SP25 CSE2431 HOMEWORK 1
SOLUTION

Instructor: Luan Duong, Ph.D.,
Electronic Submission: 11:59 pm, Monday, Feb 03, 2025
Point: 25 points (5% of total grade)

Instructions:

This homework will be submitted **online**. Student will do their homework on paper/word processor and scan/take photos/convert to **pdf file** and submit on Carmen on or before the due date. **Only pdf** will be accepted.

Question 1 (3 points): **What is the output of the following program? List ALL the possible output(s).**

```
int num = 99;

int main(int argc, char**argv){
    printf("start num is %d\n", num);
    num++;
    int rc = fork();
    if(rc>0){
        num+=10;
        printf("Parent's num is %d\n", num);
    }
    else if(rc==0){
        num+=30;
        printf("Child's num is %d\n", num);
    }
    num--;
    printf("end num is %d\n", num);
}
```

ANSWER:

<u>1st possible output:</u>

```
start num is 99
Parent's num is 110
end num is 109
Child's num is 130
end num is 129
```

```
2nd possible output:
start num is 99
Child's num is 130
end num is 129
Parent's num is 110
end num is 109
```

Question 2 (10 points): **Which of the following scheduling algorithms could result in starvation? For those algorithms that might result in starvation, describe a situation in which starvation is likely to occur? (2 points)**

1. First-come, first-served (FCFS)
2. Shortest Job First (SJF)
3. Round Robin
4. Basic MLFQ with first 4 rules.

Given the following mix of job, job lengths, and arrival times, assume a time slice of 10 seconds and compute the completion time for each job, average turnaround time, and average response time for the FIFO, RR, and STCF algorithms. (For FIFO, the order of job is the order of scheduling) (8 points)

Job	Length (s)	Arrival Time
J1	85	0
J2	30	10
J3	35	10
J4	20	80
J5	50	85

A note: There might be different scheduling order, you only need to give one solution scheduling for each of the algorithms.

ANSWER:

+++ Among the listed scheduling algorithms, starvation could happen for:

1. **Shortest Job First (SJF):** Although Shortest Job First is a non-preemptive scheduling, there is still a possibility that starvation can happen. Even though a long process may be ready to get scheduled, if short processes are continually added, this long process may never been scheduled. A simple example: although its arrival time is at 0, J2 will not have a chance to run, since shorter-time job such as J3, J4, J5, … keep coming

and flushing the processor. Another example is in slide 20 of the lecture note.

Job	Length (s)	Arrival time
J1	10	0
J2	100	0
J3	10	10
J4	10	10
J5	10	20
…	10	30

2. **<u>Basic MLFQ with first 4 rules:</u>** In multi-level feedback queue with first 4 rules, if we do not have a priority reset after some period of time, short jobs keep coming and long jobs (with low priority) has to wait and could not be executed.

+++ Here I calculate the Response time = $T_{first-run} - T_{arrival}$ (The time that the process arrived till the time that the process got first scheduled)

+++ One schedule result in:

			Scheduling Algorithms					
			FIFO		RR		SRTF/STCF	
Job	Length (s)	Arrival time	**Completion time**	**Response time**	Completion Time	Response Time	Completion Time	Response Time
J1	85	0	85	0	220	0	220	0
J2	30	10	115	(85-10)	80	(10-10)	40	(10-10)
J3	35	10	150	(115-10)	125	(20-10)	75	(40-10)
J4	20	80	170	(150-80)	145	(100-80)	100	(80-80)
J5	50	85	220	(170-85)	215	(110-85)	150	(100-85)
Average turn-around time			111s		120s		80s	
Average response time				67s		11s		9s

$$\boldsymbol{Avg\ T^{FIFO}_{turn-around}} = \frac{(85-0)+(115-10)+(150-10)+(170-80)+(220-85)}{5} = \mathbf{111s}$$

$$\boldsymbol{Avg\ T^{FIFO}_{response}} = \frac{(0-0)+(85-10)+(115-10)+(150-80)+(170-85)}{5} = \mathbf{67s}$$

$$\boldsymbol{Avg\ T^{RR}_{turn-around}} = \frac{(220-0)+(80-10)+(125-10)+(145-80)+(215-85)}{5} = \mathbf{120s}$$

$$\boldsymbol{Avg\ T^{RR}_{response}} = \frac{(0-0)+(10-10)+(20-10)+(100-80)+(110-85)}{5} = \mathbf{11s}$$

$$\boldsymbol{Avg\ T^{SRTF/STCF}_{turn-around}} = \frac{(220-0)+(40-10)+(75-10)+(100-80)+(150-85)}{5} = \mathbf{80s}$$

$$\boldsymbol{Avg\ T^{SRTF/STCF}_{response}} = \frac{(0-0)+(10-10)+(40-10)+(80-80)+(100-85)}{5} = \mathbf{9s}$$

Schedule for RR (Time slice 10s) (1 possible scheduling)

Time	Job
0	1
10	2
20	3
30	1
40	2
50	3
60	1
70	2

80	3 → J2 completes
90	1
100	4
110	5
120	3
125	1 → J3 completes
135	4
145	5 → J4 completes
155	1
165	5
175	1
185	5
195	1
205	5
215	1 → Job 5 completes
220	J1 completes

Queue right before 80 (70-80): J3 J1 (J2 being executed then complete at t=80)

Queue at 80: J1 J4 (J4 arrives at 80, J3 is being execute)

Queue at 85: J1 J4 J5

Queue at 90: J4 J5 J3 (J1 is executing)

***** Note: At t=10, there is another possiblity that: J1 finished its timeslice, and then placed ahead of the queue compared to J2 and J3 which arrived at t=10. So queue would be: J1 - J2 - J3 and thus J1 would be executed again!**

<u>Schedule for STCF/SRTF</u>

Time	Job
0	1
10	2
20	2
30	2
40	3 – J2 completes
50	3
60	3
75	1 → J3 completes
80	4
90	4

100	5 → J4 completes
110	5
120	5
130	5
140	5
150	1
160	1
170	1
180	1
190	1
200	1
210	1
220	J1 completes

Question 3 (8 points): **Analyze scheduling algorithms for the following five processes given each process' length and arrival time.**

Processes	Length (s)	Arrival Time
P1	16	0
P2	7	2
P3	2	4
P4	4	6
P5	22	8

For each of the following scheduling algorithms draw a diagram about how processes are scheduled, **calculate** the turnaround time for **each process** and calculate the **average turnaround time**.

