1) Haz un programa que ejecute en un thread una suma de los valores 300 y 500.

2)Haz un programa cuyo main imprima por pantalla “Hola, soy el main” y un thread que escriba “Hola, soy ” + el nombre del thread.

3)Haz un programa cuyo main cree 2 threads, cada uno de ellos imprimirá por pantalla el nombre del thread.

4)Haz un programa cuyo main reciba por consola una letra. Después, creará un thread y este imprimirá por pantalla la letra 4 veces.

5)Haz un programa cuyo main cree un vector de enteros tamaño 10. El vector debe tener valores en todas sus posiciones. Después, creará un thread que recibirá como parámetro este vector y multiplicará por 2 todas sus posiciones (guardándolas en el mismo vector). Sin ningún tipo de espera, el programa principal escribirá todas las posiciones del vector. Explica el resultado.

6)Haz un programa cuyo main cree un vector de enteros tamaño 10. El vector debe tener valores en todas sus posiciones. Después, creará un thread que recibirá como parámetro este vector y multiplicará por 2 todas sus posiciones (guardándolas en el mismo vector). Finalmente, el programa principal esperará a que la ejecución del thread haya terminado y escribirá todas las posiciones del vector.
Explica el resultado.

7)Haz un programa que cree N threads, cada uno de ellos imprimirá por pantalla el nombre del thread.

8)Haz un programa cuyo main vaya pidiendo por consola palabras. Al recibir una, creará un thread que se encargará de darle la vuelta a la palabra e imprimirla por pantalla. El main seguirá pidiendo palabras hasta que reciba la palabra “SALIR”. El programa principal no puede terminar hasta asegurarse de que todos los threads han terminado…

9)Haz un programa cuyo main vaya pidiendo por consola enteros. Al recibir uno, creará un thread que se encargará de determinar si es primo o no e imprimirlo por pantalla. El main seguirá pidiendo enteros hasta que reciba el número -1. Es necesario hacer control de errores para no permitir números negativos (excepto el de control). El programa principal no puede terminar hasta asegurarse de que todos los threads han terminado…
Para evitar problemas al tener demasiados threads, se limitarán los threads a 5.
En caso de recibir un número y que no queden threads disponibles se esperará hasta que alguno acabe.

Recuerda, un número es primo si solo es divisible entre sí mismo y 1.

10) Haz un programa que cree N threads, cada uno de ellos imprimirá por pantalla el nombre del thread. Asegura que se ejecutan en orden: primero thread1, después thread2…

11) Haz un programa cuyo main cree 2 threads, cada uno de ellos imprimirá por pantalla el nombre del thread. Cuando acaben, el main imprimirá “FIN”

12) Haz un programa cuyo main cree dos threads. El primer thread escribirá 20 veces la letra ‘s’ por pantalla. El segundo thread escribirá 20 veces la letra ‘o’ por pantalla.

13) Haz un programa cuyo main cree dos threads. El primer thread escribirá 20 veces la letra ‘s’ por pantalla. El segundo thread escribirá 20 veces la palabra “HOLA” por pantalla.

14) Haz un programa con una variable global SUMATOTAL. El main creará dos threads que hagan 50 incrementos cada uno a esta variable. Cuando hayan terminado, el main escribirá por pantalla el resultado.

15) Haz un programa que, dado un vector de enteros con los números del 100 al 150 (incluidos), calcule el cuadrado de todos sus números y lo imprima por pantalla. Utiliza paralelismo.

16) Se puede utilizar la instrucción System.currentTimeMillis() para conseguir el tiempo del sistema. Con una simple resta es posible calcular el tiempo de ejecución de un programa. Mide el tiempo de ejecución del ejercicio anterior en una ejecución secuencial y el tiempo de ejecución en la ejecución de tu programa paralelo. Si el tamaño del vector es N, ¿a partir de qué orden de magnitud de N empiezas a ver alguna diferencia?