Nº Orden	Apellido y nombre	L.U.	Cantidad de hojas

Organización del Computador 2
Recuperatorio del segundo parcial – 26-11-2013

1 (40)	2 (40)	3 (20)	

Normas generales

- Numere las hojas entregadas. Complete en la primera hoja la cantidad total de hojas entregadas.
- Entregue esta hoja junto al examen, la misma **no** se incluye en la cantidad total de hojas entregadas.
- Está permitido tener los manuales y los apuntes con las listas de instrucciones en el examen. Está prohibido compartir manuales o apuntes entre alumnos durante el examen.
- Cada ejercicio debe realizarse en hojas separadas y numeradas. Debe identificarse cada hoja con nombre, apellido y LU.
- La devolución de los exámenes corregidos es personal. Los pedidos de revisión se realizarán por escrito, antes de retirar el examen corregido del aula.
- Los parciales tienen tres notas: I (Insuficiente): 0 a 59 pts, A- (Aprobado condicional): 60 a 64 pts y A (Aprobado): 65 a 100 pts. No se puede aprobar con A- ambos parciales. Los recuperatorios tienen dos notas: I: 0 a 64 pts y A: 65 a 100 pts.

Ej. 1. (40 puntos)

1 - (10 puntos) Describir cómo se completarían las entradas de la GDT en función de los segmentos que se detallan en la siguiente tabla. Indicar los campos `Indice`, `Base`, `Limite`, `S`, `G`, `D/B`, `AVL`, `L`, `P`, `DPL` y `Tipo`. Los valores de `Base` y `Limite` deben indicarse en hexadecimal.

Indice	Desde	Tamaño	Permisos	Tipo
2	2 Mb	4 Mb	`nivel 1`	Código - lectura
5	1 Gb	0,5 Gb	`nivel 0`	Datos - lectura/escritura
8	4 Kb	8 Kb	`nivel 3`	Código - solo ejecución
9	2 Gb	1 Gb	`nivel 2`	Datos - lectura

2 - (12 puntos) Especificar todas las entradas de las estructuras necesarias para construir un esquema de paginación particular según la siguiente tabla. Suponer que todas las entradas no mencionadas son nulas. Los rangos incluyen el último valor. Los permisos deben definirse como usuario.

Rango Lineal	Rango Físico
`0x0000A000` a `0x0000CFFF`	`0x4000F000` a `0x40011FFF`
`0xBABA0000` a `0xBABA0FFF`	`0x8CACA000` a `0x8CACAFFF`

3 - (15 puntos) Resolver las siguientes direcciones, de lógica a lineal y a física. Utilizar las estructuras definidas en los ítems anteriores y suponer que cualquier otra estructura no está definida. Si se produjera un error de protección, indicar cuál error y en qué unidad. Definir `EPL` en todos los casos.

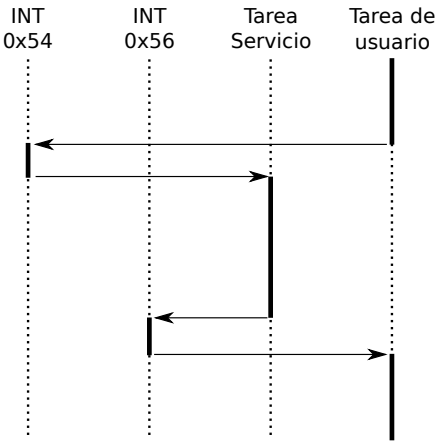
- a) `0x2B:0x0000A123` - CPL `00` - `lectura`
- b) `0x10:0xCACABABA` - CPL `11` - `lectura`
- c) `0x38:0xBABACACA` - CPL `00` - `escritura`
- d) `0x49:0x0000CFFE` - CPL `11` - `ejecución`
- e) `0x43:0x0000C000` - CPL `11` - `ejecución`

4 - (3 puntos) Suponga un sistema que tiene implementado segmentación sin solapamiento y sin paginación. Suponga además la siguiente afirmación: "*Es posible modificar código o ejecutar datos.*". Responder si la afirmación anterior es Verdadera o Falsa. Justifique indicando cómo sería posible o por qué sería imposible.

