|  |  |  |
| --- | --- | --- |
| **Apellido y Nombres** | **Legajo** | **# de Hojas** |
|  |  |  |

# Parte Teórica

Dado la siguiente declaración de un array, y asumiendo que el sistema operativo de 32 bits nos asigna la dirección **0xbfff0000** para el comienzo del mismo:

**int v[5] = {32,12,15,89,6};**

1. ¿Qué dirección de memoria contendrá el valor de v[2] ?
2. ¿Qué dirección de memoria resultará de resolver v+3 ?
3. Indique cómo accedería al valor 89 dentro del vector, por cualquier método que
4. conozca.
5. Podemos recorrer el vector utilizando post-incremento (v++) ? Justifique su respuesta.
6. Recibiríamos algún error durante la compilación o el linkeo si tratáramos de acceder al contenido de v+10 ? Justifique su respuesta.

# Parte Práctica

Se recibe en los argumentos de línea de comandos una lista de strings a procesar. La cantidad de argumentos recibidos es variable y no se conoce previamente. **El nombre del programa debe ignorarse y en ningún momento forma parte de la lista de strings a procesar.**

Se pide implementar las siguientes funciones:

**int contar\_strings(char \*\*s);**

Recibe un vector de punteros a char, terminado en NULL, y cuenta la cantidad de strings en dicha lista.

**int es\_numerico(char \*s);**

Devuelve (int)1 si el string especificado es puramente numérico (sólo contiene dígitos del 0 al 9 y/o el caracter punto), devuelve (int)0 en caso contrario.

**int es\_email(char \*s);**

Devuelve (int) 1 si el string especificado es un email (contiene el caracter @ y al menos un carater “.” luego).

**int es\_tarjeta(char \*s);**

Devuelve (int) 1 si el string especificado tiene formato de número de tarjeta de crédito, devuelve (int) 0 en caso contrario. Se pide simplemente que valide los formatos, no que verifique que sea un número de tarjeta real. Los formatos soportados serán:

**4444-4444-4444-4444 (Visa/MasterCard)**

**4444-666666-55555 (American Express)**

**void ordenar\_strings(char \*\*s);**

Ordena los strings con el siguiente criterio:

* Primero números, de menor a mayor.
* Luego tarjetas de crédito, de forma ascendente.
* Luego direcciones de mail, ordenados alfabéticamente.
* Luego cualquier otro string que no sea numérico, email o tarjeta de crédito, ordenado alfabéticamente.

**void imprimir\_strings(char \*\*s);**

Al finalizar, el programa debe imprimir la lista de strings, agrupando los válidos y los no válidos de la siguiente forma:

**$ ./parcial 11.5 yo@gmail.com 353 4444-4444-4444-4444 abdc1 test**

**== Numeros ==**

**11.5**

**353**

**== Tarjetas ==**

**4444-4444-4444-4444**

**== Emails ==**

**yo@gmail.com**

**== No Validos ==**

**abcd11**

**test**

Notas:

* Recuerde que se debe verificar que por lo menos se le hayan pasado una cantidad mínima de strings en la línea de comandos.
* Implemente todas las funciones auxiliares que crea necesarias.
* Pueden implementar archivos de prueba para usar redirección de la línea de comandos $(<prueba.txt)

# Condiciones mínimas de aprobación

* 4 funciones implementadas correctamente, o bien 3 funciones y el punto teórico. Al menos una de las funciones debe ser **ordenar\_strings** o **imprimir\_strings**.
* Cualquier error de índole conceptual, segmentation fault o memory leak invalida la solución en cuestión.

# No forma parte del exámen

* No se pide reimplementar funcionalidad existente. **Utilice funciones de la librería estándar del C siempre que sea posible.**