50.005 Quiz OS 4(15 mins)

Name: ___________________________ Student ID: __________________

Note: This quiz is closed-book and closed-notes, except for one double-sided A4 cheat sheet allowed. You also can't go online or look at anything electronic, including your laptop, smartphone, etc.

1. **[2m]** The two basic *kinds* of process/thread synchronisation problems (e.g., found in the producer-consumer problem) are __**mutual exclusion**________ and ______**condition synchronization**___________ .

2. **[1m]** In Java, a synchronized method of an object can successfully call another synchronized method of the same object although both synchronized methods are guarded by the same binary lock. **True or false?**
TRUE

3. Consider a single producer, single consumer problem in Java. The producer and consumer threads share a buffer of 5 integers (all initialized to 0). They synchronize using a semaphore declared as:

```
Semaphore num = new Semaphore(0);
```

Consider the following producer code:

```
 // insert input integer i into next slot of buffer
 public void insert(int i) {
     static int in = 0;

     buffer[in] = i;
     in = (in + 1) % 5;
     num.release();
}
```

and the following consumer code:

```
// print the next producer-inserted integer from the buffer
public void remove() {
    static int out = 0;
    int i;

    num.acquire();
    i = buffer[out];
    out = (out + 1) % 5;
    System.out.printf("%d ", i);
```

Date: 28/02/2019

```
            }
```
The producer thread calls the insert method 10 times with input parameter i equal to 1, 2, …, 10 (in that order). **Concurrently**, the consumer thread runs to call the remove method repeatedly. Note that because there are only one producer and one consumer, we ***do not* need** to provide mutual exclusion explicitly in accessing the shared buffer.

(a) **[2m]** The consumer prints the values 6, 7, 8, 4, 5, 6, 7, 8, 9, 10. Explain how this can happen.
There's nothing stopping the producer from overwriting the buffer, i.e: write before consumer finished reading the new values. The current semaphore only prevents the consumer from reading before the producer produces letters.

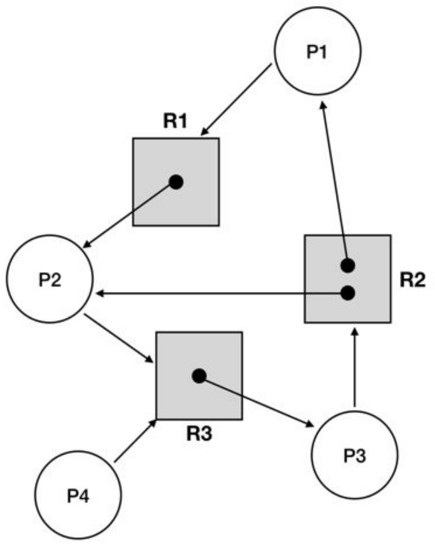
Full marks will be given for generic answer.

However the specific answer is: Producer calls insert(i) for i = 1, ..., 8. The "1 2 3" will be overwritten by "6 7 8", and num will be 8. Then, the consumer prints 8 times, "6 7 8 4 5 6 7 8", and num becomes 0. Then the producer calls insert for i = 9, 10 (num = 2). Then consumer prints 9, 10 (num = 0).

(b) **[1m]** What instead should be the correct sequence of integers printed by the consumer?
1,2,3,4,....10.

(c) **[2m]** Describe a fix to the insert and remove methods to ensure that the correct sequence in (b) will be printed.
Create another semaphore: space, initialize it to 5. Producer has to do space.acquire() before the first line of instruction in insert(). Consumer has to do space.release() just before quitting the function remove().

4. Consider the resource allocation graph below:



(a) **[1m]** Does the graph contain a cycle or not? **Yes**

(b) **[1m]** Does deadlock exist in the graph? **Yes**

(c) **[1m]** In general, a cycle in the graph is a sufficient condition for deadlock. **True or false? FALSE**

Total marks : 11 marks

Date: 28/02/2019