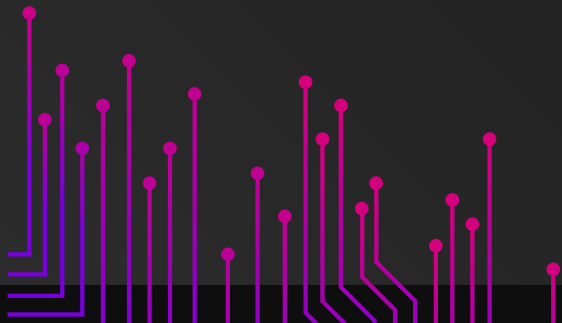
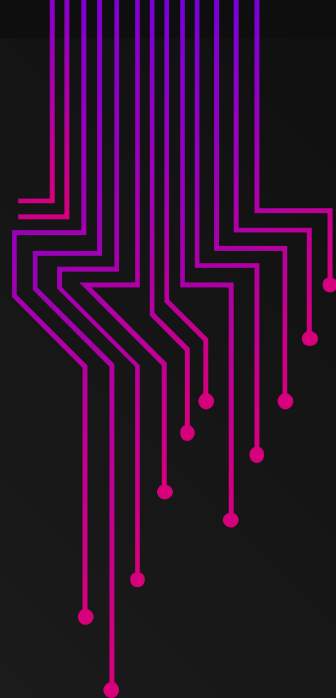


Operating System 112 Fall

Homework 2 - CPU Scheduling

Prof. 蔡文錦

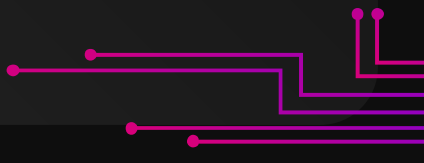
TA 林浩君, 薛乃仁, 周彥昀, 許承壹





Objective

In this homework, we are going to learn how CPU schedules processes and implement some classic scheduling algorithm by yourselves.

1. Learn how to evaluate performance of different scheduling algorithm.
 2. Having an better recognition of what context switch is.
- 

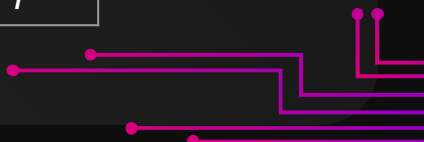


Input

The first line of input contains **N** , **M** , the amount of level in multilevel feedback queue and the number of process.

The next **N** lines(from highest priority queue to lowest priority queue) each contain ***mode_i***, ***time_quantum_i***, the algorithm of the *i*th-queue.

mode	time quantum	algorithm
0	-1	First-Come, First-Served
1	-1	Shortest Remaining Time First
2	<i>time_quantum_i</i>	Round Robin with time quantum is <i>time_quantum_i</i>





Input

The next M lines each contain $arrival_time_j$, $bust_time_j$, the arrival time, bust time of the j th-process p_j . $arrival_time_j < arrival_time_k$ when $j < k$.

Constraints:

$$1 \leq N \leq 5$$

$$1 \leq M \leq 100$$

$$0 \leq arrival_time_j \leq 10000$$

$$1 \leq bust_time_j \leq 1000$$

If $mode \neq 2$, $time_quantum_i = -1$

Else $1 \leq time_quantum_i \leq 100$





Output

The output will begin with M lines, each containing two nonnegative integer, the wait time, turnaround time of the i th-process p_i .

The next line will contain a nonnegative integer, total wait time.

The final line will contain a nonnegative integer, total turnaround time.





Process Priority

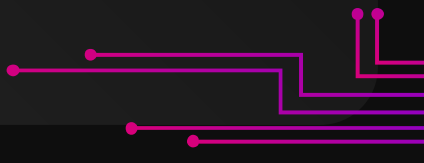
queue_time: The time when process push into queue (e.g. new arrive process, preempted process)

remaining_time: The remain time of process.

[1]: In RR, when two process have same ***queue_time***, select the process with **larger *arrival_time***.

[2]: In SRTF, when two process have same ***remaining_time***, select the process with **smaller *arrival_time***.

