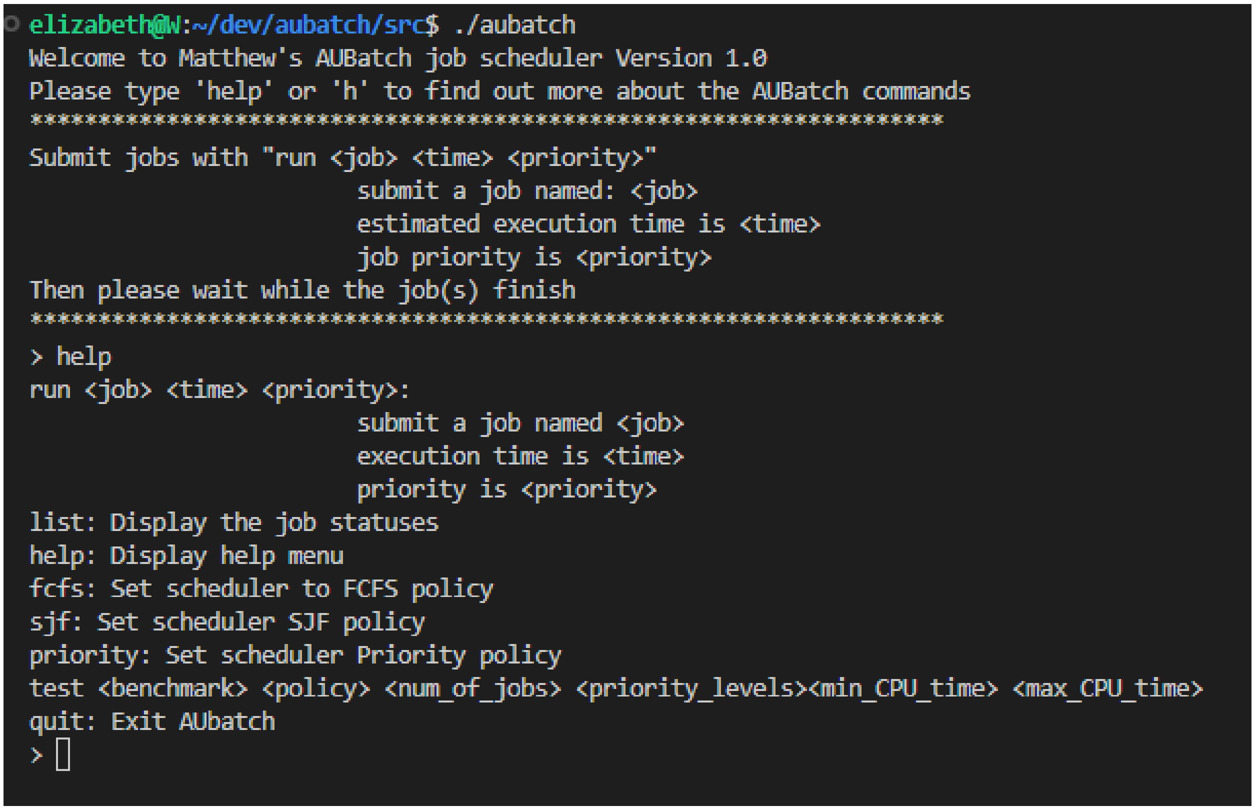
The following is a sample of the I/O that can be expected form the program:

- Program start:



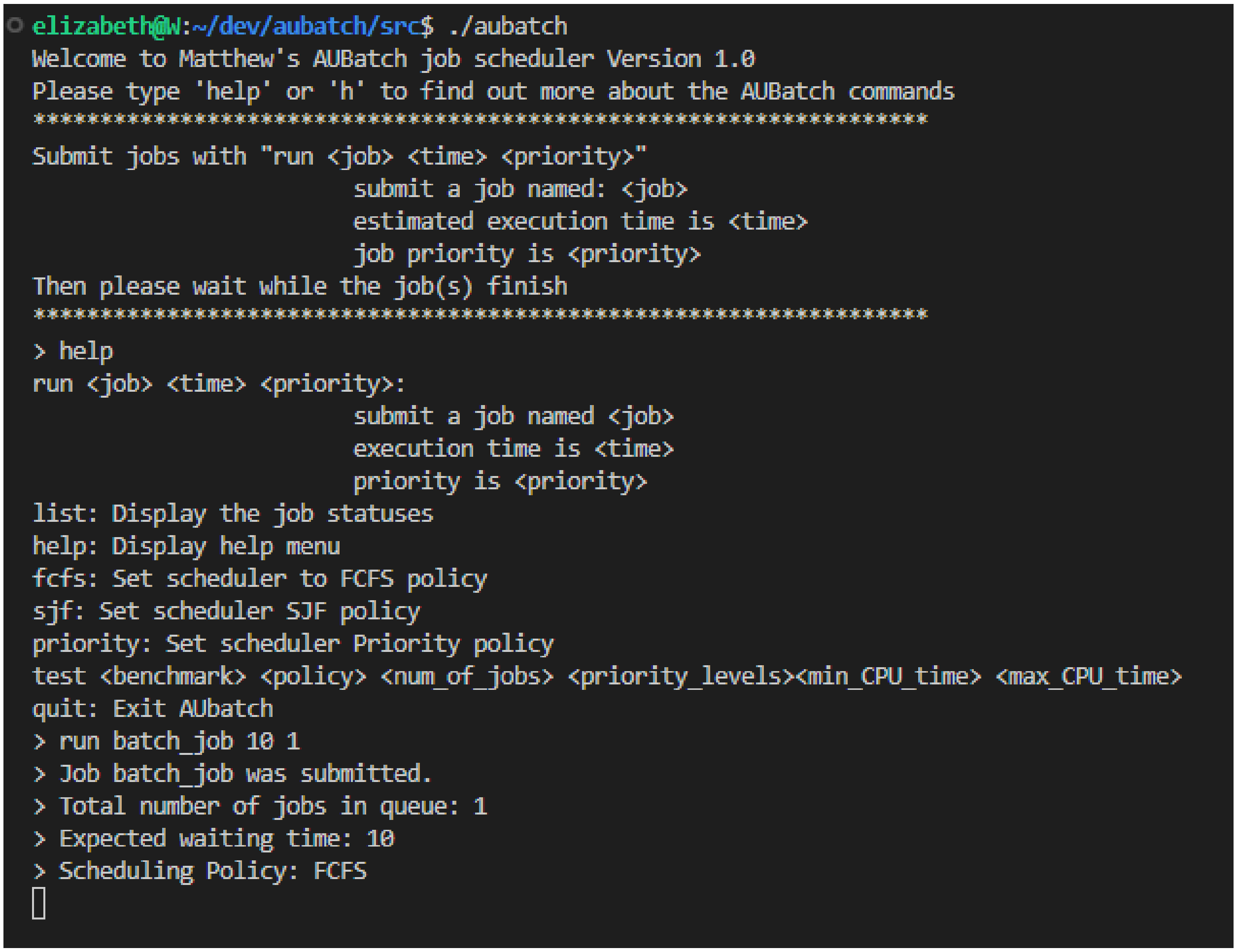
```
elizabeth@W:~/dev/aubatch/src$ ./aubatch
Welcome to Matthew's AUBatch job scheduler Version 1.0
Please type 'help' or 'h' to find out more about the AUBatch commands
*******************************************************************
Submit jobs with "run <job> <time> <priority>"
                        submit a job named: <job>
                        estimated execution time is <time>
                        job priority is <priority>
Then please wait while the job(s) finish
*******************************************************************
> 
```

- Typing 'help':



