2024-Spring

Operating System Project2 Report

Department: Computer Science & Engineering

2022320149 호수빈

Submission date: 2023/05/27

Number of Free days: 5

1. Development Environment

Window 11 / Virtual Box 7.0.10 / Ubuntu 18.04.2 / Linux kernel 4.20.11

2. Explanation of CPU scheduling and policies

Scheduling is a technique used to efficiently manage and execute multiple processes within a computer system. In an environment where multiple processes share and compete for system resources, scheduling ensures fair distribution of resources, thereby enhancing system performance.

Its objectives include fairness in resource allocation, maximizing CPU utilization efficiency, and minimizing response time.

1) FCFS

FCFS operates on the principle that the process that arrives first is executed first. However, it may suffer from drawbacks such as the convoy effect, leading to longer average waiting times.

2) SRTF

SRTF is an extension of the SJF algorithm with preemption. When a new process arrives, it checks the remaining execution time of the currently running process and switches to the new process if its remaining time is shorter. While SRTF minimizes the average waiting time, it can be challenging to accurately predict execution times.

3) RR

RR allocates a fixed time slice to each process, and when the allocated time expires, the system switches to the next process in line. RR ensures fairness and reduces response time.

4) Priority

Priority scheduling assigns a priority to each process and executes higher-priority processes before lower-priority ones. While beneficial for handling important or urgent tasks, priority scheduling may lead to issues such as starvation, where lower-priority processes may not receive CPU time even after extended waits.

3. Implementation

1) syscall_64.tbl

It's a system call table file that maps the numbers of the system calls to the names of the system-level functions used to invoke them. I added definitions for the system calls that will be used in sslab_my_queue.c.

2) syscalls.h

It's a file that defines the prototypes of system call functions and declares the functions using asmlinkage. I added definitions for the system call functions that will be used in sslab_KU_CPU.c. For example, "asmlinkage int sys_os2024_KU_CPU(char**, int);" means defining a system call function, sys_os2024_KU_CPU, which takes an integer parameter and char pointer parameter, and returns an integer value."

3) sslab_KU_CPU.c

The code defines four system calls, 'os2024_ku_cpu_fcfs', 'os_2024_ku_cpu_srtf', 'os_2024_ku_cpu_rr' and 'os2024_ku_cpu_priority' which implement scheduling operations with a waiting queue respectively.

- Ku_is_empty (void): Returns a boolean value indicating whether the queue is empty.
- Ku_is_full (void): Returns a boolean value indicating whether the queue is full.
- Ku_push (job_t job): If the queue is full or the specified process already exists in the queue, it returns an error; otherwise, it enqueues the value.
- Ku_pop (void): If the queue is empty, it returns an error; otherwise, it returns the front value of the queue and shifts the queue one position.
- Ku_is_new_id (pid_t pid): Checks the existence of the specified value in the queue.
- Ku_SRTF_pop (void): pops element that has the shortest jobTime of waiting queue.
- Ku_Priority_pop (void): Pops the element that has the highest priority of waiting queue.

3-1) os_2024_ku_cpu_fcfs

The system call takes the name and execution time of each process as arguments. If the CPU is currently in the IDLE state, it allocates the CPU to the newly arrived process. If the process that sent the request is currently running process, it checks if the process has completed; if so, it prints that the process has finished, sets the current state to IDLE if the queue is empty, otherwise it dequeues the next process from the queue. If a request comes in from a different process, it checks if the new process is not already in the queue; if not, it enqueues the new process into the queue and prints that the current CPU allocation is denied.

3-2) os_2024_ku_cpu_srtf

The system call takes the process name and execution time as arguments. In this case, the newjob variable receives the pid and execution time of the newly arrived process. If there is no new process and a process is currently running, it checks if the current job has finished; if so, it prints that the job has finished. If the queue is empty, it transitions the current state to the IDLE state; otherwise, it dequeues the next process from the queue. If a new process arrives and it is different from the current process, it compares the remaining execution time of the currently running process with the execution time of the new process and proceeds with the process with the shorter execution time. In this case, the remained process is enqueued into the waiting queue.

3-3) os2024_ku_cpu_rr

The system call takes the process name and execution time as arguments. If the CPU is in the IDLE state, it allocates the CPU to the newly arrived process. Otherwise, when checking the currently running process, if the execution time has ended, it prints that the process has completed; if the queue is empty, it transitions the CPU to the IDLE state. Otherwise, it dequeues a new process from the queue and allocates it to the CPU. When a new process arrives, it checks if the process exists in the queue, and if not, it enqueues the process into the queue. Then, it dequeues the new process so that a different process can be allocated at each time slice.