[3]: In multi-level queue, the process in **higher priority queue** can preempt the running process in **lower priority queue**.



Part 1: First Come, First Serve (25%)

Amount of test cases: 5 (5% for each)

Test case 1:

Input:

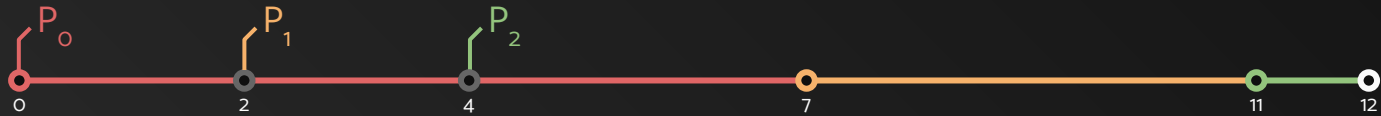
1 3	# 1 queue, 3 processes
0 -1	# queue 0: FCFS
0 7	# process 0: arr=0, bust=7
2 4	# process 1: arr=2, bust=4
4 1	# process 2: arr=4, bust=1

Output:

0 7	# process 0: wait=0, turnaround=7
5 9	# process 1: wait=5, turnaround=9
7 8	# process 2: wait=7, turnaround=8
12	# total waiting time
24	# total turnaround time

Part 1: First Come, First Serve (25%)

Incoming



FCFS

P_1

P_1
 P_2

P_2

High Priority

Low Priority

Part 2: Shortest Remaining Time First (25%)

Amount of test cases: 5 (5% for each)

⌘ Preempted by process with **smaller** burst time.

Test case 1:

Input:

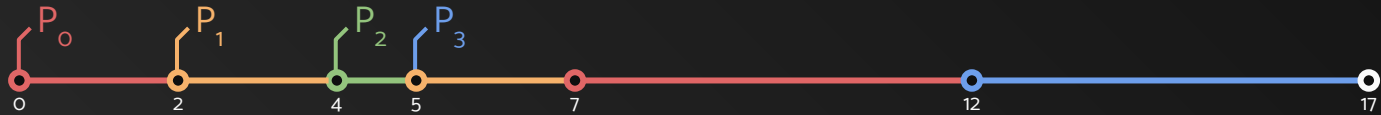
1 4	# 1 queue, 4 processes
1 -1	# queue 0: SRTF
0 7	# process 0: arr=0, burst=7
2 4	# process 1: arr=2, burst=4
4 1	# process 2: arr=4, burst=1
5 5	# process 3: arr=5, burst=5

Output:

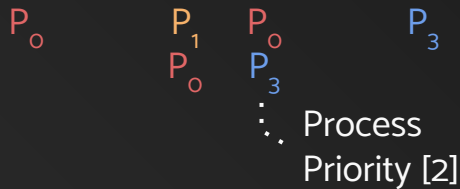
5 12	# process 0: wait=9, turnaround=16
1 5	# process 1: wait=1, turnaround=5
0 1	# process 2: wait=0, turnaround=1
7 12	# process 3: wait=2, turnaround=6
13	# total waiting time
30	# total turnaround time

Part 2: Shortest Remaining Time First (25%)

Incoming



SRTF



High Priority

Low Priority

Part 3: Round Robin (25%)

Amount of test cases: 5 (5% for each)

Test case 1:

Input:

1 4	# 1 queue, 4 processes
2 4	# queue 0: RR(4)
0 7	# process 0: arr=0, bust=7
2 4	# process 1: arr=2, bust=4
4 1	# process 2: arr=4, bust=1
5 5	# process 3: arr=5, bust=4

Output:

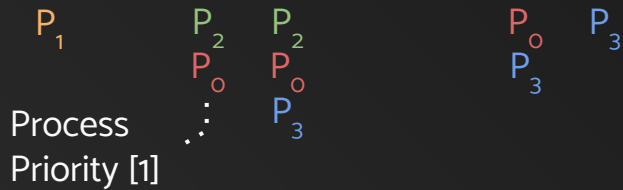
5 12	# process 0: wait=5, turnaround=12
2 6	# process 1: wait=2, turnaround=6
4 5	# process 2: wait=4, turnaround=5
7 12	# process 3: wait=7, turnaround=11
18	# total waiting time
35	# total turnaround time

Part 3: Round Robin (25%)

Incoming



RR(4)



High Priority

Low Priority

Part 4: RR + SRTF (10%)

Amount of test cases: 5 (2% for each)

✂ If the process is not finish (preempted or time quantum exceeded), push it to the next level priority queue (if available).

