FinalExam - Ver A

● Graded

Student

PO-YUN) 鄭博允 (CHENG

Total Points

83 / 106 pts

Question 1

1 **12** / 12 pts

1.1 **1a** **2** / 2 pts

✔ **+ 2 pts** Correct

1.2 **1b** **2** / 2 pts

✔ **– 0 pts** 正確

1.3 **1c** **2** / 2 pts

✔ **– 0 pts** 正確

1.4 **1d** **2** / 2 pts

✔ **– 0 pts** 正確

1.5 **1e** **4** / 4 pts

✔ **+ 2 pts** Sequential access

✔ **+ 2 pts** Direct access

Question 2

2 **16** / 20 pts

2.1 **2a** **2** / 4 pts

✔ **– 2 pts** minor mistake

2.2 **2b** **2** / 4 pts

✔ **– 2 pts** difference between deadlock and starvation is not clearly elaborated

elaborate the difference, not solving methods

2.3 **2c** **4** / 4 pts

✔ **– 0 pts** correct answer.

2.4 **2d** **4** / 4 pts

✔ **– 0 pts** correct

2.5 **2e** **4** / 4 pts

✔ **– 0 pts** correct answer

Question 3

3 **16** / 20 pts

3.1 **3a** **6** / 6 pts

- ✔ **– 0 pts** 正確

3.2 **3b** **4** / 6 pts

- ✔ **– 1 pt** RR P1 incorrect
- ✔ **– 1 pt** RR P2 incorrect

3.3 **3c** **6** / 8 pts

- ✔ **– 2 pts** 1 error

Question 4

4 **16** / 18 pts

4.1 **4a** **12** / 12 pts

✔ **– 0 pts** All scheduling correct. (You will get points if your calculation process is correct.)

4.2 **4b** **0** / 2 pts

✔ **– 2 pts** Not fully correct

4.3 **4c** **4** / 4 pts

✔ **– 0 pts** correct

Question 5

5 **13** / 20 pts

5.1 5a **6** / 6 pts

✔ **– 0 pts** 正確

5.2 5b **5** / 7 pts

✔ **– 2 pts** First loop wrong (Line11-16: only Pi not IDLE)

5.3 5c **2** / 7 pts

✔ **– 5 pts** Description Wrong

1 Not detail enoug

Question 6

6 **10** / 16 pts

6.1 6a **0** / 6 pts

✔ **– 4 pts** No conditions or the condtions is wrong.

✔ **– 2 pts** No outcomes or the outsome is wrong.

6.2 6b **6** / 6 pts

✔ **– 0 pts** Correct answer
(1) Original (miss deadline and continue): 6, 22, 17, 24
(2) Miss deadline and suspend: 6, 18 or X, 17, 20
(3) No preemption: 7, 12, 22, 24

6.3 6c **4** / 4 pts

✔ **– 0 pts** Correct

CSIE 3310, Spring 2024: Final Exam
National Taiwan University

Name	鄧	博	允			SID	b	1	1	9	0	2	0	3	8

Instructions:

- Food and drinks are PROHIBITED. Leave them outside.
- Do NOT use the pencil to answer.
- Write your name and student ID in the above boxes. One character/digit in each box.
- Write your answers in the rectangle below each question or the specific areas.
- Provide sufficient explanations or assumptions, if any.
- Ask permission to use the restroom and leave your exam sheets and phone on the seat.
- Stay seated in the last 5 minutes before the exam ends: restroom and early turning-in are NOT allowed.
- When the exam ends, STOP WRITING and stay seated until the TAs collect the exam sheets in your seating region.

Wait!!

Do NOT turn the page until you are told to start.

Ver. A

There are 12 pages in total, not including cover page: 6 questions on 8 pages and 4 blank pages for scratch. Raise your hand NOW if not.

1 Term Definitions (12 pts)

(a) (2 pts) Session Semantics in File Systems

Answer:

Changes on open file will be see by other processes after the file is closed.

(b) (2 pts) Stateless File Servers

Answer:

每個操作 (eg. read, write) 都是獨立的

(c) (2 pts) What is the purpose of the **wear leveling** on NVM devices?

