PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

SISTEMAS OPERATIVOS
4ta práctica (tipo a)
(Primer semestre de 2019)

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.
La práctica debe ser desarrollada en el cuadernillo usando lapicero.
Lo escrito con lápiz NO será evaluado.
Puntaje total: 20 puntos

Pregunta 1 (4 puntos – 20 min.)
a) (1 punto – 5 min) "It is worth noting that the problem of "page replacement" occurs in other areas of computer design as well. For example, most computers have one or more memory caches consisting of recently used 32-byte or 64-byte memory blocks. When the cache is full, some block has to be chosen for removal. This problem is precisely the same as page replacement except on a shorter time scale (it has to be done in a few nanoseconds, not milliseconds as with page replacement). The reason for the shorter time scale is that … ". Complételo.

b) (1 punto – 5 min) "In order to allow the operating system to collect useful page usage statistics, most computers with virtual memory have two status bits, *R* and *M*, associated with each page. It is important to realize that these bits must be updated on every memory reference, so it is essential that they be set by the hardware. If the hardware does not have these bits, they can be simulated using the operating system's page fault and clock interrupt mechanisms. When a process is started up, all of its page table entries are marked as not in memory. As soon as any page is referenced, a page fault will occur. The operating system then sets the *R* bit (in its internal tables), changes the page table entry to point to the correct page, with … , and restarts the instruction. If the page is subsequently modified, another page fault will occur, allowing the operating system to set the *M* bit and change the page's … ." ¿Por qué con la modificación de la página que ya está en la memoria sucede un fallo de página y esto permite marcarla con el bit *M* y cambiando qué?

c) (1 punto – 5 min) "Although second chance is a reasonable algorithm, it is unnecessarily inefficient because … . A better approach is … . Not surprisingly, this algorithm is called … .". ¿Por qué es ineficiente? ¿Cuál estrategia es mejor? ¿Cómo se llama?

d) (1 punto – 5 min) ¿En qué consiste la estrategia que se llama "*demand paging*"?

Pregunta 2 (3 puntos – 15 min.) "The aging algorithm is a descendant of the NFU algorithm, with modifications to make it aware of the time span of use. Instead of just incrementing the counters of pages referenced, putting equal emphasis on page references regardless of the time, the reference counter on a page is first shifted right (divided by 2), before adding the referenced bit to the left of that binary number. For instance, if a page has referenced bits 1,0,0,1,1,0 in the past 6 clock ticks, its referenced counter will look like this: 10000000, 01000000, 00100000, 10010000, 11001000, 01100100. Page references closer to the present time have more impact than page references long ago. This ensures that pages referenced more recently, though less frequently referenced, will have higher priority over pages more frequently referenced in the past. Thus, when a page needs to be swapped out, the page with the lowest counter will be chosen.
The following Python code simulates the aging algorithm.

```python
def simulateAging(Rs, k):
    Vs = [0]*len(Rs[0])
    print(' t  |  R-bits (0-5)        |  Counters for pages 0-5')
    for t,R in enumerate(Rs):
        for i in range(len(Vs)):
            Vs[i] = R[i] << k-1 | Vs[i] >> 1
        print('%02d  |  %s  |  [%s]'%(t, R, ', '.join([format(V, '0%db'%k) for V in Vs])))

Rs = [[1,0,1,0,1,1],[1,1,0,0,1,0],[1,1,0,1,0,1],[1,0,0,0,1,0],[0,1,1,0,0,0]]
k = 8
simulateAging(Rs, k)
```

In the given example of R-bits for 6 pages over 5 clock ticks, the function prints the following output, which lists the R-bits for each clock tick *t* and the individual counter values V_i for each page in binary representation.

```
 t  |  R-bits (0-5)        |  Counters for pages 0-5
00  |  [1, 0, 1, 0, 1, 1]  |  [10000000, 00000000, 10000000, 00000000, 10000000, 10000000]
...
```

Note that aging differs from LRU in the sense that aging can only keep track of the references in the latest 16/32 (depending on the bit size of the processor's integers) time intervals. Consequently, two pages may have referenced counters of 00000000, even though one page was referenced 9 intervals ago and the other 1000 intervals ago. Generally speaking, knowing the usage within the past 16 intervals is sufficient for making a good decision as to which page to swap out. Thus, aging can offer near-optimal performance for a moderate price."
Complete la salida que produce el programa e indique qué página será reemplazada en caso de un fallo de página.

<u>Pregunta 3</u> (3 puntos – 15 min.) Complete las siguientes tablas:

```
FIFO   1 2 3 4 1 2 5 1 2 3 4 5      Optimal   1 2 3 4 1 2 5 1 2 3 4 5      LRU   1 2 3 4 1 2 5 1 2 3 4 5
P1                                  P1                                     P1
P2                                  P2                                     P2
P3                                  P3                                     P3
                      PFs =                                PFs =                                  PFs =

FIFO   1 2 3 4 1 2 5 1 2 3 4 5      Optimal   1 2 3 4 1 2 5 1 2 3 4 5      LRU   1 2 3 4 1 2 5 1 2 3 4 5
P1                                  P1                                     P1
P2                                  P2                                     P2
P3                                  P3                                     P3
P4                                  P4                                     P4
                      PFs =                                PFs =                                  PFs =
```

<u>Pregunta 4</u> (2 punto – 10 min.) Un sistema operativo desea apoyar la posibilidad de recuperar archivos borrados. Para ello, cuando elimina un archivo, cambia la primera letra del nombre por el caracter ASCII extendido 156 (£) y mantiene los atributos en la entrada de directorio. Además, las entradas que correspondía a la cadena de bloques que ocupaba, los coloca a cero. Sin embargo es necesario que el sistema operativo tome una política al momento de asignar bloques libres a un archivo nuevo, para que la recuperación tenga éxito ¿cuál es esta política?

<u>Pregunta 5</u> (3 puntos – 15 min.) The open-file table is used to maintain information about files that are currently open. Should the operating system maintain a separate table for each user or just maintain one table that contains references to files that are being accessed by all users at the current time? If the same file is being accessed by two different programs or users, should there be separate entries in the open file table?

<u>Pregunta 6</u> (1 punto – 5 min.) If the operating system were to know that a certain application is going to access the file data in a sequential manner, how could it exploit this information to improve performance?

<u>Pregunta 7</u> (4 puntos – 20 min.) Consider a file system on a disk that has both logical and physical block sizes of 512 bytes. Assume that the information about each file is already in memory. For the following allocation strategies: contiguous and linked, answer these questions:

a. How is the logical-to-physical address mapping accomplished in this system? (For the indexed allocation, assume that a file is always less than 512 blocks long.) .

b. If we are currently at logical block 10 (the last block accessed was block 10) and want to access logical block 4, how many physical blocks must be read from the disk?

La práctica ha sido preparada por AB (4-7) y VK (1-3)
en Linux Mint 19.1 Tessa con LibreOffice Writer.

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

Pando, 14 de junio de 2019