Report

姓名：周敦翔

學號：B07902050

Design

main.c

In charge of handling input and calling the function `scheduling` with different scheduling policy.

scheduling.c

- next_process Find next candidate child process according to scheduling policy `FIFO`, `SJF`, and `PSJF`, by scanning through the process array. Return the index of the selected process. If no process is ready, return -1.
- scheduling Determine which process to execute next by calling `next_process`. Create, block and wake up a child process through function `create`, `block`, `wakeup` defined in `process.c`.
- rr_scheduling In order to achieve round-robin scheduling, an additional ready queue is implemented to find the next candidate process.

process.c

- unit_time Define a time unit.
- assign_cpu Designate the process to specific CPU by `sched_setaffinity` given the pid of the process and the core specified.
- create Use `fork` to create a child process which runs X units of time.
- block Stop a process from executing by setting the scheduling policy to `SCHED_IDLE`.
- wakeup Continue executing a process by setting the scheduling policy to `SCHED_OTHER`.

System Calls

The macro `SYSCALL_DEFINEx` is used to define new system calls.

The function `getnstimeofday` is no longer available in Linux kernel 5.6.7. Related new functions are `ktime_get_real` and `ktime_to_timespec64`.

Environment

- Virtual Machine: VMware Workstation 15 Player
- OS: Ubuntu-20.04
- Kernel Version: Linux 5.6.7

Result

Below is a comparison between theoretical value and real time result of four different scheduling policies.

```
FIFO_3.txt:
```

```
    Process P1:
        theory:     start at 0, end at 8000
        my_result:  start at 0, end at 13170

    Process P2:
        theory:     start at 8000, end at 13000
        my_result:  start at 13287, end at 21396

    Process P3:
        theory:     start at 13000, end at 16000
        my_result:  start at 21571, end at 26464

    Process P4:
        theory:     start at 16000, end at 17000
        my_result:  start at 26529, end at 28137

    Process P5:
        theory:     start at 17000, end at 18000
        my_result:  start at 28159, end at 29754

    Process P6:
        theory:     start at 18000, end at 19000
        my_result:  start at 29796, end at 31352

    Process P7:
        theory:     start at 19000, end at 23000
        my_result:  start at 31359, end at 37957

    * Average run time difference of FIFO_3 = 2075.5714285714284 units


RR_2.txt:
    Process P1:
        theory:     start at 600, end at 8100
        my_result:  start at 600, end at 13980

    Process P2:
        theory:     start at 1100, end at 9600
        my_result:  start at 1414, end at 17078

    * Average run time difference of RR_2 = 6522.0 units


SJF_2.txt
    Process P1:
        theory:    start at 100, end at 200
        my_result: start at 100, end at 277
    Process P2:
        theory:    start at 400, end at 4400
        my_result: start at 639, end at 6698
    Process P3:
        theory:    start at 200, end at 400
        my_result: start at 281, end at 632
    Process P4:
        theory:    start at 4400, end at 8400
        my_result: start at 6698, end at 13159
    Process P5:
        theory:    start at 8400,  end at 15400
```

```
        my_result: start at 13159, end at 24518

    *Average run time difference of SJF_2 = 1821.4 units


PSJF_5.txt:
    Process P1:
        theory:     start at 100, end at 200
        my_result:  start at 100, end at 269

    Process P2:
        theory:     start at 400, end at 4400
        my_result:  start at 610, end at 6937

    Process P3:
        theory:     start at 200, end at 400
        my_result:  start at 272, end at 610

    Process P4:
        theory:     start at 4400, end at 8400
        my_result:  start at 7087, end at 14317

    Process P5:
        theory:     start at 8400, end at 15400
        my_result:  start at 14317, end at 27295

    * Average run time difference of PSJF_5 = 2348.4 units
```

As we can see that in all scenarios the run-time result takes longer than theoretical prediction. It may due to the overhead of context switch between different processes, or the execution of the scheduler, both of which lead to a higher CPU usage and thus slow down the overall progress.