PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

SISTEMAS OPERATIVOS
3ra práctica (tipo a)
(Primer semestre de 2016)

Horario 0781: prof. V. Khlebnikov
Horario 0782: prof. A. Bello R.

Duración: 1 h. 50 min.
Nota: No se puede usar ningún material de consulta.
La presentación, la ortografía y la gramática influirán en la calificación.
Puntaje total: 20 puntos

Pregunta 1 (10 puntos – 50 min.) La memoria para los procesos tiene el tamaño de 256K (2^18, 0x40000) bytes y se asigna a los procesos por bloques de 4K (2^12, 0x1000) bytes. Así, el bloque #0 está en la dirección 0x00000, el bloque #1 está en la dirección 0x01000, el bloque #2 está en la dirección 0x02000, …, el bloque #63 está en la dirección 0x3f000. El proceso A ocupa 12K de los primeros 3 bloques (#0, #1, #2), los bloques de #3 a #5 están libres, el proceso B ocupa los siguientes 8K, le siguen 12 bloques libres, el proceso C ocupa los siguientes 20K, le siguen 4 bloques libres, el proceso D ocupa los siguientes 40K, le siguen 8 bloques libres, el proceso E ocupa los siguientes 4K, le siguen 16 bloques libres.

a) (2 puntos – 10 min.) Presente el estado actual del mapa de bits para la memoria manejada. El mapa se guarda en bytes y el bit menos significativo corresponde al menor número del bloque.

b) (2 puntos – 10 min.) Presente el estado actual de la lista enlazada para la memoria manejada. En los nodos se guarda el número del bloque como el inicio del fragmento de la memoria y la cantidad de los bloques que forman este fragmento.

Llegan las siguientes solicitudes de memoria: 21K, 42K, 11K, 43K (en este orden).

c) (2 puntos – 10 min.) Si el algoritmo de asignación de memoria es *First Fit*, ¿cuál será el estado final del **mapa de bits**?

d) (2 puntos – 10 min.) Si el algoritmo de asignación de memoria es *Best Fit*, ¿cuál será el estado final de la **lista enlazada**?

e) (1 punto – 5 min.) Si, después de asignación de memoria solicitada con el algoritmo *Best Fit*, termina el proceso D, ¿cuál será el estado final de la **lista enlazada**?

f) (1 punto – 5 min.) Si, después de asignación de memoria solicitada con el algoritmo *Best Fit* y la terminación del proceso D, termina el proceso C, ¿cuál será el estado final de la **lista enlazada**?

Pregunta 2 (2 puntos – 10 min.) Your ISA supports a 36b address space and your 2GiB RAM is broken into 2KiB pages. How many bits are in a VPN? (0,5 puntos), PPN? (0,5 puntos).If all the flags (valid, dirty, etc) take up 12 bits per entry, how many bits does an entire page table take up? (1 punto). Explain your answers.

Pregunta 3 (3 puntos – 15 min.) (*Operating System Concepts by Abraham Silberschatz*) Consider the two-dimensional array A:

```
int A[][] = new int[100][50];
```

where `A[0][0]` is at location 200, in a paged memory system with pages of size 200. A small process is in page 0 (locations 0 to 199) for manipulating the matrix; thus, every instruction fetch will be from page 0.

For three page frames, how many page faults are generated by the following array-initialization loops, using LRU replacement, and assuming page frame 1 has the process in it, and the other two are initially empty? Explain your answers.

a)(1 punto – 5 minutos)

```
for (int j = 0; j < 50; j++)
       for (int i = 0; i < 100; i++)
              A[i][j] = 0;
```

b)(2 puntos – 10 minutos)

```
for (int i = 0; i < 100; i++)
       for (int j = 0; j < 50; j++)
              A[i][j] = 0;
```

Pregunta 4 (5 puntos – 25 min.) As described in your textbook, virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. To speed up this translation, modern processors implement a cache of the most recently used translations, called the translation-lookaside buffer (TLB). This exercise shows how the page table and TLB must be updated as addresses are accessed.

The following list is a stream of virtual addresses as seen on a system. Assume 4-KB pages and a four entry fully associative TLB with LRU replacement policy. If pages must be brought in from disk, give them the next largest unused page number (i.e., starting at 13).

```
0x0FFF
0x7A28
0x3DAD
0x3A98
0x1C19
0x1000
0x22D0
```

Initial TLB: (Note: Values are in base-10)

Valid	Tag	PP #	LRU
1	11	12	2
1	7	4	3
1	3	6	4
0	4	9	1

The LRU column works as follows: The older an entry is, the lower its LRU number will be. Each time an entry is used, its LRU number is set to 4 (since it is now the most-recently used), and the other numbers are adjusted downward accordingly.

Initial Page Table: (Note: Values are in base-10)

Index	Valid	Physical Page or On Disk
0	1	5
1	0	Disk
2	0	Disk
3	1	6
4	1	9
5	1	11
6	0	Disk
7	1	4
8	0	Disk
9	0	Disk
10	1	3
11	1	12

We start by checking all the tags in the TLB for a match. For a correct match, the valid bit must also be set. If there is a match we have a TLB hit (H), and all we have to do is update the LRU bits.

If there is no match in the TLB, we have to check the page table. Use the page number (tag) as an index into the page table. If the entry is valid, we have a TLB miss (M). Evict the least recently used entry in the TLB and replace it with the new translation. Remember to update all of the LRU bits and set the valid bit as well.

Finally, if the entry in the page table is invalid, our page is on disk and we have a page fault (PF). As per the instructions, assign it a new page number (starting with 13), and set its valid bit in the page table. Then you must update the TLB by evicting the LRU entry as before.

The ending tables will look like the following:

Complete the table **(1 punto – 5 minutos)**

Address	Result (H,M,PF)
0x0FFF	
0x7A28	
0x3DAD	
0x3A98	
0x1C19	
0x1000	
0x22D0	

TLB: Draw one table TLB for each virtual address **(3 puntos – 15 minutos)**

Page Table: draw the page table **(1 punto – 5 minutos)**

La práctica ha sido preparada por VK(1) y AB(2,3,4)
en Linux Mint 17.3 Rosa con LibreOffice Writer.

Profesor del curso: (0781) V. Khlebnikov
(0782) A. Bello R.

Pando, 27 de mayo de 2016