SISTEMAS OPERATIVOS - PRÁCTICAS
13 de enero de 2020

Nombre _____ DNI _____

Apellidos _____ Grupo _____ Puesto _____

Entregable 1.

Apartado 1.A) Implementar un código en C que utilizando la librería de POSIX Threads cree en su función *main* (ver código adjunto):

- N hilos que ejecuten una función *fun1*.
- M hilos que ejecuten la función *fun2*.

La función *fun1* recibe como argumento de entrada un entero (*arg1*) cuyo valor se pasará en la creación del hilo, y mostrará por la salida estándar el mensaje "Hilo tipo 1 con id *id_hilo.* Mi argumento de entrada es *arg1".* **Sugerencia**: si se usa un bucle para crear los hilos, podemos por ejemplo pasar como *arg1* el iterador del bucle + 1.

La función *fun2* no recibe argumentos de entrada. Mostrará por la salida estándar el mensaje "Hilo tipo 2 con id *id_hilo"*.

La función main, después de la creación de los hilos deberá esperar a la terminación de todos ellos y devolverá al sistema el valor de la variable global *suma*, que estará inicializada a 0 y no será modificada por ninguno de los hilos en este apartado.

Apartado 1.B) Modificar la función *fun1* del apartado 1.A para que, además de lo que hacían antes, los hilos que la ejecuten incrementen de forma segura la variable global *nproducciones* (cada hilo sumará 1, la variable global estará inicializada a 0) y muestren por la salida estándar el mensaje "El número total de producciones es *nproducciones*".

Apartado 1.C) Utilizando variables condicionales, modificar el comportamiento de los hilos que ejecutan *fun1* y *fun2* usando un patrón de tipo productores - consumidores sobre un buffer global *Buffer* (de tipo circular) de acuerdo al siguiente patrón:

1. Los hilos *fun1* son los productores, que mientras haya sitio en *Buffer* y el número total de producciones (*nproducciones*) no haya alcanzado el valor MAXITEMS:
    a. Insertarán un entero en el buffer con el valor que han recibido como parámetro (*arg1*).
    b. Mostrarán por la salida de error estándar "Productor *id_hilo*, he añadido un item *valor*. El número total de producciones es *nproducciones*"
    c. Esperarán un número aleatorio de segundos entre 1 y 5

    Finalizarán cuando *nproducciones* alcance el valor MAXITEMS.
2. Los hilos *fun2* son consumidores, que mientras haya elementos en el buffer y el número total de extracciones (*nextracciones*) no haya alcanzado el valor MAXITEMS:
    a. Extraerán un elemento de *Buffer* siguiendo el orden de producción.
    b. Actualizarán la variable global *suma* sumando el valor extraído
    c. Mostrarán por la salida de error estándar el mensaje "Consumidor *id_hilo*, valor extraído *valor,* acumulado *suma,* extracciones *nextracciones*".

    Finalizarán cuando *nextracciones* alcance el valor MAXITEMS.

El hilo main, esperará a la terminación de los hilos productores y consumidores y al concluir mostrará el valor devuelto por cada uno de los hilos consumidores

Apartado	Visto	Comentarios Profesor	Firma Profesor
1.A			
1.B			
1.C			

Entregable 2.

Realizar una modificación del programa mytar de la práctica 1 de tal forma que los nombres de fichero en el archivo mtar no sean cadenas acabadas en '\0', sino que se representen mediante pares (longitud,secuencia de caracteres sin '\0'). Por simplicidad en la gestión se redefinirá la estructura `stHeaderEntry` (*mytar.h)* como sigue:

```
typedef struct {
  unsigned int len_name;
  char* name;
  unsigned int size;
} stHeaderEntry;
```

Apartado 2.A) Demostrar que el programa mytar crea archivos mtar correctamente según el nuevo formato de archivo (puntos 1-3 del ejemplo de ejecución mostrado a continuación)
Apartado 2.B) Demostrar que el programa mytar extrae correctamente los archivos mtar que siguen el nuevo formato de archivo (punto 4 del ejemplo de ejecución)

Ejemplo de ejecución

```
# 1. Compilar el programa y Crear ficheros de texto de prueba
osuser@debian:~/Mytar$ make
gcc -g -Wall  -c  mytar.c -o mytar.o
gcc -g -Wall  -c  mytar_routines.c -o mytar_routines.o
gcc -g -Wall  -o mytar mytar.o mytar_routines.o
osuser@debian:~/Mytar$ echo Hello > a.txt
osuser@debian:~/Mytar$ echo Goodbye > ab.txt
osuser@debian:~/Mytar$ echo Heeeeeey > abc.txt
osuser@debian:~/Mytar$ echo This is a longer string > abcd.txt
osuser@debian:~/Mytar$ du -b *.txt
24  abcd.txt
9   abc.txt
8   ab.txt
6   a.txt
# 2. Creación de fichero mtar
osuser@debian:~/Mytar$ ./mytar -c -f test.mtar a.txt ab.txt abc.txt abcd.txt
Mtar file created successfuly
# 3. Comprobar si estructura es correcta
osuser@debian:~/Mytar$ xxd  test.mtar
00000000: 0400 0000 0500 0000 612e 7478 7406 0000  ........a.txt...
00000010: 0006 0000 0061 622e 7478 7408 0000 0007  .....ab.txt.....
00000020: 0000 0061 6263 2e74 7874 0900 0000 0800  ...abc.txt......
00000030: 0000 6162 6364 2e74 7874 1800 0000 4865  ..abcd.txt....He
00000040: 6c6c 6f0a 476f 6f64 6279 650a 4865 6565  llo.Goodbye.Heee
00000050: 6565 6579 0a54 6869 7320 6973 2061 206c  eeey.This is a l
00000060: 6f6e 6765 7220 7374 7269 6e67 0a         onger string.
# 4. Verificar si extracción correcta
osuser@debian:~/Mytar$ mkdir tmp
osuser@debian:~/Mytar$ cd tmp/
osuser@debian:~/Mytar/tmp$ ../mytar -x -f ../test.mtar
[0]: Creating file a.txt, size 7 Bytes...Ok
[1]: Creating file ab.txt, size 8 Bytes...Ok
[2]: Creating file abc.txt, size 9 Bytes...Ok
[3]: Creating file abcd.txt, size 24 Bytes...Ok
osuser@debian:/Mytar/tmp$ ls
a.txt  ab.txt  abc.txt  abcd.txt
osuser@debian:~/Mytar/tmp$ for file in *.txt; do diff ${file} ../${file}; done
osuser@debian:~/Mytar/tmp$
```

Apartado	Visto	Comentarios Profesor	Firma Profesor
2.A			
2.B			

FIRMA CONFORME (Alumno):