



SISTEMAS OPERATIVOS

Practica 1: Llamadas al sistema operativo

Carlos Contreras Sanz 100303562
Álvaro Gómez Ramos 100307009

Índice

1. mywc.....	2
• Descripción	2
• Pruebas	2
• Comentario	2
2. myenv	3
• Descripción	3
• Pruebas	3
• Comentario	4
3. myishere.....	5
• Descripción	5
• Pruebas	5
• Comentario	6
4. Conclusiones y comentarios	6

1. mywc

- **Descripción**

En este ejercicio se nos pedía desarrollar un contador para bytes, palabras y líneas, según los criterios proporcionados en la práctica.

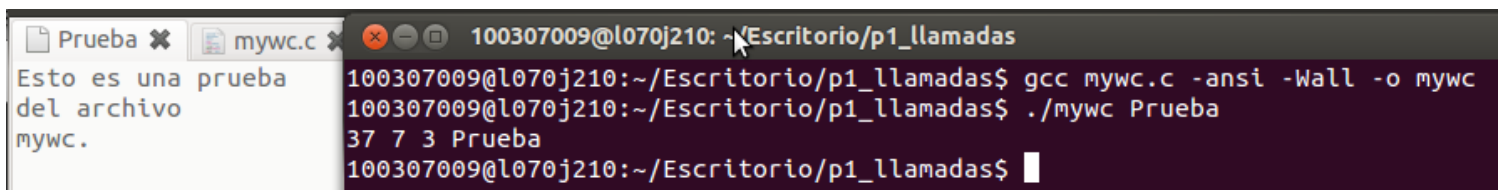
Lo primero que encontramos (después de los include) en el código es la verificación de que el numero de argumentos pasados es el adecuado, de no serlo, se abortara la ejecución, devolviendo -1. Después nos creamos las variables que usaremos, esto es: el char para uso de buffer (aux), el int que usamos para comprobación de que se lee bien, y condición de parada (leído) y los contadores para los resultados (carácter, palabra, línea).

Tras eso abrimos el fichero, con solo lectura y guardando su descriptor, que usaremos antes de nada para comprobar si ha habido algún problema en la apertura, en cuyo caso abortaremos la ejecución, devolviendo -1 y cerrando el archivo. Después entramos en el bucle en que iremos leyendo carácter a carácter hasta llegar al final del archivo (leído==0). Se sumará a los contadores según los criterio especificados en el anunciado de la práctica. Hemos usado un switch que mira que ha sucedido con el read, y si debemos de contar, entra a otro switch anidado, comparando el aux, para ver que debemos contar. En cualquier caso, al entrar en este switch, se contará un carácter, ya que es común a todos.

Por ultimo, se imprime lo que nos indica, se cierra el archivo y se devuelve 0;

- **Pruebas**

Para las pruebas de este ejercicio, se ha de tener en cuenta lo indicado por los profesores de practicas y la descripción de la misma, es decir, que se sumará carácter siempre, y con espacios y tabulaciones además se suma palabra, y con salto de linea, se suma además de palabra y carácter, una linea. Por lo tanto las pruebas se limitan a varias palabras, líneas y lo anteriormente mencionado, nunca teniendo espacio, tabulación y/o salto de linea consecutivos. Además, como se nos ha indicado, debe de existir un carácter invisible en el archivo de texto, ya que cuenta un carácter de mas.



```
100307009@l070j210: ~/Escritorio/p1_llamadas
100307009@l070j210:~/Escritorio/p1_llamadas$ gcc mywc.c -ansi -Wall -o mywc
100307009@l070j210:~/Escritorio/p1_llamadas$ ./mywc Prueba
37 7 3 Prueba
100307009@l070j210:~/Escritorio/p1_llamadas$
```

- **Comentario**

Como comentario, decir que se realizan las comprobaciones pertinentes (en el open y cada vez que se realiza un read) y se respeta en formato pedido. En caso de error (en lectura o apertura del archivo) no se muestra nada por pantalla, solo se retorna -1.

