# Review Test Submission: OS Exam 2 5348

| | |
|---|---|
| User | Yaokun Wu |
| Course | CS 5348.001 - Operating Systems Concepts - S21 |
| Test | OS Exam 2 5348 |
| Started | 4/2/21 10:01 AM |
| Submitted | 4/2/21 11:41 AM |
| Status | Completed |
| Attempt Score | 90.4 out of 127 points |
| Time Elapsed | 1 hour, 39 minutes out of 2 hours |

## Question 1

7 out of 7 points

1. Monitor

Consider a system with N printers. We use a monitor to implement a printer manager to coordinate all the printing requests. When a client has a printing job, it calls assign_printer to get the id of a free printer for submitting the printing job. If there is no free printer at the moment, the client has to wait till a printer becomes available. After printing is done, the client calls free_printer to release it. Limit your answer within 30 words.

```
Monitor printer_manager:
{  Boolean printerStatus\[N], initialized to free;
   int avail;
   condition printerAvail;

   int assign_printer ()
   {  avail = N;
      for (i=0; i<N; i++)
        if (printerStatus\[i] == free)
          {  avail = i;   printerStatus\[i] = busy;   break;  }
      printerAvail.wait;
      printerStatus\[avail] = busy;
      return (avail);
   }

   void free_printer (int printerid)
```

```
    {  avail = printerid;
       printerStatus\[avail] = free;
       printerAvail.signal;
    }
}

void printer_client:
 {  int printerid = assign_printer ();
    send printing job to printer with printerid;
    wait for printing to finish;
    free_printer (printerid);
 }
```

In function assign_printer, either the i-th printer is available and get returned, or no printer is available and the function is only returned when a printer is freed by free_printer. However, the above code is incorrect and the highlighted statement should be replaced by the code below.

```
if (avail < N) return (avail); else printerAvail.wait;
```

(a) Assume that the incorrect code (without the replacement) is used. What incorrect behavior will happen which deviates from the goal of the printer manager. Limit your answer within 30 words.

## Question 2

0 out of 7 points

1. Continued

(b) For monitor implementation, we can allow signaler or signalee to continue after signaling a condition variable. Which approach will be better considering the Monitor code for printer manager. Give concise and precise justification within 50 words. Your justification should state the specific property in the printer manager code that lead to the choice and the specific overhead reduction achieved due to the choice.

## Question 3

4 out of 3 points

2. fork() and piple()

Consider the code below which uses fork() and pipe() system calls.

```
int a = 0;   int b = 0;   int c, d;
int pfd[2];     // pipe file descriptors

void parent_function (int arg)
 {   a = 1;   b = 2;   c = 2*a + b;   d = a + 2*b;
     pipe (pfd);
     if (arg != 0) write (pfd[1], &c, sizeof(int));
     else write (pfd[1], &d, sizeof(int));
     printf ("parent:%d,%d\n", a, b);
```

```
    close (pfd[1]);
}

void child_function ()
{   read (pfd[0], &a, sizeof(int));
    read (pfd[0], &b, sizeof(int));
    printf ("child:%d,%d\n", a, b);
    close (pfd[0]);
}

void main ()
{   int ret = fork ();
    parent_function (ret);
    if (ret == 0) child_function ();
    wait();
}
```

Give the output of the **child** process. Put each line of output in the corresponding box below, following the order of printing. We give 3 boxes for three lines of output. If the output only contains 1 line, you should put "**none**" (without the quotation marks) for lines 2 and 3. Your answer for each line should contain no space (e.g., parent:2,0), (e.g, child:2,0).

First output line: **[A]**

Second output line: **[B]**

Third output line: **[C]**

## Question 4

2.4 out of 12 points

3. Message passing model

We want to implement the Producer-Consumer problem in a message passing system. Besides the N Producer processes and M Consumer processes, we also use a Buffer-Manager process to manage the buffer used for the Producers and the Consumers. Buffer-Manager uses two ports to facilitate the control of communication with the Producers and the Consumers. When the buffer is full, the Buffer-Manager should not retrieve any produced item from any Producer. When the buffer is empty, the Buffer-Manager should not retrieve any consumer request from any Consumer.

Which of the following sets of constructs are sufficient for implementing the Buffer-Manager for the bounded buffer problem specified above? Assume that the Buffer-Manager should only have a single thread and should not fork any subprocesses. Select all those that can be used to correctly and sufficiently implement the Buffer-Manager.

Note: BSBR is blocked send and blocked receive. NSNR is nonblocked send and non-blocked receive.