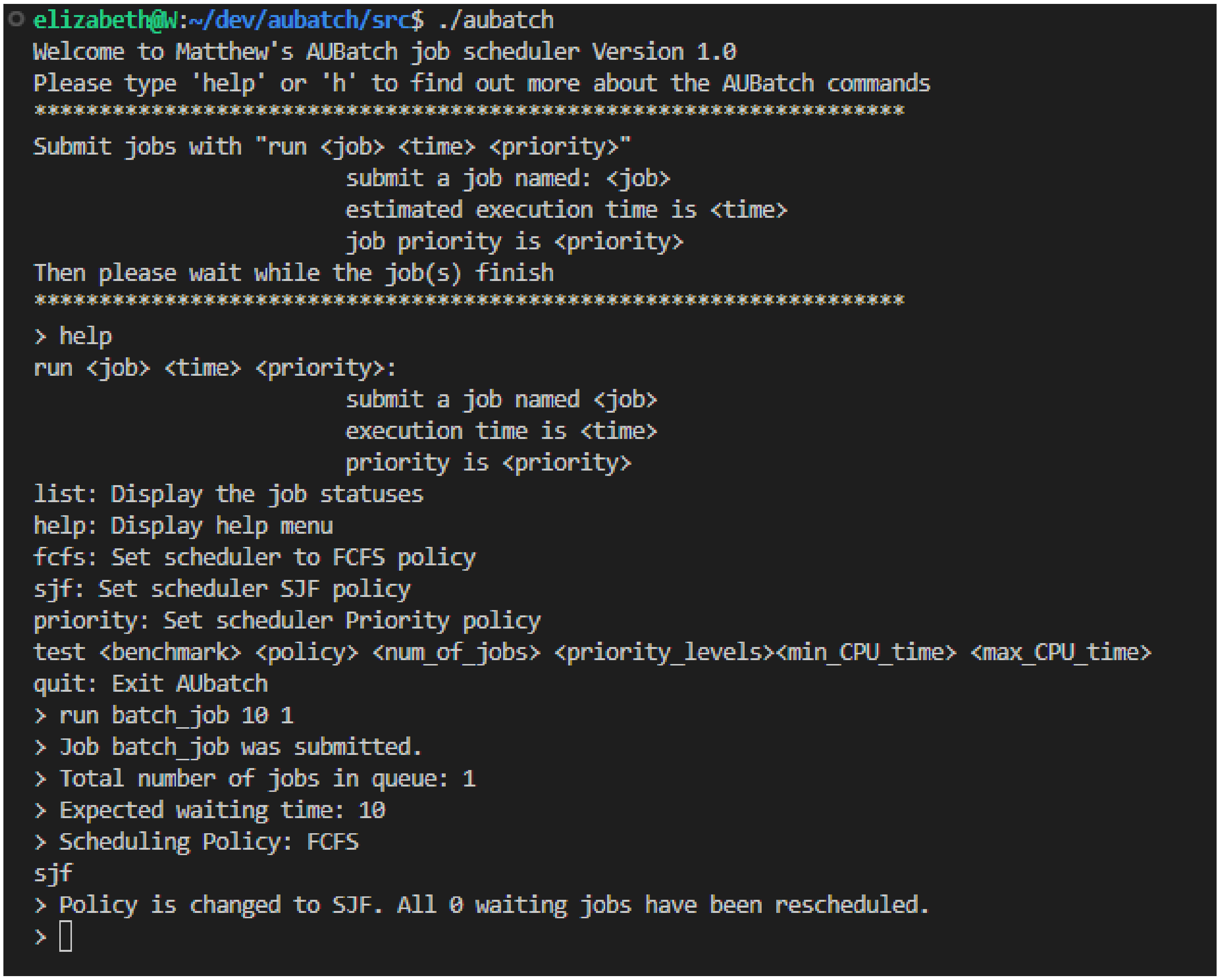
```
elizabeth@W:~/dev/aubatch/src$ ./aubatch
Welcome to Matthew's AUBatch job scheduler Version 1.0
Please type 'help' or 'h' to find out more about the AUBatch commands
*******************************************************************
Submit jobs with "run <job> <time> <priority>"
                        submit a job named: <job>
                        estimated execution time is <time>
                        job priority is <priority>
Then please wait while the job(s) finish
*******************************************************************
> help
run <job> <time> <priority>:
                        submit a job named <job>
                        execution time is <time>
                        priority is <priority>
list: Display the job statuses
help: Display help menu
fcfs: Set scheduler to FCFS policy
sjf: Set scheduler SJF policy
priority: Set scheduler Priority policy
test <benchmark> <policy> <num_of_jobs> <priority_levels><min_CPU_time> <max_CPU_time>
quit: Exit AUbatch
> 
```

- Typing ‘run batch_job 10 1’