Por ultimo en este ejercicio, cuando se nos dice que se debe mostrar el nombre, mostramos el nombre con el que han llamado al programa, es decir, el que pasan por parámetros, ya sea ruta absoluta o relativa.

2. myenv

- **Descripción**

En este ejercicio se nos pedía desarrollar un programa que copiase todos los caracteres de una estructura determinada (environ) a un archivo de texto en un directorio determinado, estando cada carácter en una línea.

Lo primero tenemos (después de los include) es la comprobación de que el número de argumentos pasados es correcto. Inmediatamente realizamos el open del archivo (como lectura solo, y lo creamos si no existiese), y comprobamos que todo ha ido bien, si no abortamos. Luego declaramos las variables que usaremos, que son las del char salto de línea, y las de los contadores para recorrer environ.

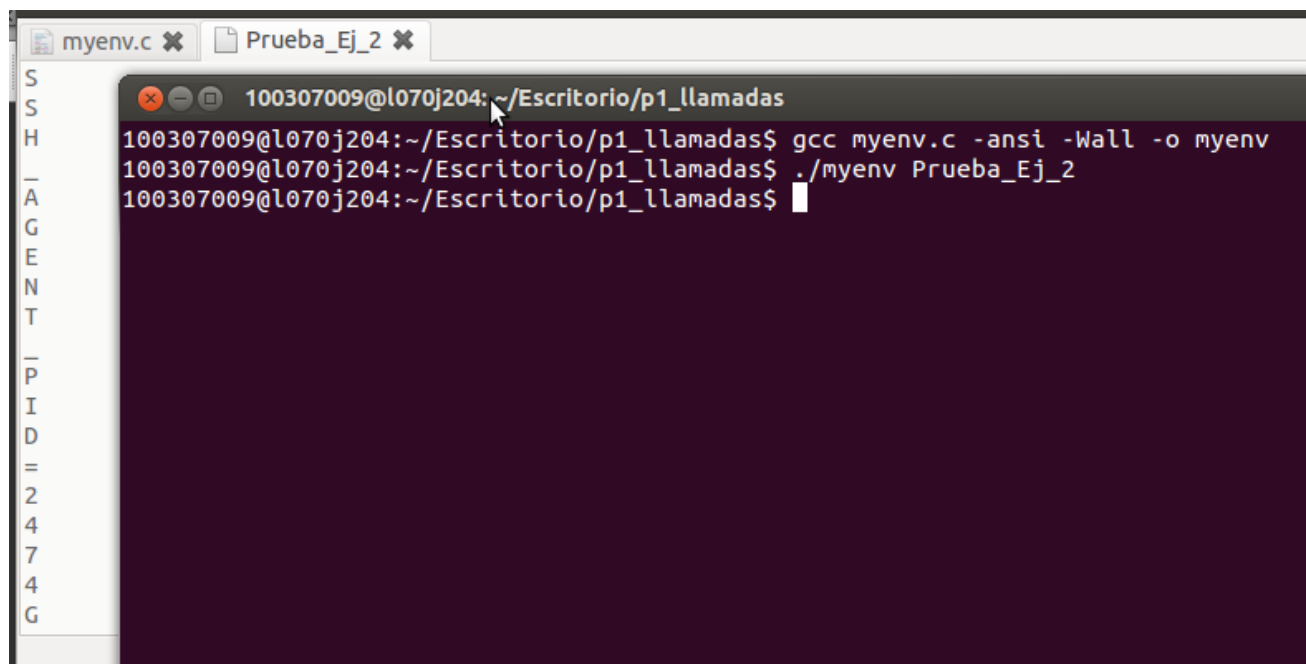
Entramos entonces en dos bucles while anidados, que primero realizan la escritura de la posición en la que estamos de environ (al ser matriz no es necesario acceder con read, y se puede hacer directamente), y después la escritura del salto de línea (el char antes declarado). Hemos decidido no incluir comprobación de que los write no dan -1 cada vez que se ejecutan, ya que al escribir de un byte en un byte, no se dará el caso en que se escriba de menos, y el error de que no hay permisos, si no nos ha saltado antes, no lo hará ahora. De modo que ara ahorrar pasos, que creemos no son necesarios, hemos decidido obviar esa comprobación dentro del while, y evitar cálculos innecesarios.