3-4) os_2024_ku_cpu_priority

The system call takes process name, execution time, and priority as arguments. If the current CPU is in IDLE state, it assigns the incoming process. While a process is in progress, if it completes, it prints "Process Finished", and if the queue is empty at this point, it transitions the CPU to IDLE state; otherwise, it pops the next process with the highest priority from the queue. If a new process comes in and it's different from the currently running one, it assigns the CPU to the higher priority process between the two, and enqueues the other process based on priority.

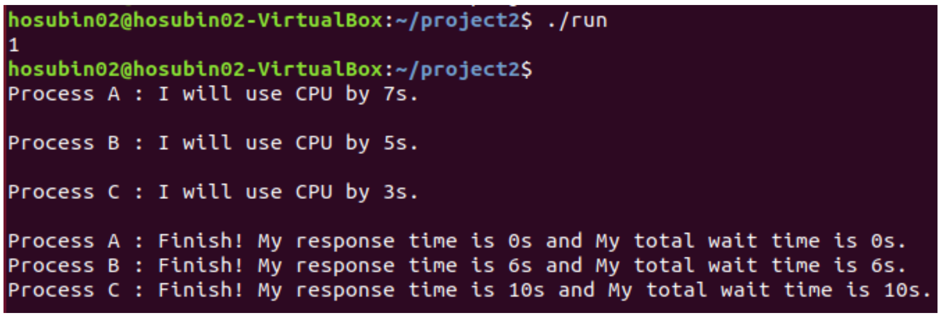
4) userprogram

I added a new variable called first to the existing skeleton code to measure the time when the CPU is initially allocated, in order to output the response time. If the process is assigned the CPU for the first time, set the first variable to 1 and set the waiting time as response. Additionally, I introduced other global variables. For priority scheduling, I add the priority argument to check the process' priority.

4. Experiment results and your analysis with screenshots of execution results(logs)

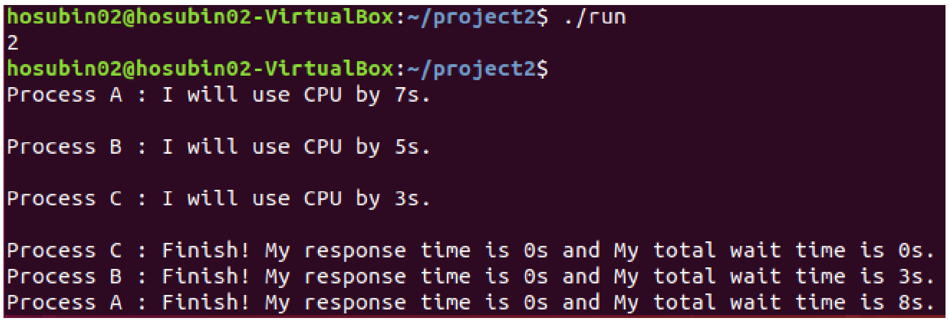
1) user application

FCFS (user process log from console)



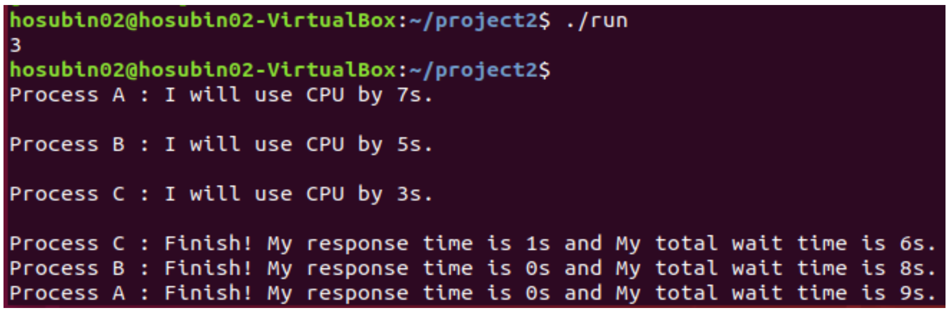
```
hosubin02@hosubin02-VirtualBox:~/project2$ ./run
1
hosubin02@hosubin02-VirtualBox:~/project2$
Process A : I will use CPU by 7s.

Process B : I will use CPU by 5s.

Process C : I will use CPU by 3s.

Process A : Finish! My response time is 0s and My total wait time is 0s.
Process B : Finish! My response time is 6s and My total wait time is 6s.
Process C : Finish! My response time is 10s and My total wait time is 10s.
```

