1. (10%) Why are segmentation and paging sometimes combined into one scheme?

2. (20%) In the consumer-producer example program, a ring buffer queue is used to store the produced item that will be take off by the consumer later.
   a. In what conditions the program need to use the lock synchronization primitive to support the correct processing? (p240, Fig 6.10/6.11 8th edition)

```
int n;
semaphore mutex = 1;
semaphore empty = n;
semaphore full = 0

do {
   . . .
   /* produce an item in next_produced */
   . . .
   wait(empty);
   wait(mutex);
   . . .
   /* add next_produced to the buffer */
   . . .
   signal(mutex);
   signal(full);
} while (true);
```

Figure 5.9 The structure of the producer process.

Fig 6.10 producer process

```
do {
   wait(full);
   wait(mutex);
   . . .
   /* remove an item from buffer to next_consumed */
   . . .
   signal(mutex);
   signal(empty);
   . . .
   /* consume the item in next_consumed */
   . . .
} while (true);
```

Figure 5.10 The structure of the consumer process.

Fig 6.11 consumer process

b. In what conditions the program need NOT to use the lock synchronization primitive to support the correct processing? (p118, fig 3.14/3.15 8th edition)

```
#define BUFFER_SIZE 10

typedef struct {
    . . .
}item;

item buffer[BUFFER_SIZE];
int in = 0;
int out = 0;
```

```
while (true) {
     /* produce an item in next_produced */

     while (((in + 1) % BUFFER_SIZE) == out)
        ; /* do nothing */

     buffer[in] = next_produced;
     in = (in + 1) % BUFFER_SIZE;
}
```

Figure 3.13 The producer process using shared memory.

Fig. 3.14 producer process

```
item next_consumed;

while (true) {
     while (in == out)
        ; /* do nothing */

     next_consumed = buffer[out];
     out = (out + 1) % BUFFER_SIZE;

     /* consume the item in next_consumed */
}
```

Figure 3.14 The consumer process using shared memory.

Fig. 3.15 consumer process

背面還有 Please Turn Over

3. (10%) Define the difference between preemptive and nonpreemptive scheduling. Explain why strict nonpreemptive scheduling is unlikely to be used in a computer center.

4. (5%) How does DMA increase system concurrency? How does it complicate the hardware and system design?

5. (5%) Can a multithreaded solution using multiple user-level threads achieve better performance on a multiprocessor system than on a single processor system? Explain.

6. (13%) Suppose a thread is running in a critical section of code. It means that the thread has acquired all the locks through proper arbitration. Can this thread get context switched? Please explain the reasons.

7. (27%) Suppose that the following processes arrive for execution at the times indicated. Each process will run the listed amount of time. You may make some reasonable assumptions and write them down explicitly, if they are necessary to answer the following questions.
   (a) Please draw Gantt charts that illustrate the execution of these processes using the following scheduling algorithms: FCFS, non-preemptive SJF, and preemptive SJF.
   (b) Which of the algorithms in (a) results in the minimum average turnarround time (over all processes)? Be sure to justify your answer.
   (c) Which of the algorithms in (a) results in the minimum average waiting time (over all processes)? Be sure to justify your answer.

Process	Arrival Time	Burst Time
P1	0	10
P2	5	3
P3	3	5
P4	4	4

8. (10 %) Are the following statements about IP addresses true or false? For each statement, you will get 2 points for correct answer, zero point for blank, or -1 point for incorrect answer.
   (a) The subnet mask for the subnet 200.23.16.0/23 is 255.255.255.0.
   (b) The subnet 200.23.16.0/23 could accommodate up to 256 hosts.
   (c) Domain Name Service (DNS) can be used to acquire IP addresses.
   (d) Address Resolution Protocol (ARP) can be used to acquire IP addresses.
   (e) Network Address Translation (NAT) is used to map MAC addresses to IP addresses.