Answer:

為了讓每個 block (pages) 的使用量平衡，不會有某些部份被過度使用或沒被使用 (主要是因為 NVM 的壽命和讀寫次數有關

(d) (2 pts) What are CPU-bound programs?

Answer:

I/O 行為遠少於 CPU 行為 (計算) 的程式

(e) (4 pts) The information in a file can be accessed in several ways. Please explain the sequential access method and the direct access method [2 pts for each].

Answer:

sequential access: 連續存取一連串相鄰的 blocks

direct access: 直接存取某一個 blocks

2 Short Answers (20 pts)

(a) (4 pts) Please describe how the log-structured file systems avoid the failure of consistency checking when the system crashes.

Answer:

由於 log-structured file system 會用 log 來記錄 file 的 metadata，
在 crash 的時候可以透過 log 來回復

(b) (4 pts) Please define *deadlock* and *starvation* for process synchronization, and elaborate the difference between them.

Answer:

deadlock 是兩個以上的 processes 互相等走對方所需的資源，導致沒有 process 可以繼續執行

starvation 則是部份優先度較低的 process 因為優先度高的 process 不斷佔用資源，導致它無法執行，可以透過 aging 解決

(c) (4 pts) Unified buffer cache can avoid *double caching* issue. Please elaborate the cause and two shortcomings out of three.

Answer:

double caching 是因為某些 程序 要 同時 經過 (淺) buffer cache 和 (深) page cache，造成存取 效率 減低、記憶體使用量增加

→ page cache → buffer cache → hardware (memory)

→ unified buffer → hardware (memory)

(d) (4 pts) Elaborate at least two characteristics of *asynchronous* and *synchronous* write operations in file systems, respectively. (Two for each.)

Answer:

asynchronous write 可以防止記憶體(檔案)的內容被同時更改，但會造成 busy waiting 等的中上的問題

synchronous write 讓多個 process 同時去更改檔案內容(可能透過 copy-on-write 等)，可以加快整體的效率，但可能會提高記憶體的使用量及 implementation 的複雜度

(e) (4 pts) Please explain the difference between preemptive and non-preemptive scheduling.

Answer:

preemptive scheduling 會根據 algorithm 去強制 對 process 做 context-switch

而 non-preemptive 則是由 process 主動做 context-switch

3 Scheduling (20 pts)

Considering the following processes, their arrival time and burst time are given as follows.

Process	Arrival Time	Burst Time
P_1	0.0	8
P_2	0.4	4
P_3	1.0	1

We are given four different CPU scheduling algorithms:

- First-come-First-Served (FCFS);
- Shortest-job-first (SJF);
- Shortest-remaining-job-first (SRJF);
- Round-Robin (quantum = 2).

(a) (6 pts) Please fill in the **turnaround time** of each process for the given scheduling algorithms in the following table. [Hint: Drawing Gantt charts is not necessary but may help you to perform the calculation.]

Answer:

Algorithm	P_1	P_2	P_3
Shortest-Job-First (SJF)	8	12.6	8
Shortest-Remaining-Job-First (SRJF)	13	5	1

(b) (6 pts) Please fill in the **waiting time** of each process for the given scheduling algorithms in the following table. [Hint: Drawing Gantt charts is not necessary but may help you to perform the calculation.]

Answer:

Algorithm	P_1	P_2	P_3
First-Come-First-Served (FCFS)	0	7.6	11
Round-Robin (RR) with quantum=2	0	4.6	3

(c) (8 pts) Please indicate if these statements are **True (T)** or **False (F)**.

Answer:

T	F	
☐	☑	The Shortest-Remaining-Job-First (SRJF) scheduling algorithm is non-preemptive.
☐	☑	First-Come-First-Served (FCFS) may incur convoy effect.
☑	☐	The Round-Robin scheduling algorithm is preemptive.
☐	☑	When a process is waiting to be assigned to a processor, it is in the waiting state.

4 Mass-Storage Systems (18 pts)

(a) (12 pts) Assume that a moving-head disk has 200 cylinders, numbered from 0 to 199. Initially, the disk head is at cylinder 140. Consider a FIFO queue with requests for I/O to blocks on cylinders in the following order: 85, 149, 90, 170, 148, 2. What are the total head movements of cylinders for each HDD scheduling algorithm to satisfy these requests?

