Xi'an Jiaotong-Liverpool University
西交利物浦大學

PAPER CODE	EXAMINER	DEPARTMENT	TEL
CPT 104	Gabriela Mogos Dongyao Jia	Department of Computing	1515

2nd SEMESTER *2023/24 RESIT* EXAMINATION

Undergraduate Stage 2

OPERATING SYSTEMS CONCEPTS

TIME ALLOWED: 2 Hours

INSTRUCTIONS TO CANDIDATES

1. This is an open-book examination.
2. Total marks available are 100, accounting for 100% of the overall module marks.
3. Answer all FOUR questions.
4. The number in the column on the right indicates the marks for each question.
5. Relevant and clear steps should be included in the answers.
6. The university approved calculator - Casio FS82ES/83ES can be used.
7. Only English solutions are accepted.
8. All materials must be returned to the exam invigilator upon completion of the exam. Failure to do so will be deemed academic misconduct and will be dealt with accordingly.

QUESTION I. Fundamentals (39 marks)

1. For a given class, the student records are stored in a file. The records are randomly accessed and updated. Assume that each student's record is of fixed size. Which of the three allocation schemes (contiguous, linked and indexed) will be most appropriate? Explain your answer.

(9 marks)

2. A system has **two processes** and **three identical resources**. Each process needs a maximum of two resources. Is deadlock possible? Explain your answer.

(4 marks)

3. List **two reasons** why the scheduling of processes and threads on a multi-processor system is more complicated than scheduling them on a uni-processor system. (6 marks)

4. One of the design decisions in Operating System memory management is the choice between **swapping** and **paging**.
Define each of these terms and explain their respective roles in Operating System memory management. (8 marks)

5. When multiple processes need to cooperate, there is a choice between **shared memory** and **message passing communication**.

Give **two** advantages and **two** disadvantages of each method. Explain your answer.

(12 marks)

QUESTION II. CPU scheduling, Memory management and Disk scheduling

(37 marks)

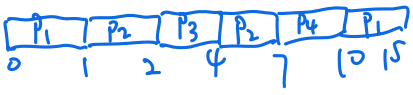
1. Consider the following scenario of processes. Their arrival time and burst time are as follows:

Process	Arrival time (ms)	Burst time (ms)
P1	0	6
P2	1	4
P3	2	2
P4	3	3

Draw the Gantt chart for the execution of the processes using the **Shortest Remaining Time First scheduling algorithm**. (4 marks)

Calculate **the average waiting time** for the system. (4 marks)

Calculate **the average turnaround time** for the system. (4 marks)



2. Calculate the number of page faults for the following sequence of page references (each element in the sequence represents a page number) using the **Least Recently Used (LRU) replacement algorithm** with frame size of **3**.

7 2 3 1 2 5 3 4 6 7 7 1 0 5 4 6 2 3 0 1

(6 marks)

3. Consider a disk queue with I/O requests on the following cylinders in their arriving order:

38, 180, 130, 10, 50, 15, 190, 90, 150

We assume a disk with 200 tracks (numbered 0 to 199) and the head is initially at position 120 and current direction of head is towards 0.

Write the sequence in which the requested tracks are serviced using the **C-LOOK algorithm** and calculate the **total head movement** (in number of cylinders) incurred while servicing these requests. (6 marks)

4. In a 32-bit machine we subdivide the virtual address into 3 segments as follows:

page number		page offset
10-bit	10-bit	12-bit

We use a **two-level** page table such that the first 10-bit of an address is an index into the first level page table and the next 10-bit is an index into a second level page table. Each page table entry is 32 bits in size.

