INTEGRANTES DEL GRUPO:

- David Gabriel Suárez (uo270943) - Moisés Sanjurjo Sánchez (uo270824) - Adolfo Rodríguez Sánchez(uo271620) - Federico Cuervo Ricardo (uo67609)

En esta parte del trabajo en grupo se tuvo que implementar una versión del primer algoritmo creado, el cual aceptase la ejecución en multihilo, para ello se añadió una constante igual al numero de hilos a utilizar, un array de “pthread_t” de tamaño igual al número de hilos y una variable global igual a un array de tamaño igual a los hilos.

Para la creación de los hilos se utiliza la función “pthread_create” al cual se le pasa como primer parámetro un identificador del hilo, un objeto atributo (que en nuestro caso es NULL) como segundo parámetro, como tercero la tarea que debe realizar, y como ultimo un argumento el cual se le pasa a la tarea.

El método que realizamos recibe como parámetro un argumento de cualquier tipo, aunque a la hora de ejecución a este se le pasa una referencia a un argumento de tipo “int”, el cual es igual al numero de hilo creado. El método en si utiliza el mismo algoritmo utilizado en la fase 1, pero utilizando un for para atribuir a cada hilo un rango de pixeles sobre el cual trabajar, por tanto, cada hilo trabajaría con una parte de la imagen. Tras haber creado los hilos, se llama a la función “pthread_join” la cual sincroniza los hilos de ejecución bloqueando el hilo de llamada hasta que no termine el hilo especificado. Se le pasan dos parámetros el primero es igual al hilo que debe esperar, y el segundo es un puntero que apunta a la dirección de retorno del hilo.

A continuación se resumen las mediciones de tiempos de ejecución realizadas en las mismas condiciones de las hechas sobre la versión implementada en la fase 1. Las mediciones fueron realizadas con 8 hilos de ejecución, tantos como soporta la máquina de pruebas.

	Media ts1 a ts10 (s)	Productividad (tareas/min)	Aceleración
SingleThread	7,92	7,58	-
MultiThread	2,03	29,53	3,90

• Federico Cuervo Ricardo: Participación en las reuniones para depurar y optimizar algoritmo inicial y eliminar código redundante.

• David Gabriel Suárez: Participación en las reuniones para depurar y optimizar algoritmo inicial y eliminar código redundante, y creación de versión de prueba de este.

• Adolfo Rodríguez Sánchez: Participación en las reuniones de depuración y optimización del código y finalización de la memoria, junto con las mediciones de los

tiempos en la máquina de pruebas, elaboración del informe sobre los mismos, y elaboración del código final a partir de anteriores versiones.

- Moisés Sanjurjo Sánchez: Participación en las reuniones de depuración y optimización del algoritmo inicial y implementación de una primera versión del programa