The University of Nottingham Ningbo China

SCHOOL OF COMPUTER SCIENCE

A LEVEL 2 MODULE, AUTUMN SEMESTER 2017-2018

COMP2046(AE2OSC), Operating Systems and Concurrency

Time allowed: TWO hours

Candidates may complete the front cover of their answer book and sign their desk card but must NOT write anything else until the start of the examination period is announced

You must answer THREE questions out of FOUR
(Only the THREE nominated solutions will be marked)

Only silent, self-contained calculators with a Single-Line Display are permitted in this examination.

Dictionaries are not allowed with one exception. Those whose first language is not English may use a standard translation dictionary to translate between that language and English provided that neither language is the subject of this examination. Subject specific translation dictionaries are not permitted.

No electronic devices capable of storing and retrieving text, including electronic dictionaries, may be used.

DO NOT turn your examination paper over until instructed to do so

ADDITIONAL MATERIAL: None

INFORMATION FOR INVIGILATORS:
Please collect the exam papers at the end of the exam.

Question 1 (Processes and process scheduling) [25 marks]

a) List the three main process states and explain all the valid transitions between them. List each of those transitions that would involve a context switch, and describe an event that might cause such a transition.

[5 marks]

b) The job of operating system is working like a resource manager, providing for an orderly and controlled allocation of these resources. Explain what is the structure used by an operating system to keep track of different resources and how to protect them. List all the information needed by operating system to manage the process abstraction.

[5 marks]

c) On a processor computer, multiprogramming enables more than a single process to apparently execute simultaneously.

i) Briefly explain how this is achieved on a uniprocessor?

ii) Suppose you have a multiprogramming system running 3 independent processes. If each process spends 50% of its time waiting for I/O, calculate what will be the CPU utilization improvement (in percentage) by doubling the number of running processes?

[5 marks]

d) There are several design goals in scheduling algorithms. In the user-oriented perspective, list and describe the criteria to evaluate these algorithms. Give an example where these criteria can be conflicting.

[3 marks]

e) Illustrate the use of a Round Robin algorithm for the runnable process list given below.

iii) Assuming all the processes arrive at time 0, illustrate times when processes start and finish running (you can use Gantt charts similar to the ones used in the lecture slides to illustrate this or a table). List the times for each context switch in your illustration. You can assume that the time slice is 15 milliseconds (if you were to need this)

	FCFS Position	CPU burst time (ms)
Process A	1	55
Process B	2	35
Process C	3	15
Process D	4	20

[3 marks]

iv) Calculate the average response time for the round robin algorithm.

[2 marks]

v) Calculate the average turnaround time for the round robin algorithm.

[2 marks]

Question 2 (Memory Management) [25 marks]

a) You are trying to run a C code 'hello world', but it can't finish running. You try to figure out the reason by monitoring the page fault rate and CPU utilization. Explain what is happening when you found increasing the number of processes leads to the increase of page fault rate and decrease of CPU utilization. List the methods can be used to recover from this problem.

[5 marks]

b) Assuming a 16-bit address space and the page table listed below, calculate the physical addresses for the following logical addresses:

i) 3286
ii) 20480
iii) 25810

Pages		Frames	
0	0000	0010	2
1	0001	0001	1
2	0010	0110	6
3	0011	0000	0
4	0100	0100	4
5	0101	0011	3
6	0110	X	X
7	0111	X	X
8	1000	X	X
9	1001	0101	5
10	1010	X	X
11	1011	0111	7
12	1100	X	X

[6 marks]

c) To overcome some of the difficulties with fixed partitioning, an approach known as dynamic partitioning was developed. Please answer questions below.

i) Explain how dynamic partitions work, and the problems associated with them, in multiprogramming operating system?

[3 marks]

ii) If a bitmap is used to keep track of free memory, what will be the size of bitmap for a 128MB memory? You can assume that the size of memory block equals to 4KB.

[2 marks]

iii) Consider a swapping system in which memory consists of the following hole sizes in memory order: (a) 10 MB (b) 4 MB (c) 20 MB (d) 11 MB (e) 18 MB (f) 16 MB (g) 9 MB and (h) 15 MB Which hole is taken for successive partition requests of 12 MB, 9 MB and 4 MB for first fit, next fit and worst fit respectively?

[3 marks]

d) Assume a machine that has a 64-bit virtual address space, and a 32-bit physical address space (i.e., the maximum amount of physical memory for a 32-bit address space is present).

i) How many entries would you expect a single level page table to have when the page size is 4 kilobytes. Briefly explain your answer.

[3 marks]

ii) Assuming that the memory access time is 90ns and the time it takes to search the translation lookaside buffer is 15ns. What is the average memory access time for a page table with 3 levels and a 95% hit rate (write down in equations how you get the result)?

[3 marks]

Question 3 (Concurrency control) [25 marks]

(a) Briefly describe the conditions to ensure a good mutual-exclusion solution in concurrent process control.