## Question 5

16 out of 16 points

4. Deadlock

Consider a system with three types of resources R1, R2, and R3. Four processes P1, P2, P3, and P4 are accessing these three resources. The system uses Banker's algorithm to achieve deadlock avoidance. The current system state and a new request for resources (last column) are given in the following table.

| | Total | MaxReq | Allocated | New Request |
|---|---|---|---|---|
| | | P1 P2 P3 P4 | P1 P2 P3 P4 | P1 P2 P3 P4 |
| R1 | 6 | 5 4 3 1 | 0 2 2 1 | 0 0 0 0 |
| R2 | 5 | 4 2 2 3 | 2 0 0 2 | 0 0 0 1 |
| R3 | 5 | 2 5 1 1 | 2 1 1 1 | 0 0 0 0 |

Follow Banker's algorithm to determine whether to grant the request.

(a) At each step, determine which process can proceed next. If P1 can proceed, then answer P1. If none can proceed, then answer "none". If multiple processes can proceed, choose the process with the lowest index.

Which process can proceed at the first step? **[A1]**

Which process can proceed at the second step? **[A2]**

Which process can proceed at the third step? **[A3]**

(b) What are the available number of resources for R1, R2, R3 after the 3 steps in (a). You should list the numbers in the order of R1, R2, R3, and separate them by comma with no space in between. For example, answer "3,4,5" means 3 of R1, 4 of R2, and 5 of R3 are available.

## Question 6

14 out of 14 points

5. Deadlock

Consider the following three system states A, B, and C.

A:

| | Total | MaxReq | Allocated | Need | Avail |
|---|---|---|---|---|---|
| | | P1 P2 P3 | P1 P2 P3 | P1 P2 P3 | |
| R1 | 6 | 2 4 3 | 2 2 1 | 0 2 2 | 1 |
| R2 | 5 | 3 4 3 | 1 2 0 | 2 2 3 | 2 |
| R3 | 4 | 1 3 3 | 1 0 2 | 0 3 1 | 1 |

B:

| | Total | MaxReq | Allocated | Need | Avail |
|---|---|---|---|---|---|
| | | P1 P2 P3 | P1 P2 P3 | P1 P2 P3 | |
| R1 | 6 | 6 6 1 | 1 1 1 | 5 5 0 | 3 |
| R2 | 3 | 2 2 0 | 1 2 0 | 1 0 0 | 0 |

C:

| | Total | MaxReq | Allocated | Need | Avail |
|---|---|---|---|---|---|
| | | P1 P2 P3 | P1 P2 P3 | P1 P2 P3 | |
| R1 | 3 | 2 3 2 | 1 0 2 | 1 3 0 | 0 |
| R2 | 7 | 5 7 3 | 1 2 2 | 4 5 1 | 2 |
| R3 | 2 | 1 1 0 | 1 1 0 | 0 0 0 | 0 |

(a) Use Banker's algorithm to check whether each system is in deadlock. Which of the system state(s) given above has(have) a deadlock? **[A]**

(b) For any system, no resources are allocated to any process in the beginning. Each system state (A, B, or C) is reached after a sequence of resource requests. If we correctly follow the Banker's algorithm when granting each request, which state(s) is(are) reachable? **[B]**

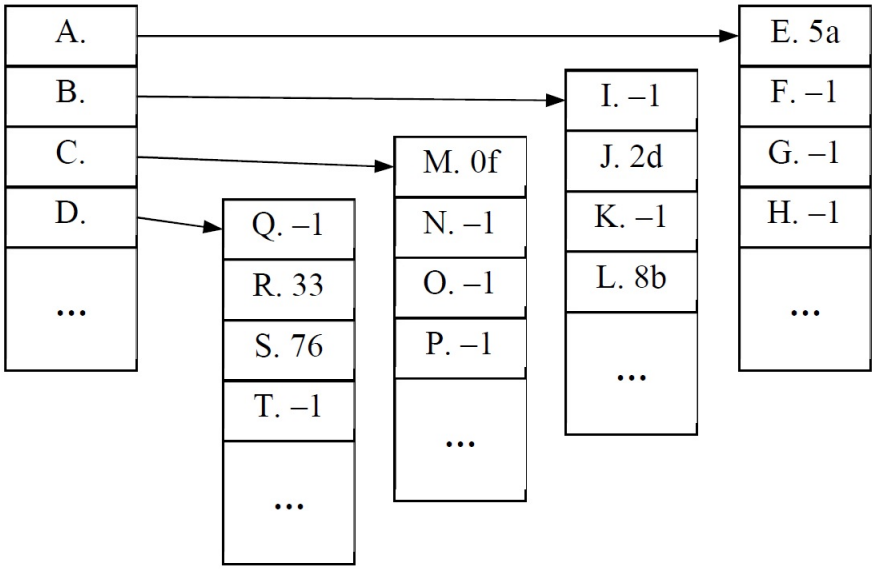
For both (a) and (b), choose all the states that satisfies the sub-question. Use the state indexes A, B, and/or C in your answer and your answer should contain no separator or space and should be in alphabetical order (e.g., ABC). If there is no state satisfies the question, then answer "**none**" (no quotation marks).

