NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING

ASSESSMENT FOR
Semester 1 AY2017/2018

CG2271 – Real-Time Operating Systems

Nov 2017 **Time Allowed: 2 hours**

INSTRUCTIONS TO STUDENTS:

1. This assessment paper contains **FOUR (4)** questions and comprises **EIGHTEEN (18)** printed pages, including this page.
2. Students are required to answer ALL the questions.
3. Write your answers within the space provided. Answers written on other parts of the answer script will not be graded unless you specify explicitly.
4. This is a **CLOSED BOOK** assessment.
5. Allow a help sheet in a single A4 paper, and calculator.
6. Please write your student number below.

Student Number: _____

This portion is for examiner's use only

Question	Marks	Remarks
Q1 [6 marks]		
Q2 [12 marks]		
Q3 [36 marks]		
Q4 [26 marks]		
Total [80 marks]		

QUESTION 1: Indicate whether the following statements are true or false. Justify your answers. [6 marks]

(A) If the utilization of a task set is higher than one, there still possibly exists a feasible schedule.

[1 mark]

(B) Compared with earlier deadline first (EDF), rate monotonic scheduling (RMS) can better utilize the processor.

[1 mark]

(C) Data inconsistency does not occur, if processes perform only read operations on the shared data.

[1 mark]

(D) Internal fragmentation in the main memory can happen when the size of memory allocation is fixed.

[1 mark]

(E) In a multi-processing system that is executing multiple tasks, if the memory usage of any task is smaller than the main memory capacity, then page faults will never happen.

[1 mark]

(F) With virtual memory support, operating systems can run a process that has the memory requirement larger than physical memory.

[1 mark]

QUESTION 2: Short Q&A [12 marks]

(A) What is priority inversion? Please describe a method to solve priority inversion.
[3 marks]

(B) We learnt internal and external fragmentation in "memory allocation" chapter. For the virtual memory model, where virtual & physical addresses are used and the mapping between them are maintained, does it have internal or external fragmentation? Indicate your answer and explain the reasons.

[3 marks]

(C) Explain why TLB (Translation Lookaside Buffer) can improve the performance of page table lookups.

[3 marks]

(D) Describe the two major kinds of overheads involved in the execution time of the procedure for page fault handling.

[3 marks]

QUESTION 3: Calculations [36 marks]

(A) Suppose that the p=calloc(n, b) returns a pointer p to n consecutive units of memory each b bytes long, in the form of an array. For example, p=calloc(5, 6) will return a pointer p to a block of memory that is 30 bytes long. free(p) frees the memory pointed to by p, and sizeof(x) returns the size of variable x in bytes.

A char variable is 1 byte long, and a double variable is 8 bytes long.

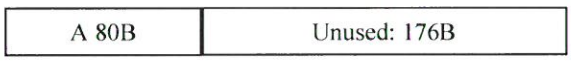
We have a memory system that consists of 256 bytes in total and memory is allocated in units of 8 bytes (e.g., even requesting for 1 byte will still return an 8-bytes of memory). We assume starting at address 0. Consider the following program fragment:

```
double *A = calloc(10, sizeof(double));
double *B = calloc(10, sizeof(double));
char *C = calloc(12, sizeof(char));
free(B);
double *D = calloc(10, sizeof(double));
```

(1) We are using linked-lists to manage free memory. Sketch the linked-lists after each memory operation for a **next fit** policy. In each sketch, you need to indicate a memory region associated with the variable name and the memory size. Replace the variable name with "unused" if a memory region is unused. Indicate FAIL if you are unable to perform the operation shown, and explain why the operation fails. For your convenience, we show the sketch for the first memory operation. Please show the sketches for the remaining memory operations.

[4 marks]

double *A = calloc(10, sizeof(double));



double *B = calloc(10, sizeof(double));

<Draw your sketch here>

char *C = calloc(12, sizeof(char));

<Draw your sketch here>

free(B);

<Draw your sketch here>

double *D = calloc(5, sizeof(double));

<Draw your sketch here>

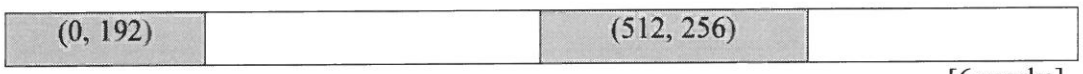
(2) What is the percentage of memory that is wasted because of internal fragmentation after these memory operations? Express your percentage with respect to the total amount of memory requested by the program.