SRTF (user process log from console)



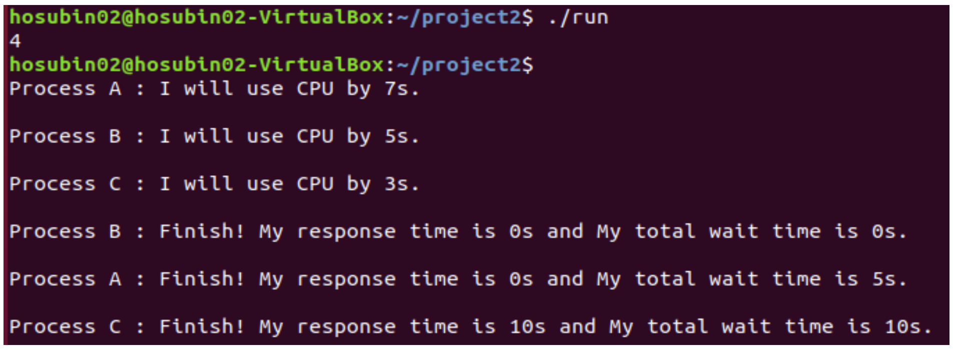
```
hosubin02@hosubin02-VirtualBox:~/project2$ ./run
2
hosubin02@hosubin02-VirtualBox:~/project2$
Process A : I will use CPU by 7s.

Process B : I will use CPU by 5s.

Process C : I will use CPU by 3s.

Process C : Finish! My response time is 0s and My total wait time is 0s.
Process B : Finish! My response time is 0s and My total wait time is 3s.
Process A : Finish! My response time is 0s and My total wait time is 8s.
```

RR (user process log from console)



```
hosubin02@hosubin02-VirtualBox:~/project2$ ./run
3
hosubin02@hosubin02-VirtualBox:~/project2$
Process A : I will use CPU by 7s.

Process B : I will use CPU by 5s.

Process C : I will use CPU by 3s.

Process C : Finish! My response time is 1s and My total wait time is 6s.
Process B : Finish! My response time is 0s and My total wait time is 8s.
Process A : Finish! My response time is 0s and My total wait time is 9s.
```

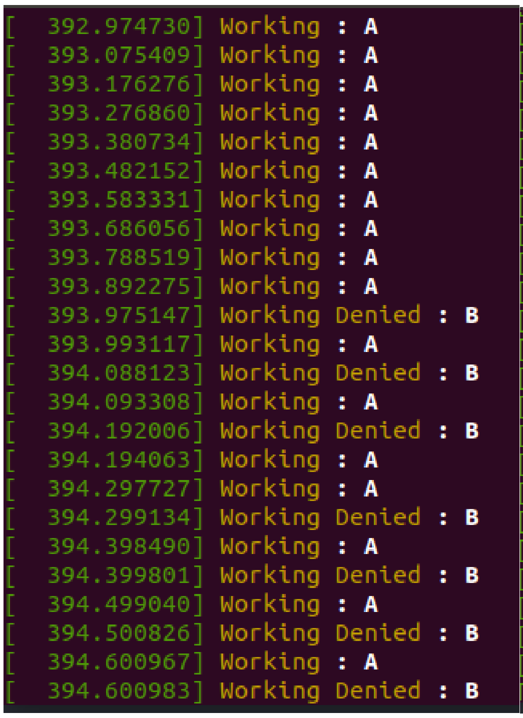
Priority (user process log from console)



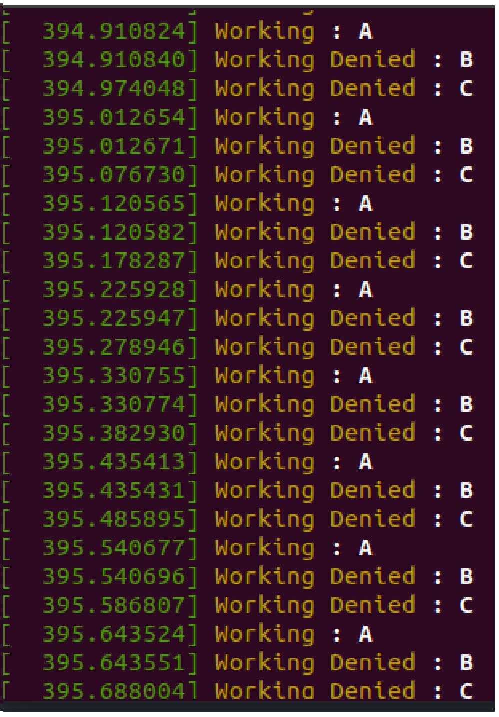
```
hosubin02@hosubin02-VirtualBox:~/project2$ ./run
4
hosubin02@hosubin02-VirtualBox:~/project2$
Process A : I will use CPU by 7s.

Process B : I will use CPU by 5s.

Process C : I will use CPU by 3s.

Process B : Finish! My response time is 0s and My total wait time is 0s.

Process A : Finish! My response time is 0s and My total wait time is 5s.

Process C : Finish! My response time is 10s and My total wait time is 10s.
```