## Question 7

16 out of 9 points

6. Two-level PT

Consider a system with **1GB** (2^30 bytes) physical memory. It uses virtual memory, demand paging and the two-level page table scheme is used to keep track of the memory allocation for each process. The virtual address is of **32** bits and **13 bits** are used for addressing the page offset, **10 bits** are used for addressing the second level process page table. The partial two-level page table for process **X** is shown below.




The first 4 entries in the root page table are indexed by A to D. Each contains a pointer (value is not given) pointing to a second level process page table as shown.

Each process page table entry is also indexed by an alphabet on the left. Its content is the **frame number** for the corresponding page (in **hexadecimal**). Each frame number should have more bits, but we only use 8 bits for simplicity. When the frame number is –1, it means the page is on disk.

(a) What is the page size of the memory (in decimal)? **[A]**

(b) How many entries will there be in the root page table (in decimal)? If the number of entries in the root page table will be different for each process, your answer should be "**undecidable**" (without quotation marks). **[B]**

(c) Consider logical address: 0x008033a8. What is the corresponding physical address? Use 32-bit hexadecimal for your answer (e.g. 001256ef). **[C]**

(d) Which entries in the two-level page table have been retrieved to compute the physical address from the logical address given in (c): 0x008033a8? Use the entry indexes (A-T) in your answer. If there are multiple entries being retrieved, put the indexes in alphabetical order with no spaces or separators in between (e.g., ABT).

## Question 8

5 out of 10 points

7. IPC and TLB

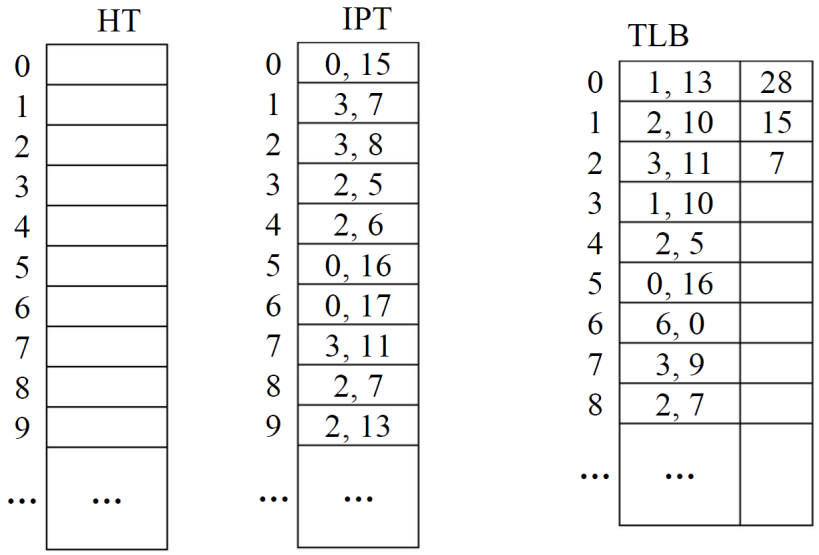
Consider a system with **1GB** (2^30 bytes) physical memory. It uses virtual memory, demand paging with **32-bit** virtual addresses and each page is of size **4KB** (2^12 bytes). As shown in the diagram below, an inverted page table (IPT) is used to map memory frames to virtual pages of processes.

Each entry in the IPT contains a tuple (*pid*, *pnum*), where *pid* specifies the process id and *pnum* specifies the page number of that process.

The hash table (HT) is also given in the diagram but its content is not shown. Assume that the HT is larger than the IPT and there will be no collision and, hence, no chaining is needed.

The diagram also shows the translation lookaside buffer (TLB). Its first column is (*pid*, *pnum*) and second column is for frame numbers (only a few sample values are given). (Leftmost part of each table is for indexing, which is not a true column of the table). TLB caches mapping entries for multiple processes and it allows each process to cache up to 6 mapping entries.

Note: All the *pid* and *pnum* and frame numbers in the diagram are in **decimal**.