[2 marks]

(3) What would be the percentage of memory that is wasted due to internal fragmentation if we allocated in units of 12 bytes instead of 8 bytes?

[4 marks]

(B) Assume there is a buddy memory allocation system with the total size of 1024 bytes. All allocations of memory are in power of 2 bytes. Suppose the initial status of the memory system is as shown below. The shaded regions indicate the memory already used by other programs. Each pair of numbers under each shaded region "(*off*, *sz*)" means that the memory region is allocated at the offset of *off*, with the size of *sz* bytes. Illustrate how the following memory requests are handled: (a) request 150 bytes; (b) request 30 bytes; (c) request 40 bytes; (d) release 30 bytes requested in (b); (e) release 150 bytes requested in (a); (f) request 120 bytes. You should draw a sketch of the memory system like the one below after each memory request.



[6 marks]

(C) Consider a memory system with 12-bit virtual address, 1024 byte main memory, and 128 byte page size.

Fill in the blanks in the following table. Assume each table entry contains one bit to indicate the status of a page and some minimum bits to indicate the ID of a physical page.

[6 marks]

Number of bits for page offset	
Number of bits of physical address space	
Maximum size in bytes of virtual memory	
Maximum Number of physical pages	
Maximum Number of virtual pages	
Total size of page table	

(D) There are three tasks as shown below. Assume all of them are ready at time 0. The deadline for each task is the same as the period.

Task	Worst Case Execution time	Period
T1	2	10
T2	5	15
T3	10	40

(1) What is the hyper-period for the task set?

[1 mark]

(2) Is this set of tasks RMS schedulable under the utilization bound analysis? Perform the calculation and explain your answer. The values of sufficient condition B(n) are given in the below table.

B(1)=1.0	B(4)=0.756	B(7)=0.728
B(2)=0.828	B(5)=0.743	B(8)=0.724
B(3)=0.779	B(6)=0.734	

[3 marks]

(3) Is T3 RM-schedulable? Justify by the critical instant analysis.

[5 marks]

(4) Give the EDF schedule of this task set for the first 20 units of time.

[5 marks]

QUESTION 4: Design and Analysis [26 marks]

(A) Consider two processes P1 and P2 in Table Q4. Semaphores S and Q are both initialized to be 0. X, Y and Z are shared variables among P1 and P2. X, Y and Z are all initialized with 2. What are all the possible combinations for the values of X, Y and Z after P1 and P2 complete?

[6 marks]

Table Q4

Process P1	Process P2
Y=1;	X=1;
V(S);	X=X + 2;
Y=Y + Z;	P(S);
Z=Y + 1;	X=X - Y;
P(Q);	V(Q);
Y=Z + Y;	Z=X + Z;

(B) Consider a system with virtual memory paging support. The hard disk is used as paging disk. CPU utilization measures the ratio of the time of executing instructions from the running processes to the total CPU time. Bandwidth utilization of an I/O device is the percentage of the device's bandwidth that is currently being consumed by I/O traffic. At some time, we observed the following time-measured utilizations:

CPU utilization 10%

Paging disk bandwidth utilization 98.7%

Bandwidth utilization of other I/O devices 5%

For each of the following actions, describe whether it will (probably) improve CPU utilization, and then briefly justify your answer.

[10 marks]

a. Install a faster CPU.

b. Install more main memory.

c. Install a faster hard disk.

d. Increase the page size.

(C) Alice and Bob share a plate of pasta. There is only one fork and thus only one of them can eat at a time. To be fair, they must take turns to eat. It does not matter who eats first. Alice and Bob are modeled as two processes, and implemented as the below pseudo code.

Semaphore declarations:

```
semaphore mutex = 1;   semaphore Bob= 0;
semaphore Alice=0;
```

Bob Process:

```
while (1) {
  P(mutex);
  eat();
  V(Alice);
  P(Bob);
  V(mutex);
}
```

Alice Process:

```
while (1) {
  P(mutex);
  eat();
  V(Bob);
  P(Alice);
  V(mutex);
}
```

(1) Identify the problem of this program, and justify your answer.

[4 marks]

(2) Write the correct solution based on the above code snippet, and justify your answer. You are only allowed to permute the order of statements and change the initial value of semaphores.

[6 marks]

BLANK PAGE
END OF ASSESSMENT