Test case 1:

Input:

2 4	# 2 queue, 4 processes
2 3	# queue 0: RR(3)
1 -1	# queue 1: SRTF
0 7	# process 0: arr=0, bust=7
2 4	# process 1: arr=2, bust=4
8 1	# process 2: arr=8, bust=1

Output:

5 12	# process 0: wait=5, turnaround=12
1 5	# process 1: wait=1, turnaround=5
0 1	# process 2: wait=0, turnaround=1
6	# total waiting time
18	# total turnaround time

Part 4: RR + SRTF (10%)

Incoming



RR(3)

P_1

High Priority

Part 4 ✕

SRTF

P_0

P_0

P_0

Process
Priority [3]

Low Priority

Part 5: Multilevel Queue (25%)

Amount of test cases: 5 (5% for each)

✂ Same as Part 4.

Test case 1:

Input:

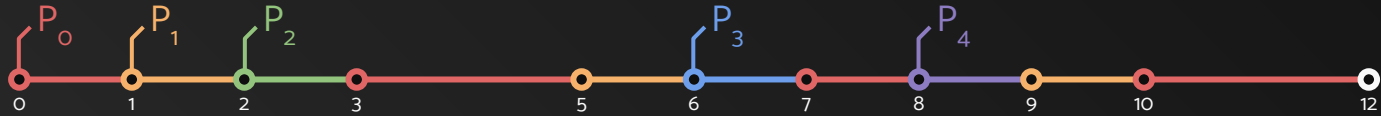
3 5	# 3 queue, 4 processes
1 -1	# queue 0: SRTF
2 2	# queue 1: RR(2)
0 -1	# queue 2: FCFS
0 5	# process 0: arr=0, bust=5
1 3	# process 1: arr=1, bust=3
2 1	# process 2: arr=2, bust=1
6 1	# process 3: arr=6, bust=1
8 1	# process 4: arr=8, bust=1

Output:

6 11	# process 0: wait=6, turnaround=12
6 9	# process 1: wait=6, turnaround=9
0 1	# process 2: wait=0, turnaround=1
0 1	# process 3: wait=0, turnaround=1
0 1	# process 4: wait=0, turnaround=1
12	# total waiting time
23	# total turnaround time

Part 5: Multilevel Queue (25%)

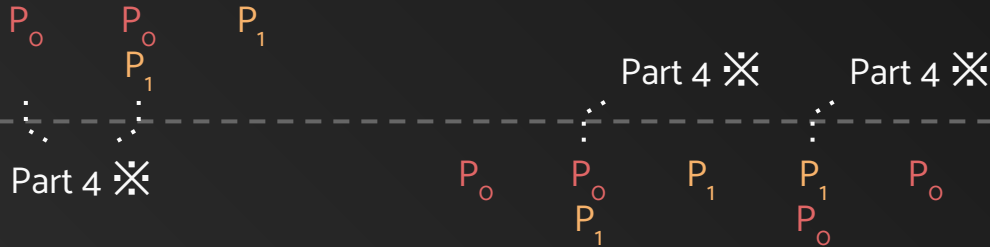
Incoming



SRTF

RR(2)

FCFS



High Priority

Low Priority



Submission and Rules

Submission:

1. You should write your code in C/C++
2. Please upload your homework in such format:
 - hw2_studentID.cpp (e.g. hw2_312551014.cpp)

Rule:

- No **plagiarism** is allowed, since the grade of this course is critical for **graduate program application in CS related field**, we will not pardon such behavior at all, so please be responsible to yourself. You can discuss with your classmates, but don't copy and paste.
 - Incorrect filename / file format will get -10% point.
 - Delayed submission will get -20% point per day.
- 