2) dmesg

FCFS (kernel log from dmesg)

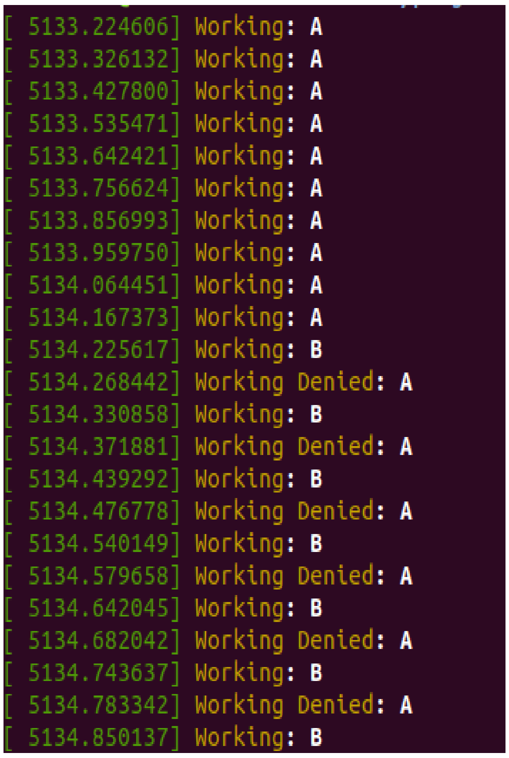


```
[  392.974730] Working : A
[  393.075409] Working : A
[  393.176276] Working : A
[  393.276860] Working : A
[  393.380734] Working : A
[  393.482152] Working : A
[  393.583331] Working : A
[  393.686056] Working : A
[  393.788519] Working : A
[  393.892275] Working : A
[  393.975147] Working Denied : B
[  393.993117] Working : A
[  394.088123] Working Denied : B
[  394.093308] Working : A
[  394.192006] Working Denied : B
[  394.194063] Working : A
[  394.297727] Working : A
[  394.299134] Working Denied : B
[  394.398490] Working : A
[  394.399801] Working Denied : B
[  394.499040] Working : A
[  394.500826] Working Denied : B
[  394.600967] Working : A
[  394.600983] Working Denied : B
```

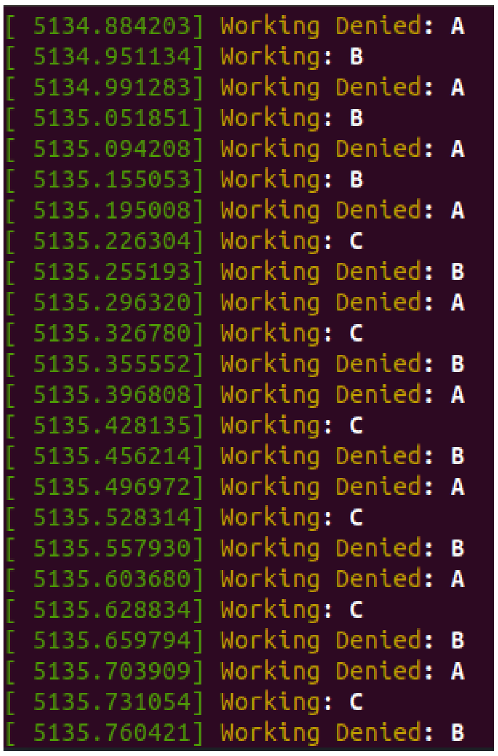


```
[  394.910824] Working : A
[  394.910840] Working Denied : B
[  394.974048] Working Denied : C
[  395.012654] Working : A
[  395.012671] Working Denied : B
[  395.076730] Working Denied : C
[  395.120565] Working : A
[  395.120582] Working Denied : B
[  395.178287] Working Denied : C
[  395.225928] Working : A
[  395.225947] Working Denied : B
[  395.278946] Working Denied : C
[  395.330755] Working : A
[  395.330774] Working Denied : B
[  395.382930] Working Denied : C
[  395.435413] Working : A
[  395.435431] Working Denied : B
[  395.485895] Working Denied : C
[  395.540677] Working : A
[  395.540696] Working Denied : B
[  395.586807] Working Denied : C
[  395.643524] Working : A
[  395.643551] Working Denied : B
[  395.688004] Working Denied : C
```