- FCFS Scheduling;
- SCAN Scheduling (Assuming that the requests are scheduled when the disk arm is **moving toward 0** and that the initial head position is at cylinder 140);
- C-SCAN Scheduling (Assuming that the requests are scheduled when the disk arm is **moving from 0 to 199** and that the initial head position is at cylinder 140).

Answer:

FCFS: 140 → 85 → 149 → 90 → 170 → 148 → 2

55 + 64 + 59 + 80 + 22 + 146 = 426

假設 SCAN 和 C-SCAN 都會移到最前、後端

SCAN: 140 → 90 → 85 → 2 → 0 → 148 → 149 → 170

140 + 170 = 310

C-SCAN: 140 → 148 → 149 → 170 → 199 → 0 → 2 → 85 → 90

59 + 199 + 90 = 348

(b) (2 pts) Which of the following items are possible RAID organization to enhance reliability ~~and~~ or performance? [Hint: Multiple choices. Points are given only when the answer is fully correct. No partial points will be given.]

(1) Mirroring (2) Block interleaved parity (3) Hashing (4) Swapping (5) Caching

Answer:

1, 2, 5

(c) (4 pts) Please briefly explain RAID levels 0 + 1 and RAID levels 1 + 0.

Answer:

RAID 01 是指先做 disk 做 RAID 0，也就是兩張本一個 disk 的內容平均分散到兩個 disk 內；再做 RAID 1，也就是將兩個 disk 的內容 mirror 一份到另外兩個 disk 內

RAID 10 則相反：先做 RAID 1 再做 RAID 0

5 Synchronization Tools (20 pts)

The Eisenberg and McGuire algorithm is an algorithm for solving the critical sections problem, a general version of the dining philosophers problem. It was described in 1972 by Murray A. Eisenberg and Michael R. McGuire.

The software solution to the critical-section problem allows n processes with a lower bound on waiting of $n - 1$ turns. These n processes share the following variables:

```
enum pstate = {IDLE, WAITING, ACTIVE};
pstate flags[n];
int turn;
```

The following shows the Eisenberg and McGuire algorithm for process P_i.

```
int index;

// Enter section
do {
  /* Announce that P_i needs the resource */
  flags[i] = WAITING;

  /* scan processes from the one with the turn up to ourselves. */
  /* repeat if necessary until the scan finds all processes idle */
  index = turn;
  while (index != i) {
    if (flags[index] != IDLE)
    index = turn;
    else
    index = (index + 1) % n;
  }

  /* now tentatively claim the resource */
  flags[i] = ACTIVE;

  /* find the first active process besides ourselves, if any */
  index = 0;
  while ((index < n) && ((index == i) || (flags[index] != ACTIVE))) {
    index++;
  }

  /* if there were no other active processes, AND if we have the turn
  or else whoever has it is idle, then proceed.  Otherwise, repeat
  the whole sequence. */
```

```
} while (!((index >= n) && ((turn == i) || (flags[turn] == IDLE))));

// Critical Section
/* claim the turn and proceed */
turn = i;

// Exit section
/* find a process which is not IDLE */
/* (if there are no others, we will find ourselves) */
index = (turn + 1) % n;
while (flags[index] == IDLE) {
  index = (index + 1) % n;
}

/* give the turn to someone that needs it, or keep it */
turn = index;

/* we're finished now */
flags[i] = IDLE;

// Remainder Section
```

Please answer the following questions:

(a) (6 pts) Please list the three requirements for the critical-section problems.

Answer:

mutual exclusive：一次只有一個 process 可以進 critical section

living：只有不在 remainder section 的 process 可以參與決定下一個進 critical section 的 process

bound-waiting：process 不能無限期的等待

(b) (7 pts) Prove the Eisenberg and McGuire algorithm shown above meets the first requirement in your answer. Please indicate the requirement at the beginning of your answer.

Answer:

first requirment：mutual exclusive

要進入 critical section 只有當每個其它 process 都被 check 過(非 ACTIVE)而且

① turn == i (輪到自己) 或

② flags[turn] == IDLE (輪到的 process 在 IDLE) 的時候

~~由於 turn 只會被指定到除了自己之外非 IDLE 的 process 或自己~~

~~當 ① 滿足 ② 就一定不會滿足~~ 當 ① 滿足，其它 process 的 ①、② 都不會有

所以不可能同時有兩個 process 進入 critical section

(c) (7 pts) Prove the Eisenberg and McGuire algorithm shown above meets the second requirement in your answer. Please indicate the requirement at the beginning of your answer.

