

Project 5 Report

🕒 Created	@December 1, 2024 11:43 AM
📁 Class	OS

▼ Solution Description

1. How did you separate scheduling mechanism from scheduling policies?

Each scheduling policy is implemented as its own function (`simulate_fcfs` , `simulate_rr` , `simulate_srtf`), called from `main` based on command line arguments provided by the user.

I did the format of the command line arguments as shown in the Project Specification:

```
./scheduler <input_task_file> <FCFS| RR | SRTF> [time_quantum]
```

The user need only select one of the policies by specifying it correctly in the command

2. How did you implement the three scheduling algorithms?

I implemented the scheduling algorithms in their own functions, as I stated above.

`FCFS` runs tasks to completion in the order they arrive.

`SRTF` runs tasks with the shortest remaining time.

`RR` runs tasks in cycles for a certain amount of time based on the `time_quantum`

3. How did you calculate waiting times?

Waiting times can be calculated as follows:

$$\text{Waiting Time} = \text{Turnaround Time} - \text{Burst Time}$$

4. How did you calculate response times?

Response times can be calculated as follows:

$$\text{Response Time} = \text{Start Time} - \text{Arrival Time}$$

5. How did you calculate turnaround times?

Turnaround times can be calculated as:

$$\text{Turnaround Time} = \text{Finish Time} - \text{Arrival Time}$$

6. How did you implement the command-line parser?

We use `argc` and `*argv` to grab arguments from the command line. The filename to parse the tasks is supplied. The user will also write in `FCFS`, `RR`, or `SRTF`. Then based off of the policy selected, we will call that scheduling policy function. If `RR` is called, we use `argc` to make sure that the `time_quantum` value was supplied.

▼ Generality and Error Checking

1. How general is your solution?

It is very general due to the three different scheduling policies. I made separation of concerns a priority, so it is very robust and extensible.

2. How easy would it be to add a new scheduling policy into your scheduler?

It would be very easy to add a new policy. I would only need to implement a function for that scheduling policy, and add the access to it via the command line parser.

3. Does your program offer input error checking?

Yes, the program validates inputs to ensure proper usage. It checks that the correct number of arguments is provided, the scheduling policy is valid (`FCFS` , `RR` , or `SRTF`), and, for `RR` , that a positive time quantum is specified. If any check fails, it displays a clear error message and exits safely. (`return EXIT_FAILURE;`)

▼ Miscellaneous Factors

1. Is your code elegant?

Yes, the code has a great separation of concerns, making it very modular. I used docstrings to give the code autocomplete/intellisense. Everything is well commented.

2. How innovative is your solution? Did you try any ideas not suggested here?

My solution is not very innovative. My implementation is very clean, but pretty standard.

3. Did you document all outside sources?

Yes.

▼ Samples

FCFS w/ 'task.list'

```

CPU_Scheduling_Project - zsh - 134x55
dutchcaz@Dutchs-MBP CPU_Scheduling_Project % ./scheduler task.list FCFS
<time 0> process 1 is running
<time 1> process 1 is running
<time 2> process 1 is running
<time 3> process 1 is running
<time 4> process 1 is running
<time 5> process 1 is running
<time 6> process 1 is running
<time 7> process 1 is running
<time 8> process 1 is running
<time 9> process 1 is running
<time 10> process 1 finished...
<time 10> process 2 is running
<time 11> process 2 is running
<time 12> process 2 is running
<time 13> process 2 is running
<time 14> process 2 is running
<time 15> process 2 is running
<time 16> process 2 is running
<time 17> process 2 is running
<time 18> process 2 is running
<time 19> process 2 finished...
<time 19> process 3 is running
<time 20> process 3 is running
<time 21> process 3 is running
<time 22> process 3 is running
<time 23> process 3 is running
<time 24> process 3 finished...
<time 24> process 4 is running
<time 25> process 4 is running
<time 26> process 4 is running
<time 27> process 4 is running
<time 28> process 4 finished...
<time 28> process 5 is running
<time 29> process 5 is running
<time 30> process 5 is running
<time 31> process 5 is running
<time 32> process 5 is running
<time 33> process 5 is running
<time 34> process 5 finished...
<time 34> process 6 is running
<time 35> process 6 is running
<time 36> process 6 is running
<time 37> process 6 is running
<time 38> process 6 is running
<time 39> process 6 is running
<time 40> process 6 is running
<time 41> process 6 finished...
<time 41> All processes finished...
===== Statistics =====
Average waiting time: 14.17
Average response time: 14.17
Average turnaround time: 21.00
Overall CPU usage: 100.00%
=====
```