SRTF (kernel log form dmesg)

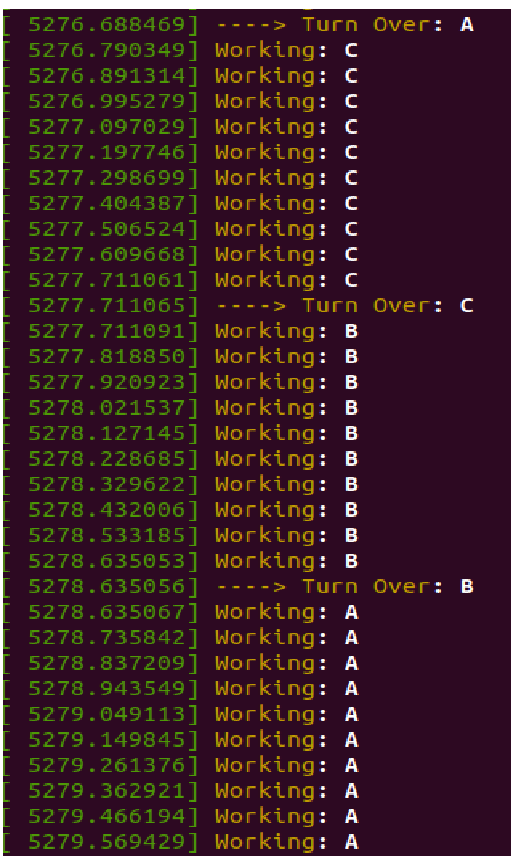


```
[ 5133.224606] Working: A
[ 5133.326132] Working: A
[ 5133.427800] Working: A
[ 5133.535471] Working: A
[ 5133.642421] Working: A
[ 5133.756624] Working: A
[ 5133.856993] Working: A
[ 5133.959750] Working: A
[ 5134.064451] Working: A
[ 5134.167373] Working: A
[ 5134.225617] Working: B
[ 5134.268442] Working Denied: A
[ 5134.330858] Working: B
[ 5134.371881] Working Denied: A
[ 5134.439292] Working: B
[ 5134.476778] Working Denied: A
[ 5134.540149] Working: B
[ 5134.579658] Working Denied: A
[ 5134.642045] Working: B
[ 5134.682042] Working Denied: A
[ 5134.743637] Working: B
[ 5134.783342] Working Denied: A
[ 5134.850137] Working: B
```

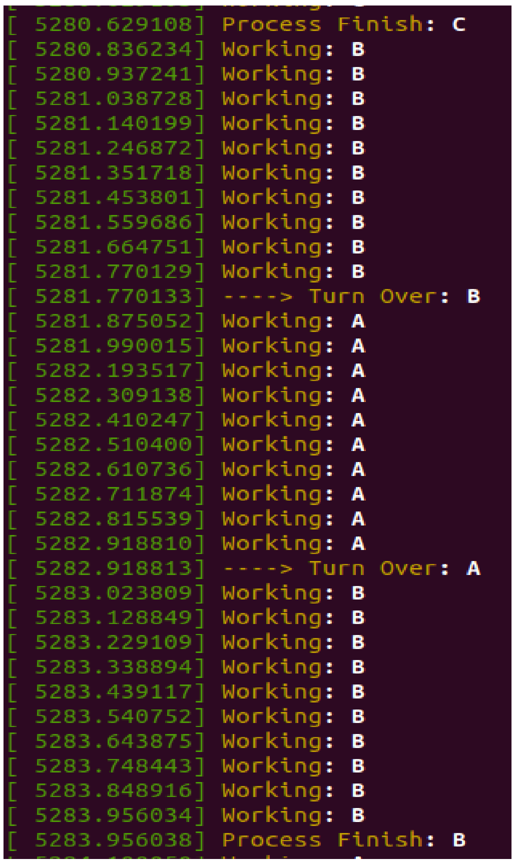


```
[ 5134.884203] Working Denied: A
[ 5134.951134] Working: B
[ 5134.991283] Working Denied: A
[ 5135.051851] Working: B
[ 5135.094208] Working Denied: A
[ 5135.155053] Working: B
[ 5135.195008] Working Denied: A
[ 5135.226304] Working: C
[ 5135.255193] Working Denied: B
[ 5135.296320] Working Denied: A
[ 5135.326780] Working: C
[ 5135.355552] Working Denied: B
[ 5135.396808] Working Denied: A
[ 5135.428135] Working: C
[ 5135.456214] Working Denied: B
[ 5135.496972] Working Denied: A
[ 5135.528314] Working: C
[ 5135.557930] Working Denied: B
[ 5135.603680] Working Denied: A
[ 5135.628834] Working: C
[ 5135.659794] Working Denied: B
[ 5135.703909] Working Denied: A
[ 5135.731054] Working: C
[ 5135.760421] Working Denied: B
```

