CITY UNIVERSITY OF HONG KONG

Course code & title	:	CS3103 Operating Systems
Session	:	Semester B 2019/20
Time allowed	:	50 minutes

This paper has 8 pages (including this cover page).

1. This paper consists of 2 sections, a total of 50 marks.
2. Section A: 26 marks.
3. Section B: 24 marks.
4. Answer ALL questions.
5. Specify the Section and Question number clearly for each answer in the answer script.
6. Submit ONE pdf file of the answer script to Canvas.
7. Use your Student ID to name the pdf file.

*This is an **open-book** quiz.*

Copy-and-paste the following honesty pledge in the beginning of the answer script and sign above your student ID and name to reaffirm your academic honesty pledge.

"I pledge that the answers in this exam/quiz are my own and that I will not seek or obtain an unfair advantage in producing these answers. Specifically,

- ***I will not plagiarize (copy without citation) from any source;***
- ***I will not communicate or attempt to communicate with any other person during the exam/quiz; neither will I give or attempt to give assistance to another student taking the exam/quiz; and***
- ***I will use only approved devices (e.g., calculators) and/or approved device models.***

I understand that any act of academic dishonesty can lead to disciplinary action."

____

Student ID: ________________________________

Student Name: ________________________________

Section A [26 marks]

1. [1 mark]

Which of the following is the most efficient for a multiple-word I/O transfer.

A) Interrupt-driven I/O
B) Programmed I/O
C) Direct memory access
D) All the above are equally efficient

2. [3 marks]

Complete the following table by inserting "high", "middle" or "low".

Type of memory	Cost per bit	Access time
Main memory	2(i)	2(iv)
Cache	2(ii)	2(v)
Disk	2(iii)	2(vi)

3. [1 mark]

The collection of program, data, stack, and attributes is referred to as the _________ .

4. [1 mark]

The requirement that when one process is in a critical section that accesses shared resources, no other process may be in a critical section that accesses any of those shared resources is ________ .

5. [1 mark]

A semaphore that does not specify the order in which processes are removed from the queue is a ________ semaphore.

6. [1 mark]

A ________ occurs when multiple processes or threads read and write data items so that the final result depends on the order of execution of instructions in the multiple processes.

7. [1 mark]

Probably the most useful combination, ________ allows a process to send one or more messages to a variety of destinations as quickly as possible.

A) blocking send, blocking receive
B) non-blocking send, blocking receive
C) non-blocking send, non-blocking receive
D) blocking send, non-blocking receive

8. [1 mark]

A situation in which two or more processes are unable to proceed because each is waiting for one of the others to do something is a ________ .

9. [1 mark]
A _______ is a semaphore that takes on only the values of 0 and 1.

10. [1 mark]
________ is a code segment within a process that requires access to shared resources and that must not be executed while another process is in a corresponding code segment.

11. [1 mark]
__________ refers to the ability of an OS to support multiple, concurrent paths of execution within a single process.

12. [2 mark]
After the I/O device issues an interrupt signal to the processor, the following events will happen. Write down the correct sequence of those events.

- (i) Processor loads new PC value based on interrupt
- (ii) Processor finishes execution of current instruction
- (iii) Processor pushes PSW and PC onto control stack
- (iv) Processor restores PSW and PC from the control stack
- (v) Process the interrupt

13. [2 marks]

Which of the following statements are correct description of kernel-level threads (KLTs)?