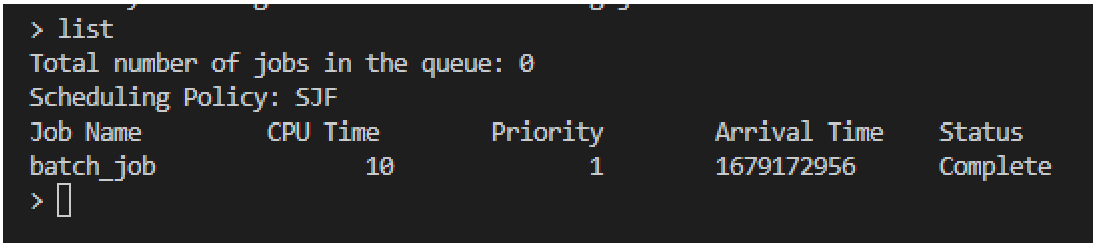
```
elizabeth@W:~/dev/aubatch/src$ ./aubatch
Welcome to Matthew's AUBatch job scheduler Version 1.0
Please type 'help' or 'h' to find out more about the AUBatch commands
*******************************************************************
Submit jobs with "run <job> <time> <priority>"
                        submit a job named: <job>
                        estimated execution time is <time>
                        job priority is <priority>
Then please wait while the job(s) finish
*******************************************************************
> help
run <job> <time> <priority>:
                        submit a job named <job>
                        execution time is <time>
                        priority is <priority>
list: Display the job statuses
help: Display help menu
fcfs: Set scheduler to FCFS policy
sjf: Set scheduler SJF policy
priority: Set scheduler Priority policy
test <benchmark> <policy> <num_of_jobs> <priority_levels><min_CPU_time> <max_CPU_time>
quit: Exit AUbatch
> run batch_job 10 1
> Job batch_job was submitted.
> Total number of jobs in queue: 1
> Expected waiting time: 10
> Scheduling Policy: FCFS
```

- Typing ‘sjf’:



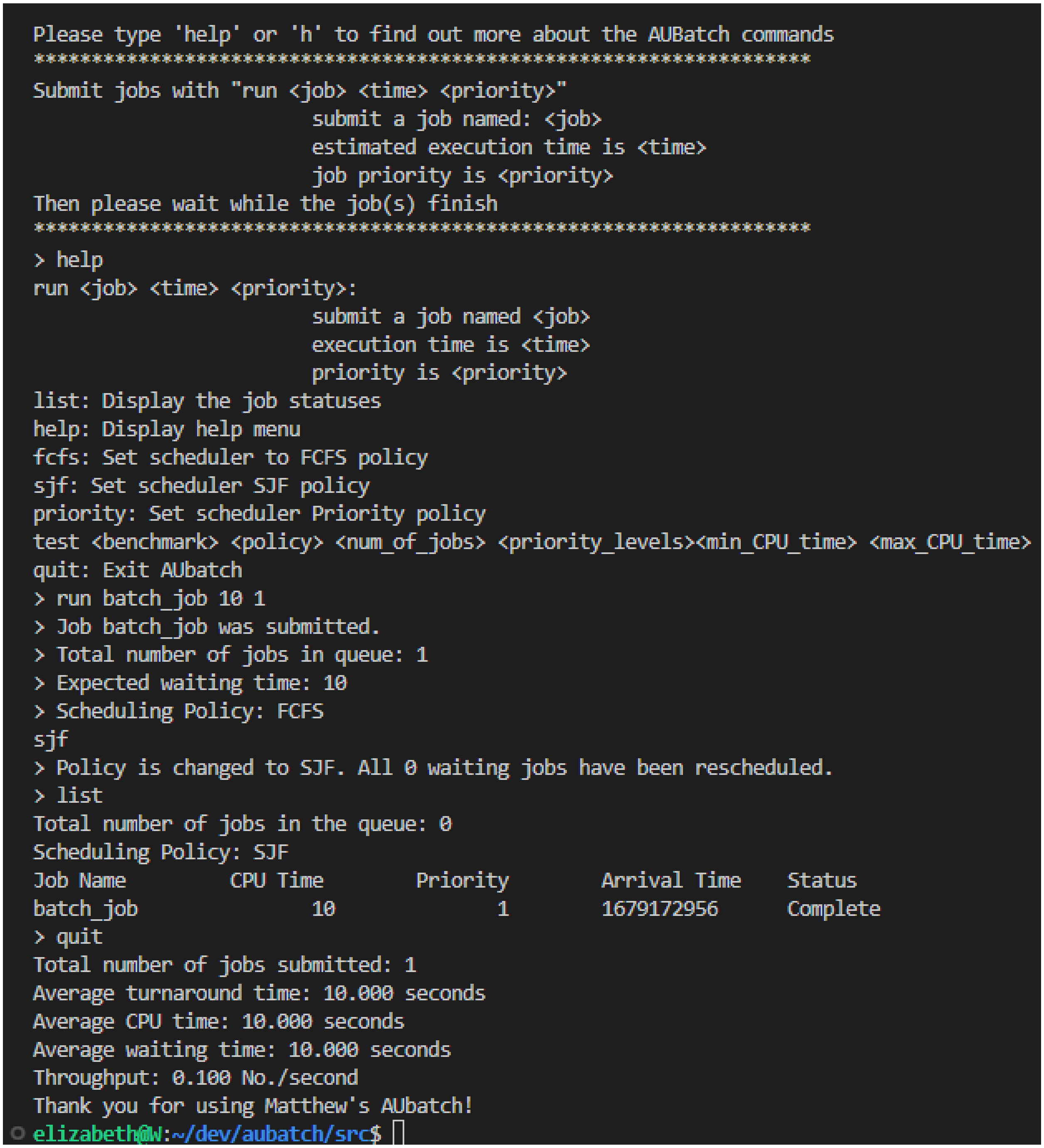
```
elizabeth@W:~/dev/aubatch/src$ ./aubatch
Welcome to Matthew's AUBatch job scheduler Version 1.0
Please type 'help' or 'h' to find out more about the AUBatch commands
*******************************************************************
Submit jobs with "run <job> <time> <priority>"
                        submit a job named: <job>
                        estimated execution time is <time>
                        job priority is <priority>
Then please wait while the job(s) finish
*******************************************************************
> help
run <job> <time> <priority>:
                        submit a job named <job>
                        execution time is <time>
                        priority is <priority>
list: Display the job statuses
help: Display help menu
fcfs: Set scheduler to FCFS policy
sjf: Set scheduler SJF policy
priority: Set scheduler Priority policy
test <benchmark> <policy> <num_of_jobs> <priority_levels><min_CPU_time> <max_CPU_time>
quit: Exit AUbatch
> run batch_job 10 1
> Job batch_job was submitted.
> Total number of jobs in queue: 1
> Expected waiting time: 10
> Scheduling Policy: FCFS
sjf
> Policy is changed to SJF. All 0 waiting jobs have been rescheduled.
> 
```