RR (kernel log from dmesg)

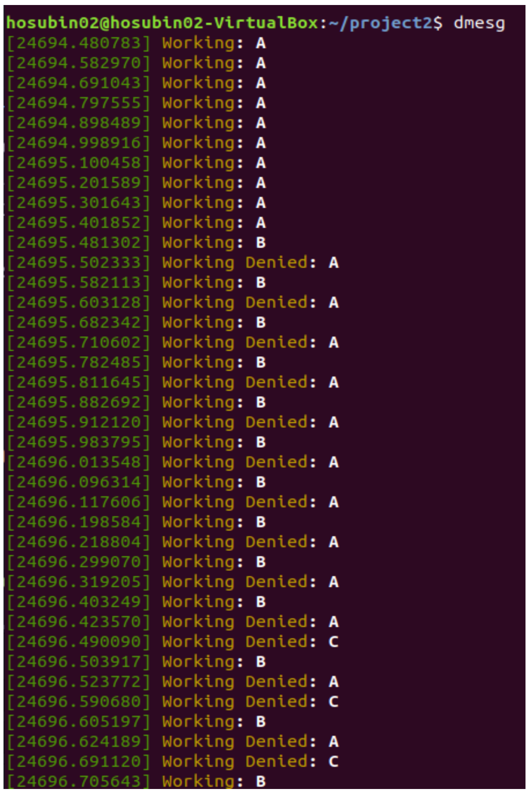


```
[  5276.688469] ----> Turn Over: A
[  5276.790349] Working: C
[  5276.891314] Working: C
[  5276.995279] Working: C
[  5277.097029] Working: C
[  5277.197746] Working: C
[  5277.298699] Working: C
[  5277.404387] Working: C
[  5277.506524] Working: C
[  5277.609668] Working: C
[  5277.711061] Working: C
[  5277.711065] ----> Turn Over: C
[  5277.711091] Working: B
[  5277.818850] Working: B
[  5277.920923] Working: B
[  5278.021537] Working: B
[  5278.127145] Working: B
[  5278.228685] Working: B
[  5278.329622] Working: B
[  5278.432006] Working: B
[  5278.533185] Working: B
[  5278.635053] Working: B
[  5278.635056] ----> Turn Over: B
[  5278.635067] Working: A
[  5278.735842] Working: A
[  5278.837209] Working: A
[  5278.943549] Working: A
[  5279.049113] Working: A
[  5279.149845] Working: A
[  5279.261376] Working: A
[  5279.362921] Working: A
[  5279.466194] Working: A
[  5279.569429] Working: A
```

