TA Yixin YANG (yyangfe@cse.ust.hk) is responsible for questions before the deadline
TA Peng YE (pyeac@cse.ust.hk) is responsible for grading and appeal handling after the deadline

Fall 2023 COMP 3511 Homework Assignment 3 (HW3)

Handout Date: October 30, 2023, Due Date: November 13, 2023

Name	Chloe hu
Student ID	21044009
ITSC email	chuap@connect.ust.hk

Please read the following instructions carefully before answering the questions:

- You should finish the homework assignment **individually**.
- When you write your answers, please try to be precise and concise.
- **Homework Submission:** submitted to **Homework #3** on **Canvas**.
- TA responsible for HW3: (Before the deadline, TA Yixin: yyangfe@cse.ust.hk and after the deadline, TA Peng: pyeac@cse.ust.hk)

1. [20 points] Multiple choices.

Please write down your answers in the boxes below:

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
A	**B**	D	A	C	D	B	D	A	B

1) What is the usage of the counting semaphore `empty` in the implementation of the bounded-buffer problem using semaphores?

A. It is used for counting the number of available slots in the buffer

B. It is used as a condition variable specifying when `empty = 0`, the producer process has to wait

C. Both A and B

D. None of the above

2) When using semaphores, a process invokes the `wait()` operation before accessing its critical section, followed by the `signal()` operation upon completion of its critical section. Consider reversing the order of these two operations—first calling `signal()`, then calling `wait()`. What would be a possible outcome of this?

A) Starvation is possible.

B) Several processes could be active in their critical sections at the same time.

C) Mutual exclusion is still assured.

D) Deadlock is possible.

3) Assume an `adaptive mutex` is used for accessing shared data on a Solaris system with multiprocessing capabilities. Which of the following statements is not true?

A) A waiting thread may spin while waiting for the lock to become available.

B) A waiting thread may sleep while waiting for the lock to become available.

C) The adaptive mutex is only used to protect short segments of code.

D) Condition variables and semaphores are never used in place of an adaptive mutex.

4) The second readers-writers problem ____.
A) requires that, once a writer is ready, it performs write as soon as possible.
B) is not used to test synchronization primitives.
C) requires that no reader will be kept waiting unless a writer has already obtained permission to use the shared database.
D) requires that no reader will be kept waiting unless a reader has already obtained permission to use the shared database.

5) Assume there are three resources, R1, R2, and R3, that are each assigned unique integer values 30, 40, and 25, respectively, as their priorities. The resources should be requested in an increasing order. What is a resource ordering which prevents a circular wait, thus a deadlock?
A. R3, R2, R1
B. R1, R2, R3
C. R3, R1, R2
D. R2, R1, R3

6) There are certain amount of instances of the same resource R in the system. Three concurrent processes P1, P2, and P3 respectively require 3,4 and 5 instances of R to finish their tasks. Based on such requirement, the system allocates 2, 3 and 4 instances of R to P1, P2, and P3, respectively. After the allocation, how many available instances of R are required at least to avoid deadlock?
A) 0
B) 1
C) 2
D) 3

7) Suppose that there are 10 resources available to three processes. At time 0, the following data is collected. The table indicates the maximum number of resources needed by each process and current resource allocation. Which of the following process sequence satisfies the safety criteria?

Process	Maximum Needs	Currently Owned
P_0	10	4
P_1	3	0
P_2	5	2

A. P1, P2, P0
B. P2, P0, P1
C. P0, P2, P1
D. It is not safe.

8) Consider a segment table of one process:

Segment #	Segment length	Starting address	Permission	Status
0	100	6000	Read-only	In memory

1	150	5500	Read/Write	In memory
2	350	4000	Read/Write	In memory

When accessing the logical address at <segment # = 2, offset = 400>, the results after address translation is:
A) No such segment
B) Return address 4400
C) Invalid permission
D) Address out of range

9) In a system with 32-bit address, given the logical address 0x0000F1AE (in hexadecimal) with a page size of 256 bytes, what is the page offset?
A) 0xAE
B) 0xF1
C) 0xA
D) 0xF100