HT

| | |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |
| 7 | |
| 8 | |
| 9 | |
| … | … |

IPT

| | |
|---|---|
| 0 | 0, 15 |
| 1 | 3, 7 |
| 2 | 3, 8 |
| 3 | 2, 5 |
| 4 | 2, 6 |
| 5 | 0, 16 |
| 6 | 0, 17 |
| 7 | 3, 11 |
| 8 | 2, 7 |
| 9 | 2, 13 |
| … | … |

TLB

| | | |
|---|---|---|
| 0 | 1, 13 | 28 |
| 1 | 2, 10 | 15 |
| 2 | 3, 11 | 7 |
| 3 | 1, 10 | |
| 4 | 2, 5 | |
| 5 | 0, 16 | |
| 6 | 6, 0 | |
| 7 | 3, 9 | |
| 8 | 2, 7 | |
| … | … | |

Though the contents of IPT and TLB are not fully shown, assume that currently processes 0, 2, and 3 only have 3, 4, and 3 pages in memory and TLB only cached 1, 3, and 2 mapping entries for them, respectively. The information for other processes is not known.

(a) Let 2^x (2 to the power of x) denote the number of entries in the IPT? What shall x be? **[A]**

(b) In the above diagram, some entries in TLB should not have been there. List all the entries in the TLB above that should have been removed. Your answer should be a list of the indexes (0-8) in numerical order separated by comma and without space in between (e.g., 0,1,2). **[B]**

Additional subquestions will follow.

## Question 9

12 out of 15 points

7. Continued

Currently, process 2 (pid = 2) is in execution and it has a sequence of memory references with the logical addresses given below. Assume that there is no access violation in any of these memory references.

A. 0x00007e28
B. 0x00006030
C. 0x00001020
D. 0x00006d58
E. 0x00001e3c
F. 0x0000df14
G. 0x000079fc

(c) Give the physical address for the logical address given in A: 0x00007e28, in hexadecimal. **[C]**

(d) Which memory reference(s) will cause page fault(s)? **[D]**

(e) Which memory reference(s) will cause new entry(entries) being put into TLB? **[E]**

For (d) and (e), your answer should contain the index(es) given above in alphabetical order with no separator or space (e.g., ABC). If none of the above memory accesses satisfy the sub-question then answer "**none**" (no quotation marks).

## Question 10

8 out of 16 points

7. Continued

The memory hierarchy includes TLB, IPT (with HT), memory, and disk. The system has no cache for memory content. Assume that each TLB access takes **10ns** (ns is nanosecond), each physical memory access takes **100ns**, and the disk access latency is **1000ns** (always getting a disk block). (Note that the above access time data are not realistic.) Also assume that the TLB hit rate is 95% and the memory hit rate is 99.9%.

Compute the total access time for some of the individual memory accesses given in the previous part of the question.

(f) What is the total access time for A in the sequence (0x00007e28)? **[F]**

(g) What is the total access time for B in the sequence? **[G]**

(h) What is the total access time for C in the sequence? **[H]**

(i) What is the total access time for D in the sequence? **[I]**

Your answer for (f) to (i) should be in nanoseconds, and should only contain the numerical value, not the unit (e.g., 100). Note, do not forget the access time for memory content.

Hint: when computing access time, first compute the number of accesses to each level of the memory hierarchy.

## Question 11

6 out of 18 points

8. Working set size

Consider the following pseudo matrix manipulation program.

Assumptions: Each integer is of size **4** bytes, the system memory has a page size of **2KB**, the instructions of the following program fits in one page, and the variables other than the matrices are stored in one page (different from the instruction page).

```
#define Max 64   // define Max as a constant
int A[Max,Max], B[Max,Max];
int C[Max,Max], D[Max,Max];
int i, j, x, …;
```

```
void read_matrices ()
 { read in A and B }

void encrypt ()
 { // all variables used in this function are global
   // and are stored in the same page as i, j, x.     }

void main ()
 { read_matrices ();
   for (i=0; i<Max; i++)
     for (j=0; j<Max; j++)
       { x = random ();
         C[i,j] = A[i,j] * B[i,j] + x;
         D[i,j] = encrypt (C[i,j]);
 }     }
```

Consider the impact of different working set sizes to the performance of the above program. With different working set sizes given below, compute the number of page faults to complete the execution of the program. Note: **Do not count** the initial loading of pages to the working set. Also, consider the optimal scenario.

(a) When working set size = 5 pages, what will be the number of page faults? **[A]**

(b) When working set size = 6 pages, what will be the number of page faults? **[B]**

(c) When working set size = 7 pages, what will be the number of page faults? **[C]**

Saturday, April 3, 2021 5:26:24 PM CDT

← OK