Después de todo se devuelve 0.

- **Pruebas**

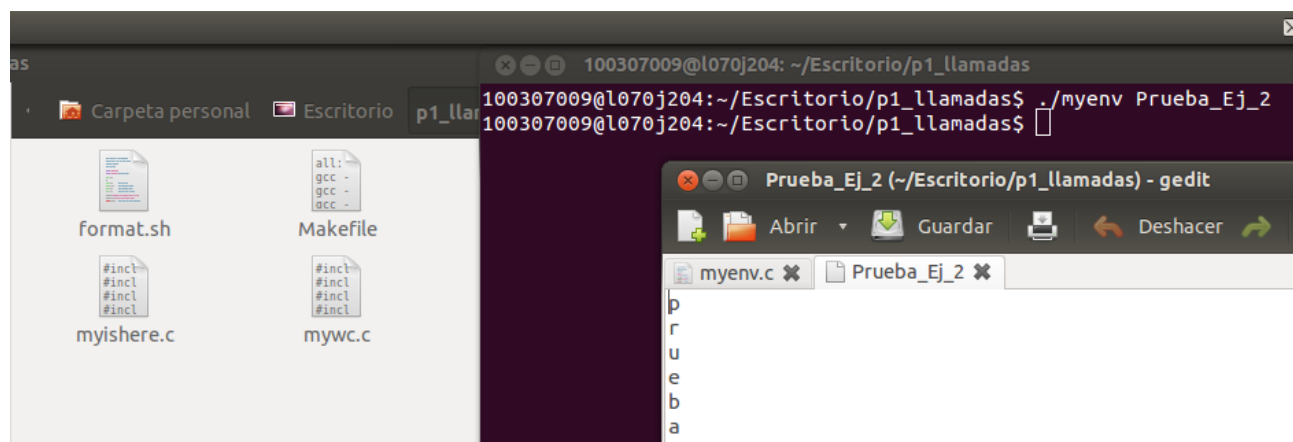
Las pruebas que se nos han ocurrido con este ejercicio, han ido desde probar con un archivo que si existía y vacío, con otro que estaba con otros datos (en cuyo caso borra los que había, o eso nos ha parecido), con uno que no existía, y creándonos nosotros una matriz environ en el código del ejercicio, y viendo que efectivamente copiaba los valores con la que lo habíamos declarado, al archivo que pasábamos por parámetros.

Aquí la prueba de funcionamiento estándar.



```
myenv.c x Prueba_Ej_2 x
100307009@l070j204: ~/Escritorio/p1_llamadas
100307009@l070j204:~/Escritorio/p1_llamadas$ gcc myenv.c -ansi -Wall -o myenv
100307009@l070j204:~/Escritorio/p1_llamadas$ ./myenv Prueba_Ej_2
100307009@l070j204:~/Escritorio/p1_llamadas$
```

Y aquí una en la que no existía el archivo, y creamos un environ para comprobar funcionamiento (dos pruebas en uno).



• Comentario

Comentar en este ejercicio (a pesar de haberlo dicho, lo repetimos) hemos decidido no incluir comprobación en los write, al considerarlo no necesario. Además señalar que tuvimos algún problema al desarrollarlo, ya que empezamos intentando acceder a environ con un read, cuando no es un archivo (error muy obvio, pero que nos llevo un rato ver), y también tuvimos problemas con la sentencia write, ya que el ultimo parámetro que se le pasa, lo poníamos como environ[ii][jj], y no como 1 (o strlen(envIRON[ii][jj])).

Decir además que en caso de error no se muestra nada por pantalla (error en el open por ejemplo), y que no creemos que sea necesario adjuntar mas imágenes de

pruebas, ya que con ellas no se refleja de manera clara cada caso, y documentar cada paso de cada prueba con ellas, seria alargar demasiado este documento.