10) A computer based on dynamic storage memory allocation has a main memory with the capacity of 55MB (initially empty). A sequence of operations including main memory allocation and release is as follows: 1. allocate 15MB; 2. allocate 30MB; 3. release 15MB; 4. allocate 8MB; 5. allocate 6MB. Using the best-fit algorithm, what is the size of the largest leftover hole in the main memory after the above five operations?
A) 7MB
B) 9MB
C) 10MB
D) 15MB

2. [20 points] Synchronization

You are asked to implement the second reader-writer solution, in which once a writer is ready, it needs to perform update as soon as possible. There are two classes of processes accessing shared data, *readers* and *writers*. Readers never modify data, thus multiple readers can access the shared data simultaneously. Writers modify shared data, so at most one writer can access data (no other writers or readers). This solution gives priority to writers in the following manner: when a reader tries to access shared data, if there is a writer accessing the data or if there are any writer(s) waiting to access shared data, the reader must wait. In another word, readers must wait for all writer(s), if any, to update shared data, or a reader can access shared data, only when there is no writer either accessing or waiting.

The following variables will be used:

semaphore mutex =1; /* lock for accessing shared variables */

semaphore readerlock=0; /* readers waiting queue */

semaphore writerlock=0; /* writers waiting queue */

int R_count = 0; /* number of readers accessing data */

int W_count = 0; /* number of writer accessing data */

TA Yixin YANG (yyangfe@cse.ust.hk) is responsible for questions before the deadline
TA Peng YE (pyeac@cse.ust.hk) is responsible for grading and appeal handling after the deadline

int WR_count = 0; /* number of readers waiting */

int WW_count = 0; /* number of writers waiting*/

Please fill in the blanks to design Writer's and Reader's code.

```
Writer() {
// Writer tries to enter
wait(mutex);
while ((R_count > 0 + W_count) > 0) { // Is it safe to write?
WW_count++;
Wait(writerlock)
WW_count--;
}
W_count++; // Writer inside
signal(mutex);

// Perform actual read/write access

// Writer finishes update
Wait(mutex)
W_count--; // No longer active
if (WW_count > 0){ // Give priority to writers
signal(writerlock); // Wake up one writer
} else if (WR_count > 0) { // Otherwise, wake reader
Signal(readerlock)
}
signal(mutex);
}
```

```
Reader() {
// Reader tries to enter
wait(mutex);
while (W_count > 0 || WW_count > 0) { // writer inside or
waiting?
WR_count++
wait(readerlock); // reader waits on readerlock
WR_count--
}
R_count++; // Reader inside!
signal(mutex);

// Perform actual read-only access

// Reader finishes accessing
wait(mutex);
R_count--; // No longer active
if (R_count == 0 && WW_count > 0) // No other active readers
signal(mutex);
}
```

TA Yixin YANG (yyangfe@cse.ust.hk) is responsible for questions before the deadline
TA Peng YE (pyeac@cse.ust.hk) is responsible for grading and appeal handling after the deadline

3. [30 points] Deadlocks

Consider the following snapshot of a system:

	Allocation					Max					Available			
	A	B	C	D		A	B	C	D		A	B	C	D
P0	2	0	0	1		4	2	3	4		3	3	2	1
P1	3	1	2	1		5	2	3	2					
P2	2	1	0	3		2	3	1	6					
P3	1	3	1	2		1	4	2	4					
P4	1	4	3	2		3	6	6	5					

Please answer the following questions using the banker's algorithm:
1) (5 points) What is the content of the matrix Need denoting the number of resources needed by each process?

	Need			
	A	B	C	D
P0	2	2	3	3
P1	2	1	1	1
P2	0	2	1	3
P3	0	1	1	2
P4	2	2	3	3

2) (10 points) Is the system in a safe state? Why?
Starting available: 3,3,2,1

	Allocation					Max				Order	Available			
	A	B	C	D		A	B	C	D		A	B	C	D
P0	2	0	0	1		4	2	3	4	4	10	11	8	7
P1	3	1	2	1		5	2	3	2	1	6	4	4	2
P2	2	1	0	3		2	3	1	6	5	12	12	8	10
P3	1	3	1	2		1	4	2	4	2	7	7	5	4
P4	1	4	3	2		3	6	6	5	3	8	11	8	6

Yes: p1, p3, p4, p0, p2 is safe to run

3) (5 points) If a request from process P1 arrives for (1, 1, 0, 0), can the request be granted immediately? Why?

