Assignment 4

> *11911839 聂雨荷*

Q1

TIME	HRRN	FIFO/FCFS	RR	SJF	PRIORITY
1	A	A	A	A	A
2	A	A	A	A	B
3	A	A	B	A	A
4	A	A	A	A	D
5	B	B	D	B	D
6	D	D	A	D	C
7	D	D	C	D	C
8	C	C	D	C	C
9	C	C	C	C	A
10	C	C	C	C	A
Avg. Turn-around Time	4.5	4.5	4.75	4.5	4.25

Q2

[**Step1**]: Implement a syscall to let user process set `labschedule_good`.

```
// libs/unistd.h
#define SYS_labschedule_set_good 254
```

```
/* In User Mode*/
// user/libs/ulib.h
void labschedule_set_good(int good);

// user/libs/ulib.c
void set_good(int good){
```

```
    sys_labschedule_set_good(good);
}

// user/libs/syscall.h
int sys_labschedule_set_good(int32_t good);

// user/libs/syscall.c
int sys_labschedule_set_good(int32_t good){
    return syscall(SYS_labschedule_set_good, good);
}
```

```
/* In Kernel Mode*/
// kern/syscall/syscall.c
static int
sys_labschedule_set_good(uint32_t arg[]){
    int val = (int)arg[0];
    labschedule_set_good(val);
    return 0;
}

static int (*syscalls[])(uint64_t arg[]) = {
    [SYS_exit]                      sys_exit,
    [SYS_fork]                      sys_fork,
    [SYS_wait]                      sys_wait,
    [SYS_exec]                      sys_exec,
    [SYS_yield]                     sys_yield,
    [SYS_kill]                      sys_kill,
    [SYS_getpid]                    sys_getpid,
    [SYS_putc]                      sys_putc,
    [SYS_gettime]                   sys_gettime,
    [SYS_labschedule_set_good]      sys_labschedule_set_good, //
add!
};

// kern/process/proc.h
void labschedule_set_good(uint32_t good);

// kern/process/proc.c
void labschedule_set_good(uint32_t good){
    current->labschedule_good = good;
```

```
}
```