3. myishere

- **Descripción**

En este ejercicio se nos pedía desarrollar un programa que buscase en un directorio determinado, un archivo.

Para ello lo primero que hacemos (tras los include) es comprobar que el número de argumentos que nos dan es el adecuado. Tras eso, abrimos el directorio que nos pasan, que guardamos en una variable de tipo puntero a DIR, y comprobamos que no haya habido error al abrirlo (es decir, que es un directorio valido, etc). Después creamos una variable dirent, que almacenará los datos de cada uno de los archivos que vayamos leyendo del directorio, con la función readdir.

Haciendo uso de una variable que nos indique cuando hemos encontrado el archivo (encontrado==0) entramos en dos bucles while anidados. Lo que hacen es, mientras que no hayamos encontrado el archivo, o mientras que no hayamos llegado al final del directorio sin encontrarlo (comparar encontrado con 0 y con null), lo que hacemos es pasar al siguiente archivo con readdir.

Encontrado, en el interior del while, se iguala al resultado de la comprobación de igualdad entre la cadena que nos pasan como nombre del archivo a buscar, con el nombre que figura en el dirent del archivo leído en ese momento, que solo será 0 en caso de ser iguales las cadenas, con lo que entraríamos en el if, imprimiríamos lo que nos dicen en ese caso, cerraríamos el directorio y saldríamos devolviendo 0.

Si no se entra en ese if, al terminar el while (porque se hayan leído todos los archivos y no se haya encontrado el deseado), se imprime que no estaba en ese directorio, se cierra y se sale devolviendo 0.

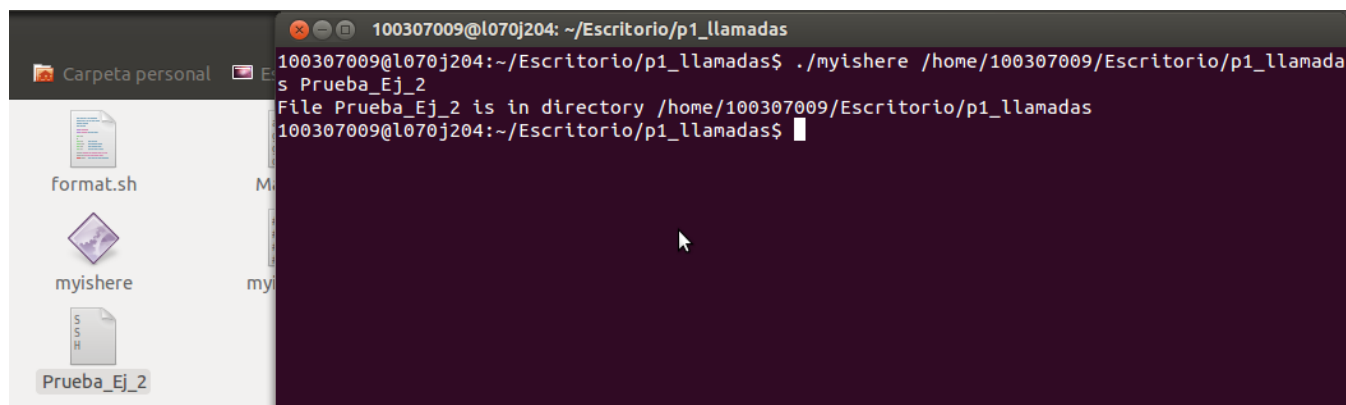
- **Pruebas**

Las pruebas de este ejercicio han consistido en:

Buscar en directorios que no existen (no devuelve nada por pantalla, solo -1).