1) First In First Out (FIFO) [2 points]
2) Shortest Job First (SJF) (Assume OS first schedules at time 0 and then schedules whenever a process terminates.) [2 points]
3) Shortest Time to Completion First (STCF) [2 points]
4) Round Robin Scheduling (Assume that the time slice is 5 seconds. Assume that if a new process arrives as the time slice of the executing process expires the executing process is put at the end of the *current* waiting queue. Assume that if a process terminates before its current time slice expires, the next process will immediately get a new and full slice. E.g. if a process starts at 10 and has only 2 remaining jobs, the next process will start at 12 and continue till 12+5. [2 points]

ANSWER:

1) **First In First Out (FIFO)**

P1 [0 16] P2 [16 23] P3 [23 25] P4 [25 29] P5 [29 51]

Turnaround time for each process:

$T_{P_1} = 16 - 0 = 16(s)$

$T_{P_2} = 23 - 2 = 21\ (s)$

$T_{P_3} = 25 - 4 = 21\ (s)$

$T_{P_4} = 29 - 6 = 23\ (s)$

$T_{P_5} = 51 - 8 = 43\ (s)$

Average turnaround time = [(16-0) + (23-2) + (25–4) + (29-6) + (51-8)]/5 = 24.8

2) **Shortest Job First (SJF)** (Assume OS first schedules at time 0 and then schedules whenever a process terminates.)

P1 [0 16] P3 [16 18] P4 [18 22] P2 [22 29] P5 [29 51]

Turnaround time for each process:

$T_{P_1} = 16 - 0 = 16(s)$

$T_{P_2} = 29 - 2 = 27\ (s)$

$T_{P_3} = 18 - 4 = 14\ (s)$

$T_{P_4} = 22 - 6 = 16\ (s)$

$T_{P_5} = 51 - 8 = 43\ (s)$

Average turnaround time = [(16-0) + (18-4) + (22–6) + (29-2) + (51-8)]/5 = 23.2

3) **Shortest Time to Completion First (STCF)**

P1 [0 2] P2[2 4] P3 [4 6] P4 [6 10] P2 [10 15] P1 [15 29] P5 [29 51]

Turnaround time for each process:

$T_{P_1} = 29 - 0 = 29(s)$

$T_{P_2} = 15 - 2 = 13\ (s)$

$T_{P_3} = 6 - 4 = 2\ (s)$

$T_{P_4} = 10 - 6 = 4\ (s)$

$T_{P_5} = 51 - 8 = 43\ (s)$

Average turnaround time = [(29-0) + (15-2) + (6–4) + (10-6) + (51-8)]/5 = 18.2

4) **Round Robin Scheduling** (Assume that the time quantum is 5 time units. Assume that if a new process arrives as the time slice of the executing process expires the executing process is put at the end of the *current* waiting queue. Assume that if a process terminates before its current time slice expires, the next process will immediately get a new and full slice. E.g. if a process starts at 10 and has only 2 remaining work, the next process will start at 12 and continue till 12+5.)
P1 [0 5] P2 [5 10] P3 [10 12] P1 [12 17] P4 [17 21] P5 [21 26] P2 [26 28] P1 [28 33] P5 [33 38] P1 [38 39] P5 [39 44] P5 [44 49] P5 [49 51]

Turnaround time for each process:
$T_{P_1} = 39 - 0 = 39(s)$
$T_{P_2} = 28 - 2 = 26\ (s)$
$T_{P_3} = 12 - 4 = 8\ (s)$
$T_{P_4} = 21 - 6 = 15\ (s)$
$T_{P_5} = 51 - 8 = 43\ (s)$

Average turnaround time = [(39-0) + (28-2) + (12–4) + (21-6) + (51-8)]/5 = 26.2

Question 4 (4 points): **Fix memory-related bugs in the following programs (one program may have more than one bugs). You can try to run it with valgrind, but remember that valgrind is not a panacea.**

```
int *add(int *a, int *b){
    int *ret = malloc(sizeof(int));
    if(a==NULL || b==NULL)
        return NULL;
    *ret = *a + *b;
    return ret;
}

int main(int argc, char**argv){
    int a = 3;
    int b = 4;
    int * ret = add(&a, &b);
    if(ret==NULL)
        printf("Error\n");
    else
        printf("3+4=%d\n", *ret);
}
```

ANSWER:

```
int *add(int *a, int *b){
    int *ret = malloc(sizeof(int));
  if(a==NULL || b==NULL) {
      free(ret);
      //or move the if check before malloc
        return NULL; }
    *ret = *a + *b;
    return ret;
}

int main(int argc, char**argv){
    int a = 3;
    int b = 4;
    int * ret = add(&a, &b);
    if(ret==NULL)
        printf("Error\n");
    else{
        printf("3+4=%d\n", *ret);
        free(ret);
     }
}
```