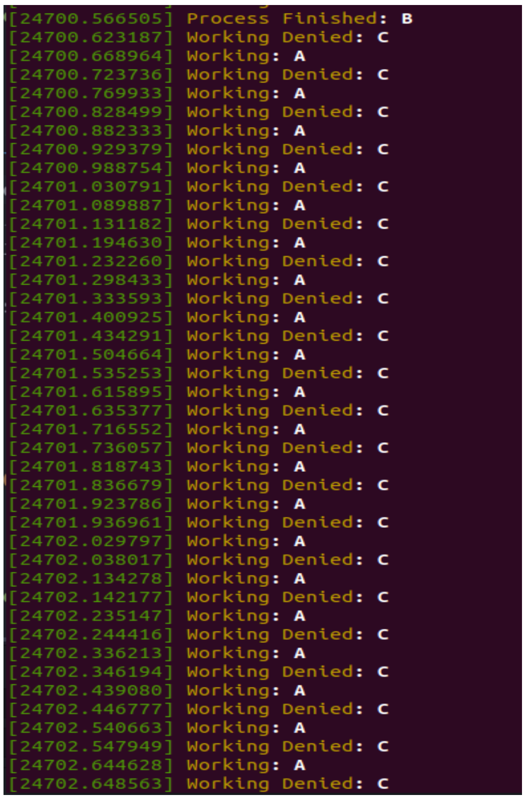


```
[  5280.629108] Process Finish: C
[  5280.836234] Working: B
[  5280.937241] Working: B
[  5281.038728] Working: B
[  5281.140199] Working: B
[  5281.246872] Working: B
[  5281.351718] Working: B
[  5281.453801] Working: B
[  5281.559686] Working: B
[  5281.664751] Working: B
[  5281.770129] Working: B
[  5281.770133] ----> Turn Over: B
[  5281.875052] Working: A
[  5281.990015] Working: A
[  5282.193517] Working: A
[  5282.309138] Working: A
[  5282.410247] Working: A
[  5282.510400] Working: A
[  5282.610736] Working: A
[  5282.711874] Working: A
[  5282.815539] Working: A
[  5282.918810] Working: A
[  5282.918813] ----> Turn Over: A
[  5283.023809] Working: B
[  5283.128849] Working: B
[  5283.229109] Working: B
[  5283.338894] Working: B
[  5283.439117] Working: B
[  5283.540752] Working: B
[  5283.643875] Working: B
[  5283.748443] Working: B
[  5283.848916] Working: B
[  5283.956034] Working: B
[  5283.956038] Process Finish: B
```

Priority (kernel log from dmesg)



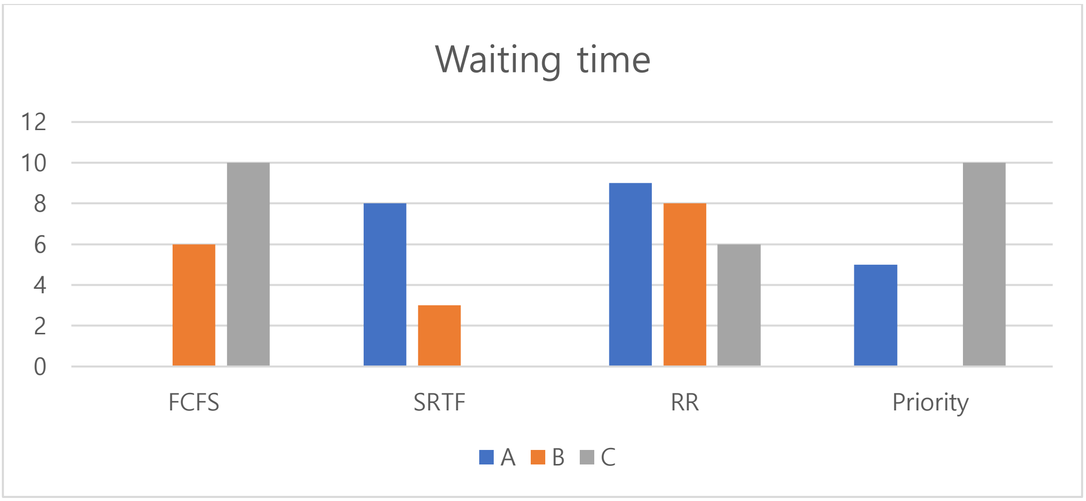
```
hosubin02@hosubin02-VirtualBox:~/project2$ dmesg
[24694.480783] Working: A
[24694.582970] Working: A
[24694.691043] Working: A
[24694.797555] Working: A
[24694.898489] Working: A
[24694.998916] Working: A
[24695.100458] Working: A
[24695.201589] Working: A
[24695.301643] Working: A
[24695.401852] Working: A
[24695.481302] Working: B
[24695.502333] Working Denied: A
[24695.582113] Working: B
[24695.603128] Working Denied: A
[24695.682342] Working: B
[24695.710602] Working Denied: A
[24695.782485] Working: B
[24695.811645] Working Denied: A
[24695.882692] Working: B
[24695.912120] Working Denied: A
[24695.983795] Working: B
[24696.013548] Working Denied: A
[24696.096314] Working: B
[24696.117606] Working Denied: A
[24696.198584] Working: B
[24696.218804] Working Denied: A
[24696.299070] Working: B
[24696.319205] Working Denied: A
[24696.403249] Working: B
[24696.423570] Working Denied: A
[24696.490090] Working Denied: C
[24696.503917] Working: B
[24696.523772] Working Denied: A
[24696.590680] Working Denied: C
[24696.605197] Working: B
[24696.624189] Working Denied: A
[24696.691120] Working Denied: C
[24696.705643] Working: B
```