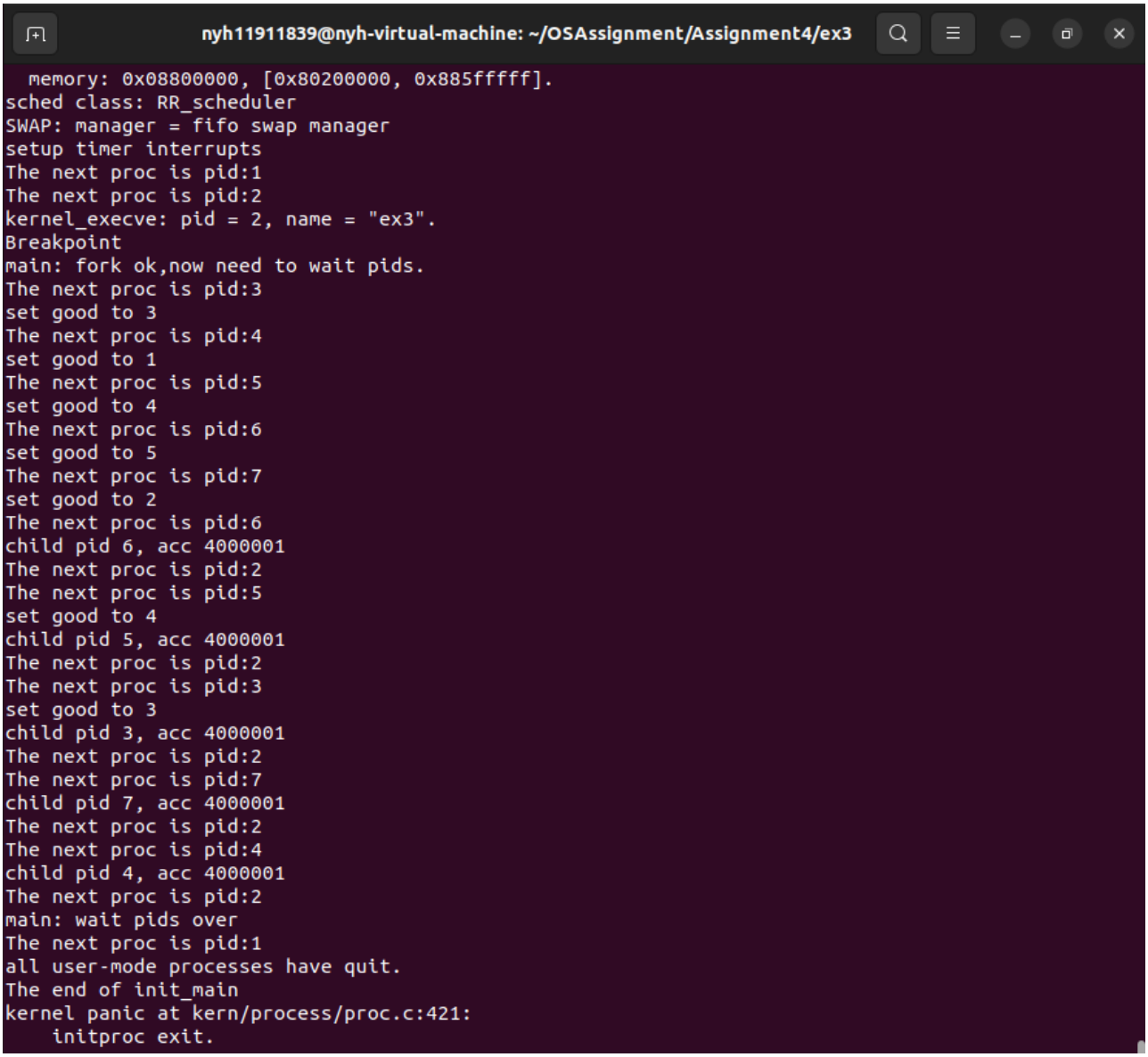
[**Step2**]: Choose the process with the largest good value to run when scheduling.

```
// kern/schedule/default_sched.c
static struct proc_struct *
RR_pick_next(struct run_queue *rq) {
    struct proc_struct *p_max = NULL;
    list_entry_t *le = list_next(&(rq->run_list));
    int32_t max_good = -1;

    if(le == &(rq->run_list)){
        return NULL;
    }

    // scan the whole queue, return the proc which contains the
largest good value
    while(le != &(rq->run_list)){
        struct proc_struct *p = le2proc(le, run_link);
        if((int32_t)p->labschedule_good > max_good){
            p_max = p;
            max_good = (int32_t)p->labschedule_good;
        }
        le = list_next(le);
    }
    return p_max;
}
```

[**Optional**] In order to reproduce the execution result as the example, we need to modify the `timebase`.

```
// kern/driver/clock.c
static uint64_t timebase = 400000;
```

The execution result is shown as:


nyh11911839@nyh-virtual-machine: ~/OSAssignment/Assignment4/ex3
memory: 0x08800000, [0x80200000, 0x885fffff].
sched class: RR_scheduler
SWAP: manager = fifo swap manager
setup timer interrupts
The next proc is pid:1
The next proc is pid:2
kernel_execve: pid = 2, name = "ex3".
Breakpoint
main: fork ok,now need to wait pids.
The next proc is pid:3
set good to 3
The next proc is pid:4
set good to 1
The next proc is pid:5
set good to 4
The next proc is pid:6
set good to 5
The next proc is pid:7
set good to 2
The next proc is pid:6
child pid 6, acc 4000001
The next proc is pid:2
The next proc is pid:5
set good to 4
child pid 5, acc 4000001
The next proc is pid:2
The next proc is pid:3
set good to 3
child pid 3, acc 4000001
The next proc is pid:2
The next proc is pid:7
child pid 7, acc 4000001
The next proc is pid:2
The next proc is pid:4
child pid 4, acc 4000001
The next proc is pid:2
main: wait pids over
The next proc is pid:1
all user-mode processes have quit.
The end of init_main
kernel panic at kern/process/proc.c:421:
initproc exit.