CC3301 Programación de software de sistemas – Tarea Recuperativa – Otoño 2024
Profesores Mateu/Ibarra/Urrea

En esta tarea Ud. deberá programar el comando *masrecientes* que recibe como parámetros el nombre de un directorio *dir* y un entero *n*. El comando debe recorrer *dir* en profundidad y mostrar en su salida estándar los *n* archivos modificados más recientemente. El siguiente es un ejemplo de uso:

```
$ ./masrecientes.bin dir1 3
dir1/igualesB/dir2/iguales/cardinales.txt
dir1/noexiste2/hola.txt
dir1/tamanos-distintos/cardinales.txt
```

Solo debe mostrar archivos regulares. No debe mostrar *dir1/tamanos-distintos* porque es un directorio, pero sí debe mostrar:

```
dir1/tamanos-distintos/cardinales.txt
```

porque ese sí es un archivo regular.

Metodología obligatoria

Para recorrer en profundidad el directorio, base su solución en el programa *list-dir.c* que viene en los archivos adjuntos y que lista recursivamente los archivos y directorios a partir del parámetro recibido. Se compila e invoca con:

```
$ make PROB=list-dir list-dir.bin
$ ./list-dir.bin dir1
dir1/iguales
dir1/iguales/cardinales.txt
dir1/dir-inconsistente
…
```

Le serán de mucha utilidad la cola *Queue* y la función *sortPtrArray* programadas en *pss.c*. Estudie los encabezados de las funciones en *pss.h*. Hay un ejemplo del uso de *sortPtrArray* en *ejemplo-sort.c*. Se compila y ejecuta con:

```
$ make PROB=ejemplo-sort ejemplo-sort.bin
$ ./ejemplo-sort.bin
…
```

Declare una cola global. Al recorrer recursivamente el directorio, por cada archivo regular encontrado agregue a la cola un puntero a una estructura creada con *malloc* que incluya el nombre del archivo y su fecha de modificación. Esa fecha está en el campo *st_mtime* de la estructura obtenida con *stat*.

Al terminar calcule el tamaño de la cola con la función *queueLength*. Esa es la cantidad de archivos regulares encontrados. Transfiera las estructuras en la cola a un arreglo del mismo tamaño que la cola y ordene el arreglo con la función *sortPtrArray*. Deberá definir una función para establecer que el criterio de ordenamiento es la fecha de modificación. Finalmente recorra los *n* primeros elementos del arreglo ordenado, mostrando los nombres en la salida estándar.

Cuidado: el programa *list-dir.c* pide memoria con *malloc* para los nombres de los archivos encontrados y después libera esa memoria. Si agrega esos mismos strings a la cola, al finalizar será referencias colgantes. Deposite en la cola una copia de esos strings obtenida mediante la función *strdup*. No se preocupe por la eficiencia, pues no se exige en esta tarea. Recuerde que Ud. debe liberar la memoria que *sanitize* diagnostique como gotera de memoria.

Instrucciones

Descargue *trec.zip* de U-cursos y descomprímalo **en un directorio del sistema de archivos de Debian**. Su programa no aprobará los tests si descomprime *trec.zip* en un directorio del sistema de archivos de Windows. Ejecute el comando *make* sin parámetros en el directorio *TRec* para recibir instrucciones acerca del archivo en donde debe programar su solución (*masrecientes.c*), cómo compilar, probar y **depurar** su solución, y los requisitos que debe cumplir para aprobar la tarea.

Entrega

Ud. solo debe entregar por medio de U-cursos el archivo *masrecientes.zip* generado por el comando *make zip*. Este incluye los archivos *masrecientes.c* y *resultados.txt*. A continuación es muy importante que descargue de U-cursos el mismo archivo que subió, luego descargue nuevamente los archivos adjuntos y vuelva a probar la tarea tal cual como la entregó. Esto es para evitar que Ud. reciba un 1.0 en su tarea porque entregó los archivos equivocados. Se descontará medio punto por día de atraso. No se consideran los días de receso, sábado, domingo o festivos.