- Typing ‘list’:



```
> list
Total number of jobs in the queue: 0
Scheduling Policy: SJF
Job Name         CPU Time        Priority        Arrival Time    Status
batch_job               10              1        1679172956      Complete
> 
```

- Typing ‘quit’:



```
Please type 'help' or 'h' to find out more about the AUBatch commands
*******************************************************************
Submit jobs with "run <job> <time> <priority>"
                        submit a job named: <job>
                        estimated execution time is <time>
                        job priority is <priority>
Then please wait while the job(s) finish
*******************************************************************
> help
run <job> <time> <priority>:
                        submit a job named <job>
                        execution time is <time>
                        priority is <priority>
list: Display the job statuses
help: Display help menu
fcfs: Set scheduler to FCFS policy
sjf: Set scheduler SJF policy
priority: Set scheduler Priority policy
test <benchmark> <policy> <num_of_jobs> <priority_levels><min_CPU_time> <max_CPU_time>
quit: Exit AUbatch
> run batch_job 10 1
> Job batch_job was submitted.
> Total number of jobs in queue: 1
> Expected waiting time: 10
> Scheduling Policy: FCFS
sjf
> Policy is changed to SJF. All 0 waiting jobs have been rescheduled.
> list
Total number of jobs in the queue: 0
Scheduling Policy: SJF
Job Name         CPU Time        Priority        Arrival Time    Status
batch_job               10              1        1679172956      Complete
> quit
Total number of jobs submitted: 1
Average turnaround time: 10.000 seconds
Average CPU time: 10.000 seconds
Average waiting time: 10.000 seconds
Throughput: 0.100 No./second
Thank you for using Matthew's AUbatch!
elizabeth@W:~/dev/aubatch/src$ 
```