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

Organización del Computador 2
Recuperatorio del Primer Parcial – 28/06/18

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.

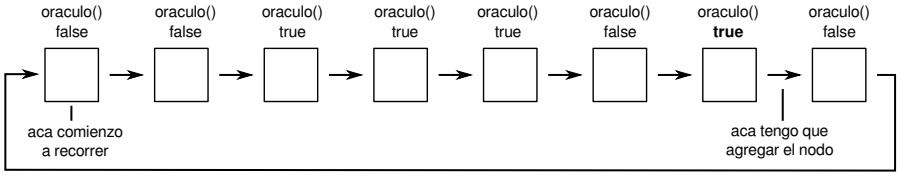
Ej. 1. (40 puntos)

Sea una lista circular que respeta la siguiente estructura:

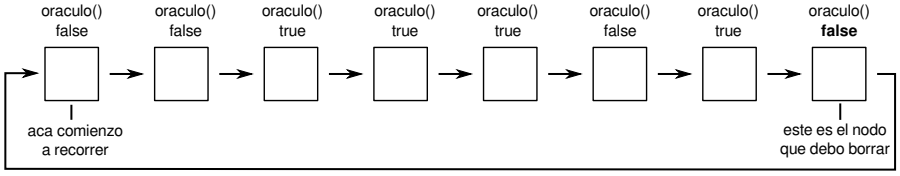
```
struct nodo {
  struct nodo* siguiente,
  int dato,
  bool (*oraculo)()
}
```

Donde `siguiente` es un puntero al siguiente nodo, `dato` es un valor entero almacenado y `oraculo` es un puntero a función que no recibe nada y devuelve un valor de tipo `bool`. `bool` es un entero de 4 bytes, que se interpreta como `false` si vale `0` y `true` en caso contrario.
Se pide escribir 2 funciones:

- `void insertarDespuesDelUltimoTrue(nodo* listaCircular, int nuevoDato, bool (*nuevoOraculo)())`:
  Inserta un nuevo nodo después del último nodo para el cual el llamado a su `oraculo` devuelva `true`. El nuevo nodo debe contener los campos `dato` y `oraculo` pasados por parámetro. Si el oráculo de ningún nodo devuelve `true`, no se debe insertar nada.



- `void borrarUltimoFalse(nodo** listaCircular)`:
  Borra el último nodo cuya llamada a `oraculo` devuelva `false`. De borrar el primero debe modificar el puntero a la lista circular. Si el oráculo de ningún nodo devuelve `false`, no se debe borrar nada.



(8p) a. Implementar la función `insertarDespuesDelUltimoTrue` en C.

(15p) b. Implementar la función `insertarDespuesDelUltimoTrue` en ASM.

(17p) c. Implementar la función `borrarUltimoFalse` en ASM.

Ej. 2. (40 puntos)

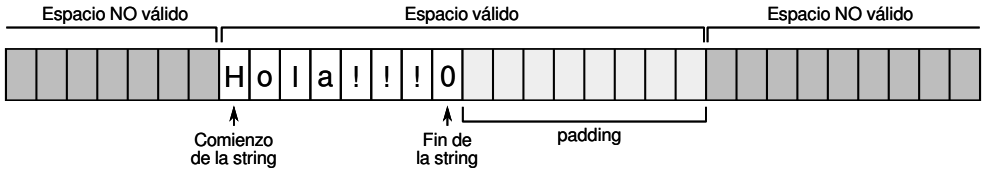
Un string de C es una cadena de caracteres de un byte codificados usando ASCII que termina en 0 (nulo). Por ejemplo, la cadena `Hola!!!` se codifica:

H	o	l	a	!	!	!	
0x48	0x6f	0x6c	0x61	0x21	0x21	0x21	0x00

Se pide implementar dos funciones:

- `double promedio(char* str)`:
  Calcula el promedio de los caracteres del string, sin considerar el caracter nulo.
  Por ejemplo, para la string anterior: $\frac{0x48+0x6f+0x6c+0x61+0x21+0x21+0x21}{7} = 69{,}57\ldots$.
- `void reemplazar(char* str, int longitud, char viejo, char nuevo)`:
  Toma un string y dos caracteres, y reemplaza cada ocurrencia del primer caracter por el segundo en el string. Por ejemplo, para la string anterior, reemplazamos cada ocurrencia de `a` por `b` queda `Holb!!!`. Si reemplazamos cada ocurrencia de `!` por `x`, queda `Holaxxx`. Si reemplazamos cada ocurrencia de `r` por `q`, el string queda igual.

Nota: La longitud de la string puede ser arbitrariamente grande, considerar que el área ocupada por la string es múltiplo de 16 bytes, es decir, se agrega *padding* al final de la misma que puede ser leído y modificado. Suponer a demas que este *padding* es inicialmente cero.



(20p) a. Implementar, usando SIMD, la función `promedio`.

(20p) b. Implementar, usando SIMD, la función `reemplazar`.

Ej. 3. (20 puntos)

Los desarrolladores del software que ejecuta un satelite están muy seguros que la radiación cósmica destruyo parte del procesador principal del mismo.

Este daño impide al procesador ejecutar instrucciones básicas, tales como `sub`, `or`, `call`, `loop`, `jmp` y ni ningún salto condicional.

Se pide desarrollar la siguiente función en ASM:

```
int masterLock( int (*connectionRelease)(int), int (*magneticToroid)(int), int value ) {
    int response = connectionRelease(value);
    response = response + magneticToroid(value);
    return (67 - response);
}
```

(5p) a. ¿Qué instrucciones de ASM permiten modificar el registro `Instruction Pointer` (`rip`)?

(15p) b. Escribir en ASM la función `masterLock`, considerando las restricciones del sistema.

Nota: Recordar que el inverso aditivo de un número en complemento a dos es: $notBitAbit(n) + 1$