Objetivo

La primera sesión servirá para que el alumno entre en contacto por primera vez (en esta asignatura) con el lenguaje C. En esta primera sesión se repasarán algunos tipos básicos (int, float, short, etc.) y compuestos (structs, arrays, etc.) de C. También se repasarán los punteros y el uso de ficheros. Finalmente, en esta sesión se introducirá la guía de estilos de la asignatura.

Motivación

Más concretamente, con esta sesión, el alumno ha de ejercitar:

- Tipos básicos (int, char, short, float, etc.).
- Creación y uso de nuevas estructuras (structs).
- Creación y uso de vectores estáticos y dinámicos.
- Uso de bucles y condicionales.
- Operaciones básicas de entrada y salida.
- Operaciones sobre ficheros de texto.
- Aplicación de la guía de estilos.

Documentación previa

Para completar esta sesión se recomienda la lectura de las siguientes referencias:

SALVADOR, J. (2009). *Introducció al llenguatge de programació C*.

Enunciado: LogsManager

La Salle, aprovechando el dominio de C de los nuevos alumnos de SO, ha aceptado un proyecto propuesto por una empresa de administración de sistemas.

La empresa está trabajando con 4 servidores con un nuevo sistema operativo que, en su versión beta, acumula cientos de errores, avisos y mensajes informativos recopilados en sus ficheros de *log*. Al final del día estos *logs* han de ser almacenados lógicamente para su posterior análisis.

Cada servidor consta de 1 fichero de *log* donde se van acumulando los diferentes mensajes del sistema.

Disponemos de 4 ficheros `SERVER1.dat`, `SERVER2.dat`, `SERVER3.dat` y `SERVER4.dat`.

Cada fichero está encabezado por un *header* formado por dos líneas: el identificador del servidor y la fecha correspondiente al día en el cual se han recopilado los mensajes. Por convenio, la fecha podrá estar introducida tanto en el formato `dd/mm/yyyy` como `yyyy-mm-dd`.

```
ID_SERVER                    [2 bytes]
Fecha creación               [dd/mm/yyyy ó yyyy-mm-dd]
```

Ejemplo:

```
1
03/09/2018
```

Cada uno de los ficheros de *log* contiene un número de líneas desconocido separadas por el carácter '\n'.

El formato de cada una de las líneas es el siguiente:

```
tipo[gravedad]/mensaje/proceso/hh:mm:ss
```

- **tipo [2 bytes]**: hay 4 tipos diferentes de *logs* en el sistema
  - 0: Aplicaciones
  - 1: Eventos
  - 2: Servicios
  - 3: Sistema
- **gravedad [2 Bytes]**: índica el grado de importancia del error registrado de más leve (0) a más grave (5).
- **mensaje [n Bytes]:** mensaje que aporta más información sobre el evento registrado. Este campo es optativo, puede estar vacío.
- **proceso [n Bytes]**: proceso fuente del evento o error.
- **hh:mm:ss [2 Bytes, 2Bytes, 2Bytes]**: tiempo exacto en el cual se ha registrado el evento o error en el fichero de *logs.*

Ejemplo:

```
2[4]/Kernel logging (proc) stopped/galileo kernel/08:30:05
```

El programa en C deberá ser capaz de procesar la información de estos ficheros y ofrecer las siguientes funcionalidades al usuario:

1. **Mostrar *logs* según tipo:** se introduce un número por consola indicando el tipo de log que se quiere visualizar. Se tendrán en cuenta los *logs* de los 4 ficheros.

2. **Mostrar *logs previos***: se introduce la fecha actual por consola (en formato `hh:mm:ss`) y se muestra los *logs* registrados a una hora anterior o igual a la introducida.

3. **Mostrar *logs* según la gravedad:** se mostrarán únicamente los *logs* con el índice de gravedad (de 0 a 5) introducido por consola.

4. **Salir del programa:** el programa liberará todos los recursos necesarios y se cerrará.