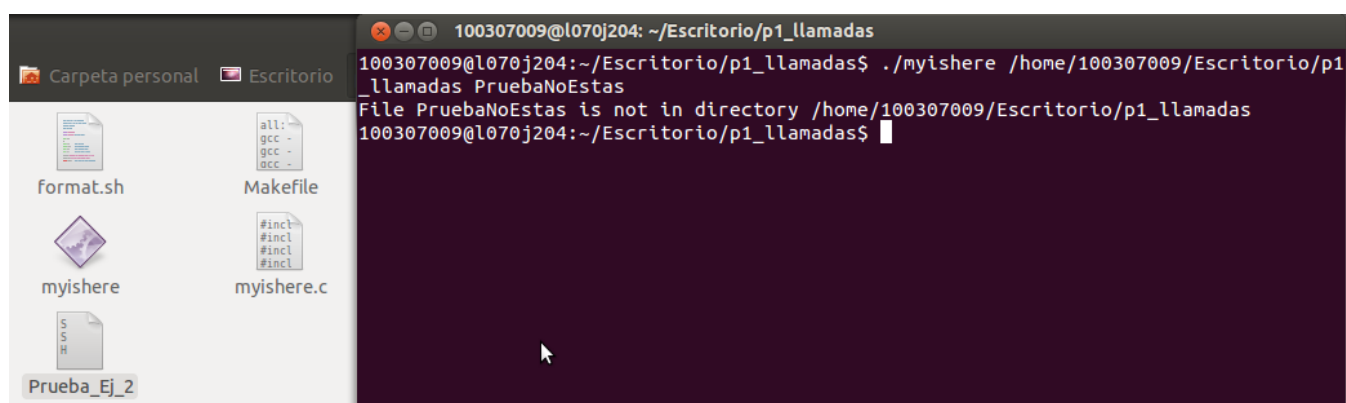
```
100307009@l070j204: ~/Escritorio/p1_llamadas
100307009@l070j204:~/Escritorio/p1_llamadas$ gcc myishere.c -ansi -Wall -o myishere
100307009@l070j204:~/Escritorio/p1_llamadas$ ./myishere /home/noExisteNoNo SiSiTuBusca
100307009@l070j204:~/Escritorio/p1_llamadas$
```

Buscar algo que si este en el directorio, dando una respuesta por pantalla.



The image shows a file manager window on the left with a sidebar containing 'Carpeta personal' and 'Escritorio'. The main pane displays files: 'format.sh', 'myishere', and 'Prueba_Ej_2'. The 'Prueba_Ej_2' file is selected. On the right, a terminal window titled '100307009@l070j204: ~/Escritorio/p1_llamadas' shows the command `./myishere /home/100307009/Escritorio/p1_llamadas Prueba_Ej_2` being executed. The output is `File Prueba_Ej_2 is in directory /home/100307009/Escritorio/p1_llamadas`.

Buscar algo que no este en el directorio, dando una respuesta por pantalla.



The image shows a file manager window on the left with a sidebar containing 'Carpeta personal' and 'Escritorio'. The main pane displays files: 'format.sh', 'myishere', 'Prueba_Ej_2', 'Makefile', and 'myishere.c'. The 'Prueba_Ej_2' file is selected. On the right, a terminal window titled '100307009@l070j204: ~/Escritorio/p1_llamadas' shows the command `./myishere /home/100307009/Escritorio/p1_llamadas PruebaNoEstas` being executed. The output is `File PruebaNoEstas is not in directory /home/100307009/Escritorio/p1_llamadas`.

- **Comentario**

En este ejercicio debemos señalar que se busca en el directorio que nos pasan, pero no en los sub directorios, asi, por ejemplo, si me pasan /home/ solo buscara en /home/, y no en /home/desktop.

4. Conclusiones y comentarios

Creemos que esta practica es bastante útil, o al menos así nos ha resultado a nosotros, ya que ayuda a asentar los conocimientos dados en clase, que si no se aplican y usan no se asientan (y sobre todo, si no nos obligan a usarlo por nuestra cuenta y a pelearnos con ellos).

Se agradece además la guía de los profesores, y del propio guion de prácticas.

Es además útil para seguir familiarizándonos con el entorno de Linux y la programación en C, asi com con los punteros y sus peculiaridades.