Control Laboratorio 2 (grupo 43) SO 16-17q2

Makefile (0.5 puntos)

Crea un **Makefile** que permita generar todos los programas del enunciado a la vez y cada uno de ellos por separado. Añade una regla (***clean***) para borrar todos los binarios y/o ficheros objeto, y dejar sólo los ficheros fuente. Los programas deben generarse si, y solo si, ha habido cambios en los ficheros fuente.

Control de errores (0.5 puntos)

Para todos los programas que se piden a continuación deben comprobarse los errores de todas las llamadas a sistema, controlar los argumentos de entrada y definir la función ***Usage()***.

Pares.c (1,5 puntos)

Escribe un programa en C, llámalo **pares.c**, que reciba por la entrada estándar enteros en formato **int** y que por la salida estándar muestre los que sean pares en formato texto y separados por un retorno de carro. Para probarlo dispones en el lind del Racó de un fichero **entrada** con enteros en formato **int**.

Cuenta_pares.c (3 puntos)

Escribe un programa llamado **cuenta_pares.c**. Este programa recibe como parámetro un fichero de enteros en formato **int** y como resultado muestra por la salida estándar una frase indicando cuántos pares hay en el fichero:

```
$> cuenta_pares fichero_de_enteros

En fichero_de_enteros hay XXX pares
```

Para ello **cuenta_pares.c** creará dos procesos hijo:

- Uno que mutará a **pares** (del ejercicio anterior), este hijo leerá del fichero pasado como parámetro al padre y volcará su salida estándar (pares en formato texto separados por retorno de carro) en una pipe anónima. Dispones en el link del Racó de un binario de **pares**, por si el tuyo no estuviera completo.
- El otro hijo mutará a **wc** (ver manual y sus opciones) y leerá de la pipe anterior y contará líneas. Asumiremos que el número de líneas leídas es igual al número de pares.

La frase resultado la escribirá el padre a partir del resultado de **wc**. Este resultado se enviará al padre mediante una pipe con nombre llamada **wc2padre**. Ésta pipe la debes crear mediante comandos de Shell.

Pares_v2.c (2 puntos)

Modifica el programa **pares.c** y llámalo **pares_v2.c** para que **además** escriba los pares encontrados en formato **int** en un fichero que llamareis **pares.bin** (no debe haber ningún tipo de separador entre los enteros que escribas en el fichero). Si el fichero no existe deberéis crearlo con permisos de lectura y escritura para ti y tu grupo, y permisos solo de lectura para el resto. Si el fichero ya existe previamente muestra, antes de continuar, un mensaje por el canal de error indicando que el fichero **pares.bin** se sobreescribirá.

Control Laboratorio 2 (grupo 43) SO 16-17q2

N_pares_directos.c (1,5 puntos)

Crea un programa llamado **N_pares_directos**, que a partir del fichero **pares.bin** generado por pares_v2.c indique cuántos números (pares) hay en el fichero, con las siguientes consideraciones:

- Mensaje de salida: `En pares.bin hay XXX números pares.`
- El fichero **pares.bin** no será parámetro de **n_pares_directos**
- Si **pares.bin** no existe el programa debe indicar, por el canal de error, este hecho con un mensaje dedicado y acabar. Cualquier otro error al acceder a pares.bin se debe tratarse como un error normal y acabar.
- Se valorará sobre todo la eficiencia de este programa en cuanto al acceso a disco en número de operaciones de lectura, escritura y posicionamiento.

Preguntas (1 punto = 0.25 + 0.25 + 0.5):

Responde en el fichero respuestas.txt:

1. Indica la línea de comandos utilizada para crear la pipe **wc2padre**

2. Crea un hardlink a pares.bin en el directorio actual. Llámalo **pares.hl**. Indica la línea de comandos utilizada.

3. Utilizando comandos del sistema, demuestra fehacientemente que pares.bin y pares.hl son realmente hardlinks al mismo fichero. Indica las líneas de comandos utilizadas, sus resultados y explica la razón por la que son hardlinks.

Qué hay que hacer

- El Makefile
- Los códigos de los programas en C
- La función Usage() para cada programa
- El fichero respuestas.txt con todas las respuestas a las preguntas

Qué se valora

- Que sigas las especificaciones del enunciado
- Que el uso de las llamadas al sistema sea el correcto
- Que se comprueben los errores de **todas** las llamadas al sistema
- Que el código sea claro y correctamente indentado
- Que el Makefile tenga bien definidas las dependencias y objetivos
- Que la función Usage() muestre por pantalla como debe invocarse correctamente el programa en el caso que los argumentos recibidos no sean los adecuados
- El fichero respuestas.txt

Qué hay que entregar

Un único fichero tar.gz con el código de todos los programas, el Makefile, y las respuestas en respuestas.txt:

```
tar zcvf clab2.tar.gz Makefile respuestas.txt *.c
```