Ejemplo de ejecución

```
--------- Logs Manager ---------

Procesando fichero SERVER1.dat... Creado el día 25 de Ago de 2018.

Procesando fichero SERVER2.dat... Creado el día 25 de Ago de 2018.

Procesando fichero SERVER3.dat... Creado el día 25 de Ago de 2018.

Procesando fichero SERVER4.dat... Creado el día 25 de Ago de 2018.



----- Menu -----
1 - Mostrar logs segun el tipo
2 - Mostrar logs previos
3 - Mostrar logs segun la gravedad
4 - Salir del programa

Opcion: 1

Introduce el tipo: 2
*--
Tipo: 2
Gravedad: 4
Mensaje: secserv kernel: Kernel log daemon terminating.
Proceso: secserv asp
Hora: 20:04:14
--*

*--
Tipo: 2
Gravedad: 1
Mensaje:
Proceso: cloudcmd
Hora: 20:00:00
--*



----- Menu -----
1 - Mostrar logs segun el tipo
2 - Mostrar logs previos
3 - Mostrar logs segun la gravedad
4 - Salir del programa

Opcion: 2

Introduce tiempo (hh:mm:ss): 20:00:00
*--
Tipo: 0
Gravedad: 0
Mensaje: Card NEXTQSCB = 4.
Proceso: CARD INFO
```

```
Hora: 13:13:00
--*

*--
Tipo: 1
Gravedad: 1
Mensaje: STACK: 0xe1 0xe1 0x163 0x178
Proceso: new fish
Hora: 00:14:10
--*

*--
Tipo: 1
Gravedad: 4
Mensaje: scsi1: Dumping Card State in Message-out phase, at SEQADDR
0x16b
Proceso: scsi
Hora: 01:23:34
--*

*--
Tipo: 2
Gravedad: 1
Mensaje:
Proceso: cloudcmd
Hora: 20:00:00
--*

*--
Tipo: 3
Gravedad: 5
Mensaje:    0    SCB_CONTROL[0x0]    SCB_SCSIID[0x17]    SCB_LUN[0x0]
SCB_TAG[0xff].
Proceso: QUINFIFO
Hora: 09:07:58
--*

*--
Tipo: 3
Gravedad: 5
Mensaje: swapper: page allocation failure. order:0, mode:0x4020
Proceso: swapper
Hora: 12:13:59
--*



----- Menu -----
1 - Mostrar logs segun el tipo
2 - Mostrar logs previos
3 - Mostrar logs segun la gravedad
4 - Salir del programa

Opcion: 3

Introduce el índice de gravedad: 2
No hay resultados.
```

```
----- Menu -----
1 - Mostrar logs segun el tipo
2 - Mostrar logs previos
3 - Mostrar logs segun la gravedad
4 - Salir del programa

Opcion: 3

Introduce el índice de gravedad: 5
*--
Tipo: 3
Gravedad: 5
Mensaje:    0    SCB_CONTROL[0x0]    SCB_SCSIID[0x17]    SCB_LUN[0x0]
SCB_TAG[0xff].
Proceso: QUINFIFO
Hora: 09:07:58
--*

*--
Tipo: 3
Gravedad: 5
Mensaje: BUG: unable to handle kernel NULL pointer dereference at
(null).
Proceso: sys
Hora: 20:31:01
--*

*--
Tipo: 3
Gravedad: 5
Mensaje: swapper: page allocation failure. order:0, mode:0x4020
Proceso: swapper
Hora: 12:13:59
--*



----- Menu -----
1 - Mostrar logs segun el tipo
2 - Mostrar logs previos
3 - Mostrar logs segun la gravedad
4 - Salir del programa

Opcion: 4

Bye!
```

Consideraciones

- Se puede asumir que siempre habrá un mínimo de un log por fichero.
- El *output* del programa ha de ser **completamente idéntico** al que se muestra en el ejemplo de ejecución.
- El formato de hora es de 24 horas.
- Todos los ficheros se crean diariamente. Es decir, los 4 ficheros de entrada corresponden a la misma fecha.
- Para seleccionar una opción de menú se deberá apretar la tecla asociada seguida de la tecla Intro.
- En caso de elegir una opción inexistente de menú, se volverá a solicitar que se introduzca una opción válida.
- Solo se cerrará el programa en caso de introducir la opción 4
- Toda la información ha de ser cargada en un array de estructuras que **optimice el tamaño de dichas estructuras**, es decir, se han de seleccionar los tipos de datos adecuados que se ajusten mejor a los bytes y tipos de cada campo.
- **En el inicio del .c creado debe de haber comentado los logins y nombres y apellidos de los alumnos que realizan la sesión**.
- La sesión ha de seguir una estructura modular, por lo tanto, se debe separar el código en funciones siempre que sea posible y conveniente.
- **Se deberá estructurar el código en ".c" y ".h" y compilar usando un makefile**
- **Se ha de compilar utilizando los flags –Wall y –Wextra.**
- **Cualquier práctica que contenga warnings o errores será no apta**.
- Para la entrega de la sesión se deberá encapsular todo en un ".tar" con el nombre "S0_login1_login2.tar".