[4 marks]

(b) Consider the following program:

```
#include <stdio.h>
#include<stdlib.h>
#include<pthread.h>


pthread_mutex_t lock;       //declare a mutex
char str1[40] ="this is a hello message from a thread 1\n";
char str2[40] ="this is a hello message from a thread 2\n";
void print_string(char str[], int iter){
    int i,j;
    for(i = 0; i < iter;i++)
    {
        for(j=0; j<40; j++) { printf("%c",str[j]);}
    }
}
void * print1(void * n) { //print str1 n times
    print_string(str1, *((int*)n));
    return NULL;
}
void * print2(void * n){ // print str2 n times.
    print_string(str2, *((int*)n));
    return NULL;
}
int main()
{
    int number_of_messages = 500;
    pthread_t tid1,tid2;
    pthread_mutex_init(&lock,NULL);  //initialize a mutex lock
    pthread_create(&tid1, NULL, print1, (void *) &number_of_messages);
    pthread_create(&tid2, NULL, print2, (void *) &number_of_messages);
    pthread_join(tid1,NULL);
    pthread_join(tid2,NULL);
    pthread_mutex_destroy(&lock);
    return 0;
}
```

The program attempts to use two interleaved threads to print to the console two different text messages multiple times concurrently but wants to ensure that each message is printed out in entirety before another message is printed. That is, the output should be something like this:

```
this is a hello message from thread 1
this is a hello message from thread 2
this is a hello message from thread 1
this is a hello message from thread 2
… …
```

Unfortunately the program outputs the following:

```
tthhiiss  iiss  aa  hheellllloo  mmeessssaaggee  ffrroomm  aa
tthhrreeaadd  12

tthhiiss  iiss  aa  hheellllloo  mmeessssaaggee  ffrroomm  aa
tthhrreeaadd  12

tthhiiss  iiss  aa  hheellllloo  mmeessssaaggee  ffrroomm  aa
tthhrreeaadd  12

tthhiiss  iiss a  a hello message from a thread 2
hello message from a thread 1
```

(i) Please explain what went wrong and how the program can be corrected by giving correct code segments.

[3 marks]

(ii) The program uses a mutex to synchronise threads. What are the pros and cons of using mutex in comparison with a semaphore?

[3 marks]

(c) Consider the following piece of C code:

```
void main(){
   fork();
   fork();
   exit();
}
```

How many child processes are created upon execution of this program? WHY?.

[3 marks]

(d) Multi-threading is a useful technique to speed up many matrix related operations. The following C program is designed to compute the product of all elements in a matrix through multi-threading. Please complete the missing program blocks in functions `calc` and `main` so that the program would always print out the expected results.

```
#include <stdio.h>
#include<stdlib.h>
#include<pthread.h>
#include<semaphore.h>

int sum = 0;
int row=4;
int col=5;
int matrix[4][5]={{1,1,1,1,1},{2,2,2,2,2},{3,3,3,3,3},{4,4,4,4,4}};
sem_t s;

void * calc(void * row_index) {
    int r = *((int*) row_index);

    /* add you code segment 1 here */

}
int main()
{
    pthread_t tid[row];
    if(sem_init(&s,0,1)==-1){
```

```
        printf("Unable to create semaphore!\n" );
        exit(-1);
    }
    int r[row];
    int i;
    for(i=0; i<row; i++){
        r[i]=i;
        if(pthread_create(&tid[i], NULL, calc, (void *) &r[i]) == -1) {
           printf("unable to create thread!\n");
           exit(-1);
        }
    }


    /* add you code segment 2 here */


    printf("The sum of the matrix is: %d\n", sum);
    return 0;
}
```

[8 marks]

(e) Explain how i-node addresses the issues of FAT file systems.

[4 marks]

Question 4 (Miscellaneous) [25 marks]

(a) Explain how hard links differs from soft links with respective to i-node allocations.

[4 marks]

(b) Explain what starvation is in process scheduling and how it can be addressed.

[4 marks]

(c) Consider the following multi-process concurrency problem scenario: Processes A, B, C and D are sharing resources of type of X, Y, Z and W. Each resource type has exactly one unit available. For the following process scheduling, explain whether a deadlock will occur or not. You may use a directed graph as part of your explanation.

A requests X, A requests Y,
B requests W, C requests Z,
D requests Y, B requests Y,
D requests W, A requests W

[4 marks]

(d) Consider a concurrency problem with four processes, P1, P2, P3, and P4, and five types of resources, RS1, RS2, RS3, RS4 and RS5. Assume that the total number of existing resources vector E = (2 4 1 4 4) and a following state:

C

	RS1	**RS2**	**RS3**	**RS4**	**RS5**
P1	0	1	1	1	2
P2	0	1	0	1	0
P3	0	0	0	0	1
P4	2	1	0	0	0

R

	RS1	**RS2**	**RS3**	**RS4**	**RS5**
P1	0	1	1	1	2
P2	0	1	0	1	0
P3	0	0	0	0	1
P4	2	1	0	0	0

Using the Bankers algorithm with multiple resources to show that there is a deadlock in the system and identify the processes that are deadlocked.

[5 marks]

(e) List one advantage and one disadvantage for each of the following file systems:

i) Contiguous
ii) Linked list
iii) File allocation table
iv) I-nodes

[4 marks]

(f) Consider a disk with 300 tracks, and the following sequence of requested tracks 80, 150, 230, 50, 220, 50, 190, 260, 210, 270. Assume that the disk arm is located at position 60 and is moving in the up direction (i.e. 1 to 300). Calculate the number of tracks crossed if using the `shortest seek time first` disk scheduling algorithm.

[4 marks]