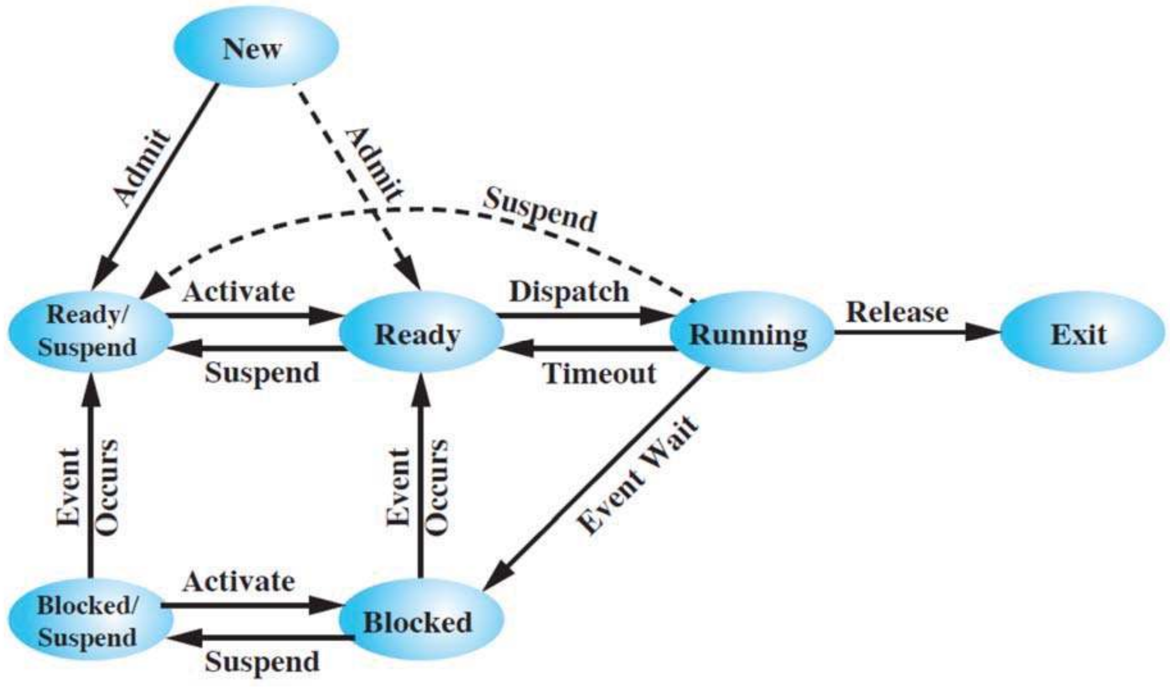
- (i) KLTs are created by invoking an application-level function.
- (ii) The kernel is aware of the existence of KLTs
- (iii) The transfer of control from one thread to another within the same process requires a mode switch to the kernel is one of the advantages of the KLT approach.
- (iv) Using KLTs cannot provide a better performance than a single-threaded solution on a single-processor system.

A) (ii), (iii) and (iv)
B) (i) only
C) (ii) only
D) (ii) and (iv)

14. [3 marks]
Complete the following table by indicating whether the state is for process only, thread only, or both.

State	process/thread/both
Running	14(i)
Suspend	14(ii)
Blocked	14(iii)

Consider the following 7-state process model in answering questions 15 and 16.



15. [3 marks]
Complete the following table by inserting the state of a process.

State	Status of the process	Location of the process
15(i)	Available for execution	Main memory
15(ii)	Completed running	Main memory
15(iii)	Awaiting an event	Secondary memory

16. [3 marks]
Complete the table by filling the state transition (may or may not be shown on the diagram).
The first one is an example.

Event	Transition
The dispatcher chooses a process to run.	Ready → Running
A process requests a resource which is not currently available.	16(i)
OS swaps out a periodic process waiting for the next execution.	16(ii)
A resource for which a process waiting in the main memory becomes available.	16(iii)

Section B [24 marks]

Q1 [8 marks]

Consider the following program fragment.

