Sistemas Distribuidos

1. Librería de Threads colaborativos (I)

Implemente una biblioteca de threads colaborativos formada por los siguientes ficheros:

a) `threads.h` , que es el fichero de cabeceras que ofrece la interfaz de la biblioteca de threads:

```
void initthreads(void);
int createthread(void (*mainf)(void*), void *arg, int stacksize);
void exitsthread(void);
void yieldthread(void);
int curidthread(void);
```

b) `threads.c` , que debe implementar las funciones que forman la interfaz de la biblioteca:

`initthreads` es la función que inicializa la biblioteca de threads. Debe llamarse al inicio del programa principal que se enlace con la biblioteca, y únicamente puede llamarse una vez. Cada thread debe tener un identificador (entero positivo) que no se debe reutilizar. Después de llamar a esta función de inicialización, el hilo que ejecuta el programa principal pasará a ser el thread con identificador 0. Dicho thread puede usar como pila el segmento de pila del propio proceso.

`createthread` crea un nuevo thread, que ejecutará la función que se le pasa como argumento (que a la vez acepta como parámetro un `void *`). También se le pasa el tamaño de la pila para dicho thread. Esta función crea el nuevo thread, pero no cambia el contexto (no pone el nuevo thread a ejecutar). Se puede suponer un límite para el número de threads concurrentes (32 threads). Esta función devuelve el identificador del nuevo thread. Si no se puede crear el nuevo thread por algún motivo, se debe devolver -1.

`exitsthread` termina la ejecución del thread actual. No admite ningún argumento. Todo thread debe acabar mediante una llamada a esta función.

`yieldthread` hace que el thread que ejecuta pueda ceder el procesador a otro thread. La política de planificación será similar a una política Round-Robin con un cuanto de 200 ms. Si cuando se llama a yieldthread el thread ha consumido su cuanto, se elegirá otro thread (si lo hay) para ejecutar. Si todavía no ha consumido su cuanto, podrá continuar ejecutando. Se recomienda implementar el planificador en una función auxiliar.

`curidthread` retorna el identificador del thread actual.