Exámen Final – 25 de febrero de 2022

# Sistemas Operativos – OSTEP

## Virtualización

**Ejercicio 1.** Explique detalladamente como funcionan estos tres programas indicando cuantos procesos se crean, que hacen los padres, etc.
Suponga que en ambos casos se invocan con, y el archivo `entrada.txt` existe y no es ejecutable.

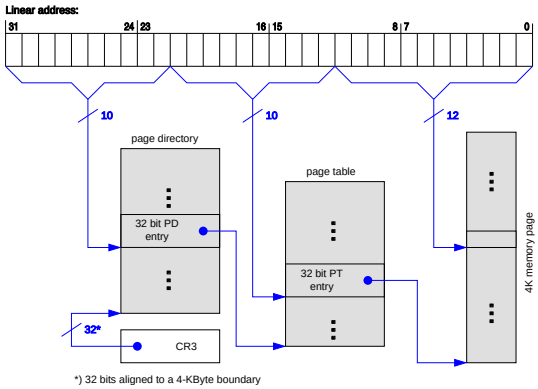
```
$ echo "Hola" > entrada.txt
$ ./a.out entrada.txt
```

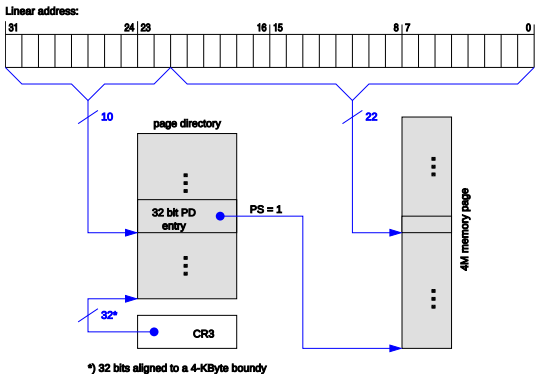
```c
int main(int argc, char ** argv) {
  close(6);
  close(4);
  close(2);
  close(0)
  open(argv[1], 0);
  open(argv[0], 0);
  open(argv[0], 0);
  open(argv[1], 0);
  read(2, buf, L);
  write(1, buf, L);
}
```

```c
int main(int argc, char ** argv) {
    execvp(argv[argc-1], argv);
    if (fork())
      main(argc, argv);
  }
}
```

```c
int main(int argc, char ** argv) {
    fork();
    execvp(argv[0], argv);
    main(argc, argv);
  }
}
```

**Ejercicio 2.** Para el sistema de paginado i386 con el bit de `PSE` (page size extension) dentro de `CR4` habilitado, el esquema de paginación puede ser $(10, 10, 12)$ o $(10, 22)$, dependiendo del bit 7 de cada entrada del *pagedir* denominado `PS` (page size).




Sabiendo que `CR3=0x0BEBE` y que el contenido de los marcos físicos son los siguientes:

```
0x0BABA                0x0BEBE                0x0B0B0
----------------       ----------------       ----------------
0x000: 0x10000 (P)     0x000: 0x0BEBE (P)     0x000: 0x00000 (P)
0x001: 0x10000 (P)     0x001: 0x103FF (P,PS)  0x001: 0x00001 (P)
...                    ...                    ...
...                    ...                    ...
0x3FE: 0x10000 (P)     0x3FE: 0x0B000 (P,PS)  0x3FE: 0x001FE (P)
0x3FF: 0x00000 (P)     0x3FF: 0x0BABA (P)     0x3FF: 0x001FF (P)
```

(a) Pasar de virtual a física: `0x00000505`, `0x00401FEE`, `0x00402FEE`, `0xFF8338FF`.

(b) Pasar de física a todas las virtuales: `0x00000000`, `0x0BFFFFFF`, `0x10000AAA`.

Ejercicio 3. Considere el siguiente multiprograma de dos componentes `P0` y `P1`, donde la **atomicidad es línea a línea** y todas las variables $(i, j, a)$ son compartidas.

| Pre: $i$=0 $\wedge$ $j$=0 | |
|---|---|
| `P0 : while(i<16) {`<br>`  a[i] = j;`<br>`  ++i;`<br>`}` | `P1 : while(j<32) {`<br>`  ++j;`<br>`}` |
| Post: ? | |

(a) Exprese de manera concisa todos resultados posibles en el arreglo $a[0, 16)$. Explique.
(b) ¿Se puede cumplir con la postcondición $(\forall k : 1 \leq k < 16 : a[k-1] < a[k])$? Explique.
(c) Sincronice con semáforos para lograr que **siempre** se satisfaga la postcondición $(\forall k : 1 \leq k < 16 : a[k-1] < a[k])$ sin hacer peligrar la terminación y maximizando la concurrencia. Solo puede agregar ifs para hacer los wait y post, no se puede tocar el resto del programa.

Ejercicio 4. Debajo se muestra solución **incorrecta** al problema de la sección crítica de dos procesos, presentada por Hyman en 1966, a fin de competir con el algoritmo de Dekker.
(a) Muestre un **escenario de ejecución** donde no se cumple la condición de sección crítica.
(b) ¿**Siempre** funcional mal? Explique.

| $Pre : \neg flag_0 \wedge \neg flag_1 \wedge turn = 0$ | |
|---|---|
| $P_0$ : **do** $true \rightarrow$ | $P_1$ : **do** $true \rightarrow$ |
| 1 $NCS_0$ | A $NCS_1$ |
| 2 ; $flag_0 := true$ | B ; $flag_1 := true$ |
| 3 ; **do** $turn = 1 \rightarrow$ | C ; **do** $turn = 0 \rightarrow$ |
| 4 **do** $flag_1 \rightarrow$ **skip** | D **do** $flag_0 \rightarrow$ **skip** |
| 5 **od** | E **od** |
| 6 $;turn := 0$ | F $;turn := 1$ |
| 7 **od** | G **od** |
| 8 ; $CS_0$ | H ; $CS_1$ |
| 9 ; $flag_0 := false$ | I ; $flag_1 := false$ |
| **od** | **od** |

Ejercicio 5. El disco rotacional *Seagate Mach.2 Exos 2X14* de 14 TiB e intefaz SAS 3.0, tiene una velocidad de rotacional de 7200 RPM, 4.16 ms de latencia de búsqueda y 524 MiB/s de tasa de transferencia máxima.
Este es **el disco rotacional más rápido del mundo** y eso es gracias a que tiene **dos juegos de cabezales independientes**.

(a) Indicar cuantos *ms* tarda en dar una vuelta completa.

(b) Indicar la tasa de transferencia de lectura **al azar** de bloques de 8 MiB [1].

(c) Si la tasa de transferencia máxima está dada por la velocidad rotacional que no requiere cambio de pista (no sufre del *seek time*), deducir cuantos MiB almacena cada cilindro.

Ejercicio 6. En un sistema de archivos de tipo UNIX, tenemos los bloques de disco dispuestos dentro del *i-nodo* con 12 bloques directos, 1 bloque indirecto, 1 bloque doble indirecto, 1 bloque triple indirecto. Cada bloque es de 4 KiB.
(a) Calcule la capacidad máxima **de un archivo** para números de bloque de 16, 24 y 32 bits.
(b) Calcule la capacidad máxima **de la *data region*** para números de bloque de 16, 24 y 32 bits.
(c) Realice un análisis de que longitud conviene para codificar el número de bloque.

[1] Acá hay que contemplar en las cuentas que hay dos cabezales independientes esperando por datos.