Yes, because it fulfils the first requirement that $Request \le Need$ which in this case is (1,1,0,0) ≤ (2,2,3,3).

4) (10 points) If a request from process P4 arrives for (0, 0, 2, 0), can the request be granted immediately? Why?

No. The request fulfils the first two requirements where $Request \le Need$ (0, 0, 2, 0) < (2,2,3,3) , $Request \le Available$ (0, 0, 2, 0) < (3,3,2,1) .

The alloc for p4 would become (1,4,5,2), need would become (2,2,1,3) and avail would become (3,3,0,1). We now check if it is still safe:

Starting available: 3,3,0,1

	Need			
	A	B	C	D
P0	2	2	3	3
P1	2	1	1	1
P2	0	2	1	3
P3	0	1	1	2
P4	2	2	1	3

Since there are 0 instances in the C column and each process needs at least 1 instance of C, the system is not in a safe state and therefore can't process the request immediately.

4. [30 points] Memory Management

1) (15 points) Consider the segment table shown in Table A. Translate each of the virtual addresses in Table B into physical addresses. Indicate errors (out of range, no such segment) if an address cannot be translated.

Table A

Segment number	Starting address	Segment length
0	260	90
1	1466	160
2	2656	130
3	146	50
4	2064	370

Table B

TA Yixin YANG (yyangfe@cse.ust.hk) is responsible for questions before the deadline
TA Peng YE (pyeac@cse.ust.hk) is responsible for grading and appeal handling after the deadline

Segment number	Offset
0	420
1	144
2	198
3	296
4	50
5	32

0	Illegal address; out of range (420 > 90)
1	1466+144 = **1610**
2	Illegal address; out of range (198 > 130)
3	Illegal address; out of range (296 > 50)
4	2064+50 = **2114**
5	No such segment

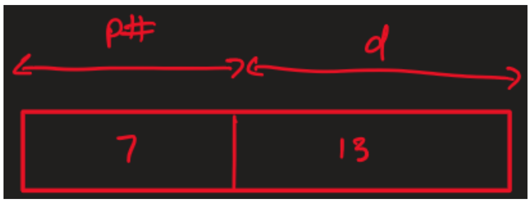
2) (15 points) Consider a virtual memory system providing 128 pages for each user program; the size of each page is 8KB. The size of main memory is 256KB. Consider one user program occupied 4 pages, and the page table of this program is shown as below:

Logical page number	Physical block number
0	9
1	8
2	6
3	3

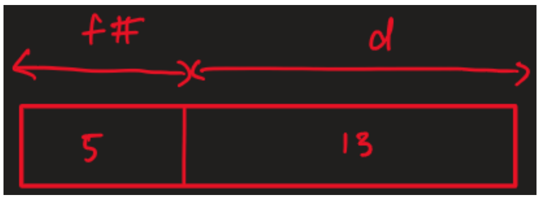
Assume there are three requests on logical address 040AFH, 140AFH, 240AFH. First please describe the format of the logical address and the physical address. Then please illustrate how the virtual memory system will deal with these requests.

First, size(page) = 8KB = 2^13B. Therefore, page offset (d) = 13 bits.
Size(virtual memory) = 128 pages * 8KB = 1024KB = 2^20 B. Therefore, page number is 7 and offset is 13.

TA Yixin YANG (yyangfe@cse.ust.hk) is responsible for questions before the deadline
TA Peng YE (pyeac@cse.ust.hk) is responsible for grading and appeal handling after the deadline



Size(physical memory) = 256KB = 2^18B. Therefore, frame number is 5 and offset is 13.



For 040AFH, the binary representation is 0000 0100 0000 1010 1111. The page number is the first 7 bits 0000010 which is 2 in decimal and the last 13 bits are the offset. From the page table, a page number of 2 translates to a frame number of 6. So, the physical address is 00110 0 0000 1010 1111. The first 5 bits are the frame number, and the last 13 bits are the offset.

For 140AFH and 240AFH, the page number is 0001 010 (11) and 0010 010 (18) respectively and are not in the page table. So, the system encounters a page fault.