Nota: `1GB=0x40000000, 2GB=0x80000000, 3GB=0xC0000000, 4GB-1=0xFFFFFFFF, 1MB=0x100000, 4KB=0x1000`.

Ej. 2. (40 puntos)

En un sistema en dos niveles corren varias tareas de usuario concurrentemente durante intervalos fijos de tiempo. Las tareas acceden a los servicios que provee el sistema, por medio de la interrupción `0x54`. En este sistema, se llama servicio a un conjunto de tareas especiales que al ser ejecutadas proveen alguna funcionalidad. Las tareas-servicio retornan el control al sistema por medio de la interrupción `0x56`, fin de servicio. Los servicios se identifican por un `id` de servicio, y toman como parámetros una página de datos de 4kB que debe ser mapeada al área de memoria de la tarea-servicio, previo a ser esta ejecutada. El sistema ejecutará la tarea-servicio hasta que esta termine.



Se cuenta con las siguientes funciones,

- `gdt* get_tss_descriptor(id service)`:
  retorna el puntero al descriptor de `tss` de la tarea-servicio `service`.
- `tss* get_base_tss(gdt* gdt_descriptor)` :
  dado un descriptor de `tss` obtiene la base de la `tss`.
- `void* get_lineal_address(cr3 c, void* address)` :
  dado un `cr3` y una dirección virtual, retorna la dirección lineal.
- `void mapear_pagina(cr3 c, void* lineal, void* virtual)` :
  mapea la dirección lineal a la virtual en el `cr3` dado.
- `void desmapear_pagina(cr3 c, void* virtual)` :
  desmapea la dirección virtual en el `cr3` dado.
- `gdt* next_task()` :
  retorna el puntero al selector de la próxima tarea.

El mecanismo pedido debe respetar,

- En `eax` se pasa el `id` del servicio solicitado.
- En `ebx` se pasa la dirección virtual de la pagina de 4kB de párametro.
- La dirección de mapeo de la pagina de párametro en el servicio es `0x5312000`

1 - (9 puntos) Describa las entradas en la IDT de la interrupción `0x54`, `0x56` y reloj.

2 - (20 puntos) Escriba en ASM el handler de la interrupción `0x54`, `0x56` y reloj.

3 - (11 puntos) Escriba en C la función `void* get_lineal_address(cr3 c, void* address)`.

Ej. 3. (20 puntos)

En un sistema tipo en dos niveles con segmentación *flat* y paginación activa se desea implementar una extraña funcionalidad. El sistema cuenta con un *scheduler* que se encarga de ejecutar una a una las tareas durante un tiempo. Las tareas son intercambiadas desde una rutina que comienza ejecutando la instrucción *PUSHAD*, siendo esta la única que afecta la pila de la tarea a ser desalojada. Todos estos registros son salvados en la pila y reestablecidos al continuar la ejecución.

La funcionalidad a implementar asociada a la interrupción `0x77`, consiste en desarrollar una rutina lea un registro de una tarea en particular y lo retorne a la tarea llamadora.

Se cuenta con la siguiente función,

- `tss* dame_tss(id task)`: retorna el puntero a la `tss` de la tarea `task`.

La función pedida debe respetar,

- En `eax` se pasa el `id` de la tarea cual leer.
- En `bl` se pasa número del registro a leer ordenado desde 1 en adelante como `edi`, `esi`, `ebp`, `esp`, `ebx`, `edx`, `ecx` y `eax`.

1 - (20 puntos) Escriba el código en ASM de la interrupción `0x77`.

Recordar:

```
  Code | Mnemonic  | Description
-------+-----------+-------------------------------------------------------------
   60  | PUSHAD    | Push EAX, ECX, EDX, EBX, original ESP, EBP, ESI, and EDI
```