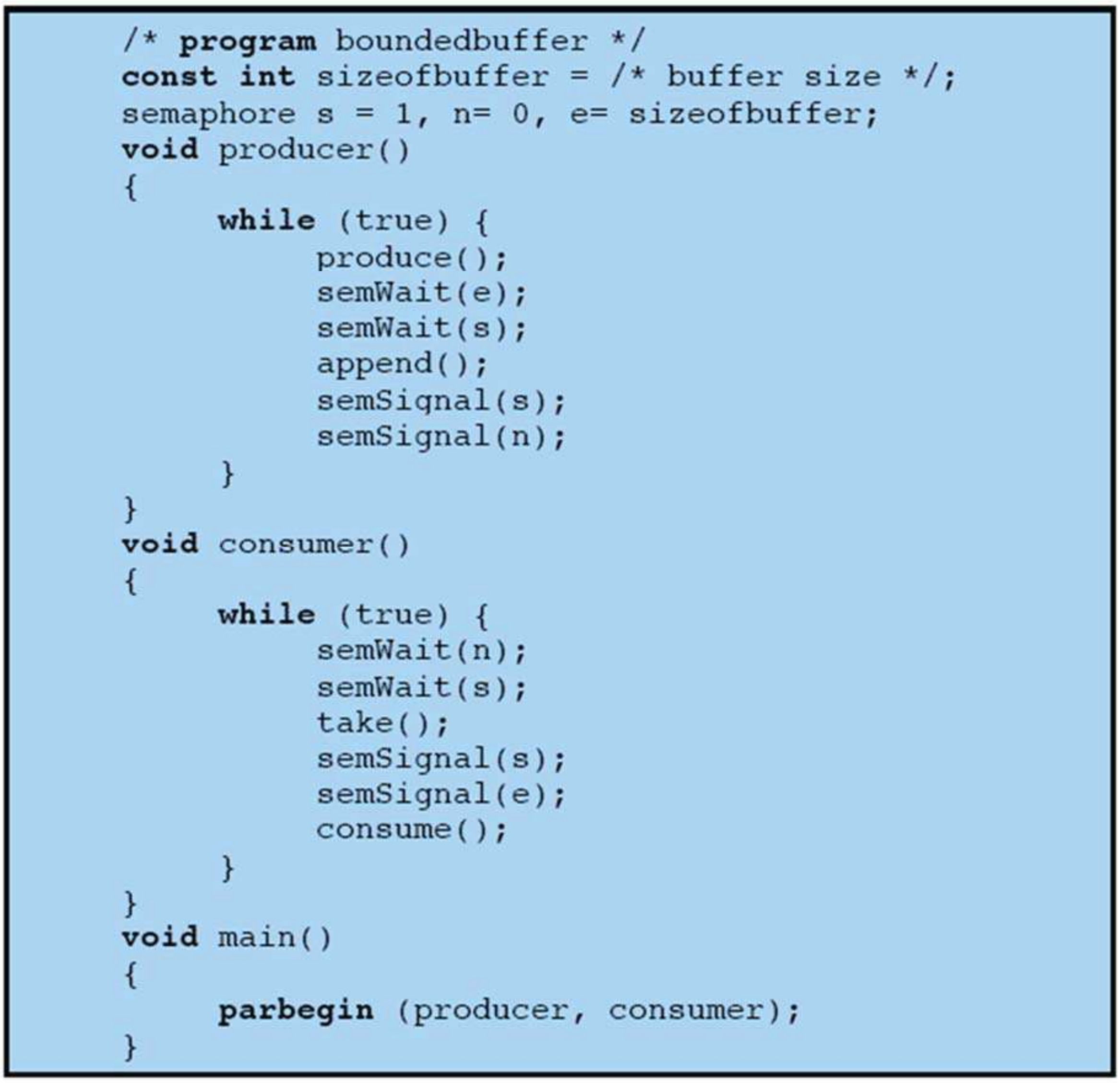
```
int x = 0;

void *runner (void *arg) {
      x+=2;
      cout << "thread: " << x << endl;
      pthread_exit(NULL);
}

int main ()
{
      // variables are declared here

      pid = fork();
      x++;
      if (pid == 0) {
            pthread_create(&tid, NULL, runner, NULL);
            pthread_join(tid, NULL);
            x+=3;
            cout << "process1: " << x << endl;
      }
      else {
            wait(NULL);
            x+=4;
            cout << "process2: " << x << endl;
      }
       pthread_exit(NULL);
}
```

(i) [4 marks]
List the outputs of the program. Show the order and values clearly.

(ii) [4 marks]
Briefly explain the values of the above output. No need to explain the order.

Q2 [4 marks]

Refer to the following solution to the bounded-buffer producer/consumer problem using semaphore. Assume that buffer size is 10.



```
/* program boundedbuffer */
const int sizeofbuffer = /* buffer size */;
semaphore s = 1, n= 0, e= sizeofbuffer;
void producer()
{
     while (true) {
          produce();
          semWait(e);
          semWait(s);
          append();
          semSignal(s);
          semSignal(n);
     }
}
void consumer()
{
     while (true) {
          semWait(n);
          semWait(s);
          take();
          semSignal(s);
          semSignal(e);
          consume();
     }
}
void main()
{
     parbegin (producer, consumer);
}
```

(i) What is the value of each of the three semaphores when a producer is appending data into an empty buffer while no consumer is waiting?

(ii) State the role of the semaphore `s`?

Q3 [12 marks]

Consider the following solution using ***semaphores*** to the *one-writer many-readers* problem.

```
int             readcount;
Semaphore       sem_x, sem_y;
```

Writer

```
semWait(sem_x);
/* writing performed */
semSignal(sem_x);
```

Readers

```
semWait(sem_y);
readcount++;
if readcount==1 then semWait(sem_x);
semSignal(sem_y);
/* reading performed */
semWait(sem_y);
readcount--;
if readcount==0 then semSignal(sem_x);
semSignal(sem_y);
```

A) [7 marks]
Suppose a *reader* is now reading in the system. No other readers or writer are waiting for the time being.
(i) [3 marks]
What are the current values of the three variables?

(ii) [2 marks]
What will happen when the *writer* wants to write while the *first reader* is still reading?

(iii) [2 marks]
Following the *writer*, what will happen when a *second reader* wants to read while the *first reader* is still reading?

B) [2 marks]
Which semaphore can be replaced by a ***mutex***? Explain

C) [3 marks]
Explain the potential issue/problem with this solution.

- END -