1.(40%)There are two example program, the following one(figure 5.9, 5.10) uses semaphore to implement it.(p240, Fig 6.10/6.11 8th edition,p269, Fig 6.9/6.10 9th edition)

1-a.(5%) What is the critical section in figure 5.9, 5.10 ?

1-b(5%)Please complete the code in produce/remove an item area. Such that multiple producers and consumers can be parallel running correctly in figure 5.9, 5.10?

```
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.

```
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.

In another consumer-producer example program figure 3.13 and 3.14, a ring buffer queue is used to store the produced item that will be take off by the consumer later.

1-c.(5%)The following figure 3.13 and 3.14 are another example program. What are the critical section in these multiple producers and multiple consumers program?

1-d.(5%)The critical section problem is caused by the race condition that the timer interrupt will break the updating statement of list queue pointer into non-atomic operation.
If there are only one producer and only one consumer program in running, Please indicate what is the critical region in 3.13 and 3.14 example program?

1-e.(5%)If the bounded buffer sharing is supported with hardware test&set locking. Please implement the primitive that can support multiple cooperation programs concurrently and effectively running in multiprocessor/multicore system. (p118, fig 3.14/3.15 8th edition,p123/124 9th 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.

```
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.

1-f.(5%) If kernel monitor(R.C.A. Hoare Monitor) approach is adopted to support producer and consumer program. Any processes that can not call consumer or producer function in monitor simultaneously. That means the monitor restricts the parallel processing of these two functions. What is your suggestion to use 3.13/3.14 or 5.9/5.10 algorithm in monitor to support multiprocessor parallel processing? Why?

2-a.(10%)Show how to implement the mutual exclusion operations in multiprocessor environments using the swap() instruction.

2-b.(10%)The solution should exibit minimal busy waiting . Please proof the three requirements for this lock operation.

3. (15 %) Are the following statements about IP addresses true or false? For each statement, you will get 3 points for correct answer, zero point for blank, or -2 point for incorrect answer.
   (a) If the resource allocation graph contains a cycle, then deadlock exists.
   (b) If a system is in unsafe state, then deadlock exists.
   (c) The subnet mask for the subnet 200.23.16.0/23 is 255.255.255.0.
   (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.

4. (20 %) Suppose that a disk drive has 5000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 134, and the previous request was at cylinder 125. The queue of pending requests, in FIFO order, is 86, 1470, 913, 3774, 948, 4509, 1022, 2750, 130. Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests, for SSTF and SCAN disk scheduling algorithms?

5. (15 %) Suppose that a scheduling algorithm (at the level of short-term CPU scheduling) favors those processes that have used the least processor time in the recent past. (a) Will this scheduling algorithm favor CPU-bound processes or I/O-bound processes? Why? (b) Please discuss if the starvation situation will permanently occur in this scheduling algorithm?