Ejercicios prácticos sobre Programación en C

Entrega 2

Programación Para Sistemas (PPS) 2022/23

Ejercicio 4.

Realice un programa (maxmin.c) que reciba como entrada el nombre de un fichero de texto que contiene un número real (float) por línea y que después de leerlo escriba por pantalla el valor máximo y el valor mínimo en una sola línea (formato: %10.2f %10.2f). Si el fichero está vacío debe escribir los valores 0.00 y 0.00. Suponer que los números del fichero son números reales válidos. Hay que comprobar que en la llamada se ha pasado un argumento de entrada, en caso contrario terminar sin imprimir nada y devolver al sistema operativo el número -1. Si el fichero no se puede leer también se devolverá el número -1 sin imprimir nada. En el resto de los casos, el número a devolver será el 0. Suponga que "fich.txt" contiene las siguientes 3 líneas:

10.2
12.55
15.0
Entonces, la salida de la llamada: *./maxmin fich.txt* deberán ser la siguiente línea:

15.00 10.20

Ejercicio 5.

En este ejercicio muestra el procesado de **argumentos en la línea de orden**, el manejo de **cadenas de caracteres** y de **ficheros tipo texto**. En concreto, el programa leerá un archivo de texto cuyo *path* se le suministra en la línea de orden o, en su defecto, de la entrada estándar. Dicho archivo contendrá, obligatoriamente, una **primera línea de cabecera** que define los campos de datos separados por un carácter ‘,’ (formato *csv, comma separated value*). La segunda y sucesivas líneas corresponde a los datos pudiendo existir líneas en blanco. En este caso, el programa las ignorará. Todos los datos se tratarán como caracteres. Finalmente, el programa mostrará en la salida estándar las líneas de datos etiquetadas de acuerdo a la cabecera. A continuación se muestra un ejemplo de entrada y del resultado que el programa debe producir:

```
ENTRADA (fichero o stdin)
=========================

title,album,duration,release,artist,type
Black Hole Sun,Superunknown,05:06,1994,Soundgarden,Rock
Smells Like Teen Spirit,Nevermind,05:01,1991,Nirvana,Grunge
Breed,Nevermind,03:03,1991,Nirvana,Grunge
Lithium,Nevermind,04:17,1991,Nirvana,Grunge
Once,Ten,03:51,1991,Pearl Jam,Grunge
Even Flow,Ten,04:53,1991,Pearl Jam,Grunge
Alive,Ten,05:40,1991,Pearl Jam,Grunge
Jeremy,Ten,05:18,1991,Pearl Jam,Grunge
Forty Six & 2,Aenima,06:04,1994,Tool,Metal Progresivo
Lateralus,Lateralus,09:24,2001,Tool,Metal Progresivo

SALIDA
======

title: Black Hole Sun; album: Superunknown; duration: 05:06; release: 1994; artist: Soundgarden; type: Rock
title: Smells Like Teen Spirit; album: Nevermind; duration: 05:01; release: 1991; artist: Nirvana; type: Grunge
title: Breed; album: Nevermind; duration: 03:03; release: 1991; artist: Nirvana; type: Grunge
title: Lithium; album: Nevermind; duration: 04:17; release: 1991; artist: Nirvana; type: Grunge
title: Once; album: Ten; duration: 03:51; release: 1991; artist: Pearl Jam; type: Grunge
title: Even Flow; album: Ten; duration: 04:53; release: 1991; artist: Pearl Jam; type: Grunge
title: Alive; album: Ten; duration: 05:40; release: 1991; artist: Pearl Jam; type: Grunge
title: Jeremy; album: Ten; duration: 05:18; release: 1991; artist: Pearl Jam; type: Grunge
title: Forty Six & 2; album: Aenima; duration: 06:04; release: 1994; artist: Tool; type: Metal Progresivo
title: Lateralus; album: Lateralus; duration: 09:24; release: 2001; artist: Tool; type: Metal Progresivo
```

Por último, se supone que la longitud máxima de las líneas en el fichero es de 2048 bytes y que el número máximo de campos es 15. En todo caso, esto debiera definirse a través de constantes de manera que se pudieran cambiar los límites sin tener que modificar código.

El programa se codificará en tres archivos, dos fuentes .c (***main.c*** y ***parser.c***) y una cabecera .h (***parser.h***). Este último se suministra como parte del material de apoyo. Únicamente se entregará el fichero ***parser.c*** que incluirá la función ***parser()*** responsable del procesado del fichero y de la escritura del resultado. La función devuelve 0 si no se ha detectado ningún error y -1 en caso contrario. Los mensajes que informen de los errores como, por ejemplo, que el número de campos no es correcto deberán escribirse en el descriptor de error estándar, nunca en el de la salida estándar.
El programa principal deberá controlar que los argumentos son los correctos, cerrar los descriptores de ficheros abiertos y todo lo que sea pertinente.

Ejercicio 6.

Escribe un programa (**matrizdinamica.c**) que cree una matriz de *m filas x p columnas*, donde los valores de m y p serán dos enteros proporcionados como argumentos del programa. Se puede asumir que siempre se ejecuta el programa con sus 2 argumentos y que son dos números enteros válidos. Los elementos de la matriz (a_{ij}) serán de tipo *long int* y se calculan con las siguientes reglas:

- Fila 1 todos los elementos serán 1´s ($a_{1j}=1$ j=1….p)
- Columna 1 todos los elementos serán 1´s ($a_{i1}=1$ i=1….m)
- Restos de elementos $a_{ij}= a_{i-1j} + a_{ij-1}$ i=2….m j=2….p Si al calcular un elemento, se cumple que $a_{ij} >1.e6$ entonces $a_{ij} =1$

Una vez creada la matriz hay que escribirla en la salida estándar en formato matricial como se muestra en el ejemplo. Utilizar para imprimir cada elemento el formato *"%li\t"*. Una línea por fila, que acabará siempre con \n. Si no se pudo crear la matriz por problemas de memoria, el programa acabará sin mostrar nada y devolviendo el código 71. Si no hay ningún problema en la ejecución, devolverá el

código 0. Al finalizar se debe liberar toda la memoria dinámica solicitada antes de finalizar el programa. Suponiendo que nuestro programa se llama matrizdinamica, la ejecución: ./matrizdinamica 4 5 produciría la siguiente salida:

```
1     1     1     1     1
1     2     3     4     5
1     3     6     10    15
1     4     10    20    35
```

El código debe funcionar correctamente con valores grandes de m y p. El sistema de entrega hará comprobaciones de la correcta ejecución de vuestro código para valores de m y p grandes.

Opciones de compilación.

Todos los programas deben compilarse con las opciones `-ansi -pedantic -Wall -Wextra -Werror`.