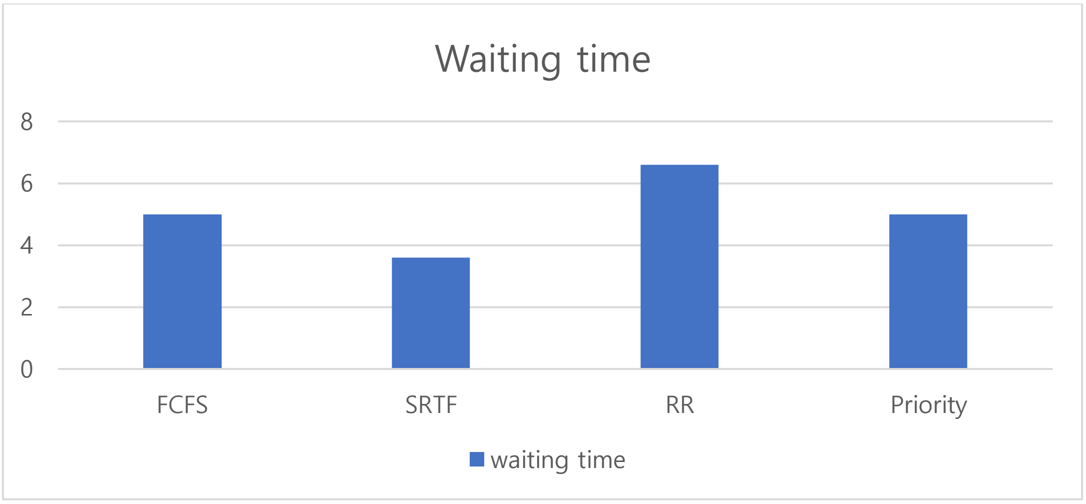


```
[24700.566505] Process Finished: B
[24700.623187] Working Denied: C
[24700.668964] Working: A
[24700.723736] Working Denied: C
[24700.769933] Working: A
[24700.828499] Working Denied: C
[24700.882333] Working: A
[24700.929379] Working Denied: C
[24700.988754] Working: A
[24701.030791] Working Denied: C
[24701.089887] Working: A
[24701.131182] Working Denied: C
[24701.194630] Working: A
[24701.232260] Working Denied: C
[24701.298433] Working: A
[24701.333593] Working Denied: C
[24701.400925] Working: A
[24701.434291] Working Denied: C
[24701.504664] Working: A
[24701.535253] Working Denied: C
[24701.615895] Working: A
[24701.635377] Working Denied: C
[24701.716552] Working: A
[24701.736057] Working Denied: C
[24701.818743] Working: A
[24701.836679] Working Denied: C
[24701.923786] Working: A
[24701.936961] Working Denied: C
[24702.029797] Working: A
[24702.038017] Working Denied: C
[24702.134278] Working: A
[24702.142177] Working Denied: C
[24702.235147] Working: A
[24702.244416] Working Denied: C
[24702.336213] Working: A
[24702.346194] Working Denied: C
[24702.439080] Working: A
[24702.446777] Working Denied: C
[24702.540663] Working: A
[24702.547949] Working Denied: C
[24702.644628] Working: A
[24702.648563] Working Denied: C
```

5. Experiment results and analysis

Comparing the average wait time by policy, the average wait time of FCFS was 5s, the average wait time of SRTF was 3.6s, the average wait time of RR was 6.6s, and the average wait time of priority was 5s. Therefore, it can be seen that the average wait time of SRTF is the smallest, and the overall wait time can be reduced by using the policy.





6. Difficulties and efforts

At first, I struggled with implementing the RR scheduling in a user program for loop but after that, I understand how to schedule RR scheduling in kernel. And I progressed through the assignment, I gained a clear understanding of how system calls work and the intricacies of scheduling. Initially, I was unsure how to calculate the response time and considered using a real-time CPU clock. But I realized that this could be resolved by setting a variable to check when the CPU was first allocated. It was my first assignment that took me over 30 hours. So I really wanted to give up, but after I finished it, I felt proud.