Answer:

second requirement : living .

由於 turn 的指定當在 critical section 和 exit section 內，

因此不會有在 remainder section 參與決定的問題

6 Priority Inversion (16 pts)

Predictability is essential for embedded real-time systems and is affected by the scheduler and synchronization mechanisms in the operating systems. Unfortunately, when the non-preemptive resources are shared among real-time tasks, the real-time scheduler may not be able to estimate the worst-case completion times although it knows how much time on each resource a real-time process will use.

Assume that there are four aperiodic real-time tasks in the system: τ_1, τ_2, τ_3, and τ_4 and they are scheduled by a fixed-priority preemptive scheduling algorithm. Each task has two types of executions: preemptible and non-preemptible critical-section parts. This critical section is protected by the mutex lock. Each task is defined by its arrival time t_a, the execution time of the first preemptible part t_p^1, the execution time of the non-preemptible part t_n, and the execution time of the second preemptible part t_p^2, and relative deadline d. These tasks are defined as follows. $\tau_1 = (t_a, t_p^1, t_n, t_p^2, d) = (3, 3, 0, 0, 5)$, $\tau_2 = (4, 1, 2, 2, 14)$, $\tau_3 = (5, 10, 0, 20)$, and $\tau_4 = (1, 1, 2, 30)$.

The priority level of these four tasks is inversely defined by their task IDs. Hence, task τ_1 has the highest priority level and task τ_4 has the lowest priority level.

(a) (6 pts) Please elaborate *priority inversion* in the real-time scheduling, including its conditions (4 pts) and outcomes (2 pts).

Answer:

- Conditions:

  若有 process 要要更高的優先度時，像是 deadline 快到了等

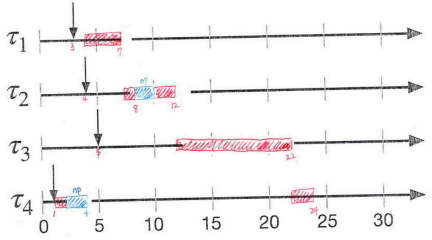
  就會產生 priority inversion

- Outcomes:

  processes 之間的優先度會調換

(b) (6 pts) Please (1) compute the completion time of each task and (2) draw the schedule of the four tasks, using the provided timelines. On the drawing, please indicate the execution of preemptive and non-preemptive parts. (You may use different types of lines or coloring.)

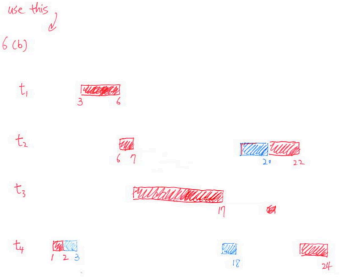
Answer: τ_1 6 τ_2 22 τ_3 17 τ_4 24



(c) (4 pts) What is the completion time of task τ_2 when the **priority inheritance protocol** (PIP) is applied? PIP boosts the priority level of a low-priority task when it blocks a high-priority task to the priority level of the blocked task, and drops the priority level to the original level when the blocking ends.

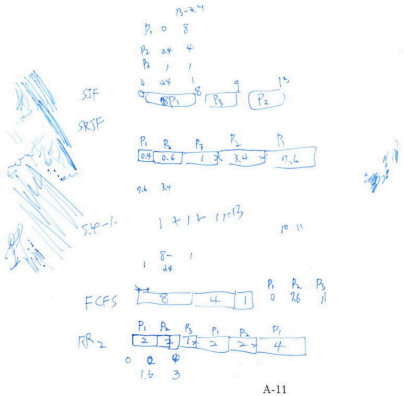
Answer: 12

This page is intentionally left blank for scratch and will NOT be graded.



This page is intentionally left blank for scratch and will NOT be graded.

This page is intentionally left blank for scratch and will NOT be graded.



This page is intentionally left blank for scratch and will NOT be graded.