(a). What is the page size in such a system? (3 marks)
(b). How many entries are in the 1st level page table? (3 marks)
(c). How much memory does the 1st page table occupy? (3 marks)

(a). $2^{12} = 4kB$ (b). $2^{10} = 1024$ (c). $2^{10} \times 32$ bits $= 4kB$

5. Three processes P1, P2, and P3 of size 900, 190, and 888 bytes, respectively, need space in memory.

If partitions of equal size, that is, 2000 bytes, are allocated to P1, P2, and P3, will there be any fragmentation in this allocation? If, yes, then what is the size of the space left? (4 marks)

P_1: 2000 - 900 = 1100 bytes P_3: 2000 - 888 = 1112 bytes
P_2: 2000 - 190 = 1810 bytes left: 1100 + 1810 + 1112 = 4022 bytes

QUESTION III. Resource allocation (12 marks)

Consider a system with the following information.

Available

R1	R2	R3	R4
1	0	1	0

Process	Max				Allocation			
	R1	R2	R3	R4	R1	R2	R3	R4
P0	0	1	2	3	0	1	1	2
P1	2	0	0	2	1	0	0	2
P2	0	2	0	1	0	1	0	1
P3	2	2	0	0	1	1	0	0
P4	0	2	2	0	0	1	2	0

3 0 1 2
3 1 3 5
5 3 3 5

Is this system in a **safe state**? If your answer is yes, please give a safe sequence and resources available after each process finished. If your answer is no, please specify the processes that might involve in a deadlock (unsafe). (6 marks)

safe. P_1, P_0, P_3, P_2, P_4

If a request from P3 arrives for (1, 0, 0, 0), can that request be safely granted immediately? Explain your answer. (6 marks)

No. $(1,0,1,0) - (1,0,0,0) = (0,0,1,0) > (0,0,0,0)$

But then the system is in a unsafe state.

QUESTION IV. Operating System in C Language (12 marks)

Consider the provided C language code implementing a barber shop scenario using semaphores, which includes a barber process and multiple customer processes:

```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>

#define MAX_CUSTOMERS 10
#define NUM_CHAIRS 5

sem_t barber_ready, customer_ready, mutex;
int num_waiting = 0;
```

barber process:	customer process:

void *barber(void *arg) {
  while (1) {
  sem_wait(&customer_ready);
  num_waiting--;
  sem_post(&barber_ready);
  printf("Barber is cutting hair\n");
  // Perform haircut
  }
}
	void *customer(void *arg) {
	  if (num_waiting < NUM_CHAIRS) {
	  num_waiting++;
	  sem_post(&customer_ready);
	  sem_wait(&barber_ready);
	  printf("Customer is getting a haircut\n");
	  // Receive haircut
	  } else {
	  printf("No available chairs. Customer leaves.\n");
	  }
	}

```
int main() {
  pthread_t barber_thread, customer_threads[MAX_CUSTOMERS];
  sem_init(&barber_ready, 0, 0);
  sem_init(&customer_ready, 0, 0);
  pthread_create(&barber_thread, NULL, barber, NULL);
  for (int i = 0; i < MAX_CUSTOMERS; i++) {
    pthread_create(&customer_threads[i], NULL, customer, NULL);
```

```
    }
    pthread_join(barber_thread, NULL);
    for (int i = 0; i < MAX_CUSTOMERS; i++) {
        pthread_join(customer_threads[i], NULL);
    }
    return 0;
}
```

a) Explain the purpose of the *barber_ready* and *customer_ready* semaphores in the barber shop scenario. (4 marks)

barber_ready indicates the barber is ready to cut hair,
customer_ready indicates the customer is ready to have haircut, The two semaphores will promise a synchronization between two processes.

b) Discuss potential race conditions that may occur in this implementation and propose a solution to prevent them. (4 marks)

The num_waiting will be modified by two processes. Add a mutex to the critical section.

c) Describe a scenario where starvation might occur in this implementation and suggest a modification to the code to mitigate this issue. (4 marks)

当椅子被坐满，新的顾客直接走掉，永远无法得到服务.

增加座位信号量，将每个顾客为等待座位.

END OF EXAM PAPER