RR w/ 'task.list' & time_quantum set to 4

```

CPU_Scheduling_Project - zsh - 134x55
dutchcaz@Dutchs-MBP CPU_Scheduling_Project % ./scheduler task.list RR 4
<time 0> process 1 is running
<time 1> process 1 is running
<time 2> process 1 is running
<time 3> process 1 is running
<time 4> process 2 is running
<time 5> process 2 is running
<time 6> process 2 is running
<time 7> process 2 is running
<time 8> process 3 is running
<time 9> process 3 is running
<time 10> process 3 is running
<time 11> process 3 is running
<time 12> process 1 is running
<time 13> process 1 is running
<time 14> process 1 is running
<time 15> process 1 is running
<time 16> process 4 is running
<time 17> process 4 is running
<time 18> process 4 is running
<time 19> process 4 is running
<time 20> process 4 is finished...
<time 20> process 2 is running
<time 21> process 2 is running
<time 22> process 2 is running
<time 23> process 2 is running
<time 24> process 5 is running
<time 25> process 5 is running
<time 26> process 5 is running
<time 27> process 5 is running
<time 28> process 6 is running
<time 29> process 6 is running
<time 30> process 6 is running
<time 31> process 6 is running
<time 32> process 3 is running
<time 33> process 3 is finished...
<time 33> process 1 is running
<time 34> process 1 is running
<time 35> process 1 is finished...
<time 35> process 2 is running
<time 36> process 2 is finished...
<time 36> process 5 is running
<time 37> process 5 is running
<time 38> process 5 is finished...
<time 38> process 6 is running
<time 39> process 6 is running
<time 40> process 6 is running
<time 41> process 6 is finished...
<time 41> All processes finished...
===== Statistics =====
Average waiting time: 22.00
Average response time: 8.33
Average turnaround time: 28.83
Overall CPU usage: 100.00%
=====
```

SRTF w/ 'task.list'

```

CPU_Scheduling_Project - zsh - 134x55
dutchcaz@Dutchs-MBP CPU_Scheduling_Project % ./scheduler task.list SRTF
<time 0> process 2 is running
<time 1> process 2 is running
<time 2> process 2 is running
<time 3> process 3 is running
<time 4> process 3 is running
<time 5> process 3 is running
<time 6> process 3 is running
<time 7> process 3 is running
<time 8> process 3 is finished...
<time 8> process 4 is running
<time 9> process 4 is running
<time 10> process 4 is running
<time 11> process 4 is running
<time 12> process 4 is finished...
<time 12> process 2 is running
<time 13> process 2 is running
<time 14> process 2 is running
<time 15> process 2 is running
<time 16> process 2 is running
<time 17> process 2 is running
<time 18> process 2 is finished...
<time 18> process 5 is running
<time 19> process 5 is running
<time 20> process 5 is running
<time 21> process 5 is running
<time 22> process 5 is running
<time 23> process 5 is running
<time 24> process 5 is finished...
<time 24> process 6 is running
<time 25> process 6 is running
<time 26> process 6 is running
<time 27> process 6 is running
<time 28> process 6 is running
<time 29> process 6 is running
<time 30> process 6 is running
<time 31> process 6 is finished...
<time 31> process 1 is running
<time 32> process 1 is running
<time 33> process 1 is running
<time 34> process 1 is running
<time 35> process 1 is running
<time 36> process 1 is running
<time 37> process 1 is running
<time 38> process 1 is running
<time 39> process 1 is running
<time 40> process 1 is running
<time 41> process 1 is finished...
<time 41> All processes finished...
===== Statistics =====
Average waiting time: 10.50
Average response time: 9.00
Average turnaround time: 17.33
Overall CPU usage: 100.00%
=====
```