Apellido y Nombres	Legajo	# de Hojas

Parte Teórica

Enumere los distintos tipos de implementación de librerías, disponibles para lenguaje C bajo Linux, explicitando para cada una su función y caso de uso.

Parte Práctica

Se dispone de un sistema de control de acceso para el cual se requieren implementar funciones de seguimiento y control de los ingresos y egresos de un establecimiento.

El programa recibe a través de la señal **SIGUSR1** una notificación de que se ha producido una acción de ingreso o egreso en una determinada puerta. Para conocer los detalles de ese acceso cada vez que se produce, se invoca a la siguiente función, que retorna una cadena de caracteres con el siguiente formato:

```
char* trama_acceso();
```

Ingreso ('I') o Egreso ('E') [1 byte]	Puerta (00 a 99) [2 bytes]	CUIT sin guiones [11 bytes]

La función trama_acceso() ya se encuentra implementada y no forma parte del exámen. Sólo debe utilizarla cuando sea necesario.

Se pide mantener un control de las secuencias de ingreso o egreso en una lista doblemente enlazada con la siguiente estructura.

```
typedef struct nodo_s {
      char accion;          // Ingreso 'I' o Egreso 'E'
      short int puerta;     // Numero de puerta
      char cuit[20];        // CUIT formateado del usuario
      time_t timestamp;     // Fecha y hora de la accion
      nodo_s * prev;
      nodo_s * next;
} nodo_t;
```

Se pide implementar las siguientes funciones:

```
nodo_t * nuevo_acceso(char * a);
```
Recibe la trama devuelta por la función **trama_acceso()** y devuelve un nuevo nodo

con los datos pre-cargados, incluyendo el timestamp, que debe calcularse en el momento.

`int registrar_acceso(nodo_t ** L, char * a);`
Recibe la trama devuelta por la función **trama_acceso** y agrega el nodo a la lista. Los nodos se agregan de forma secuencial, siempre al final.

`int analizar_accesos(nodo_t * n);`
Recibe el último nodo ingresado y recorre la lista en busca de anomalías. Si la acción fué un ingreso, debe haber un egreso inmediato anterior para el mismo CUIT, o nada, **nunca otro ingreso**. Si fué un egreso, debe haber sí o sí un ingreso previo, **nunca nada ni otro egreso inmediato anterior para el mismo CUIT**. Devuelve OK en si no hay problemas o ALARMA si hubo un error.
La función "registrar_acceso" debe llamar a "analizar_accesos" luego de agregar cada nodo, y devolver el resultado del análisis.

`int signal_acceso(int signum);`
Recibe la señal **SIGUSR1** y actúa en consecuencia. Tenga en cuenta que dentro del scope de esta función es el único espacio de memoria donde se tiene almacenado el comienzo de la lista.

`char * formatear_cuit(char *);`
Recibe un buffer con los 11 dígitos del CUIT y los devuelve formateado apropiadamente con los guiones (XX-12345678-X). El formateo debe realizarse dentro del mismo vector de caracteres (in-place), no en un nuevo espacio de memoria. Asuma que el buffer tiene el tamaño suficiente.

`void imprimir_acceso(nodo_t * n);`
Imprime un nodo cualquiera en pantalla con el siguiente formato de ejemplo:

[INGRESO] Puerta: 20, CUIT: 20-12345678-0, Fecha: 2017-11-27 08:00:00

Condición mínima de aprobación

- Un total de 4 (cuatro) funciones implementadas correctamente, entre las que debe estar incluida **registrar_acceso** y/o **analizar_accesos**.
- Cualquier error de índole conceptual, segmentation fault o memory leak invalida la solución en cuestión.

No forma parte del exámen

- No se pide reimplementar la rueda. **No se pide implementar la función trama_acceso().**
- Utilice funciones de la librería estándar del C siempre que sea posible, como time() o strftime() definidas en time.h, o cualquier función de string.h.