Los ejercicios que se plantean a continuación van a realizar tareas
que van a requerir una coordinación sencilla a través del método join().

Ejercicios hilos que se coordinan con join()

1. Importa a Eclipse el proyecto ParesMensaje Runnable Anonima AL y complétalo.
Dentro del método main():

- crea un hilo que implemente Runnable con clase anónima (consulta los apuntes) que escriba los pares entre 2 y 20 junto con el nombre del hilo. Después de escribir cada nº par el hilo se duerme 1 segundo
- crea un hilo que implemente Runnable con clase anónima que muestre el mensaje "*ejemplo con join()*" 10 veces junto con el nombre del hilo. Después de cada mensaje el hilo se duerme 1 segundo
- inicia los hilos
- haz que el hilo main() espere a que los dos hilos anteriores terminen
- escribe un mensaje que indique que el hilo main() ha terminado
- propaga las excepciones producidas por join()

posible ejecución

```
Thread-7  2
Thread-8 Ejemplo con join()
Thread-7  4
Thread-8 Ejemplo con join()
Thread-8 Ejemplo con join()
Thread-7  6
Thread-8 Ejemplo con join()
Thread-7  8
Thread-8 Ejemplo con join()
Thread-7  10
Thread-7  12
Thread-8 Ejemplo con join()
Thread-8 Ejemplo con join()
Thread-7  14
Thread-8 Ejemplo con join()
Thread-7  16
Thread-7  18
Thread-8 Ejemplo con join()
Thread-7  20
Thread-8 Ejemplo con join()
Aquí acaba el hilo main()
```

Ejecuta varias veces el programa para observar los resultados que se producen

2. Importa el proyecto Array contar pares con join AL y complétalo.
La clase TareaContarPares:

- será un objeto Runnable que calculará los pares que hay en un *array* (dados unos límites)
- la tarea concurrente que se codificará en el método run() además de contar los pares, en cada iteración dormirá el hilo 500 *msg.*
- al final se escribe un mensaje indicando el nombre del hilo y su finalización

posible ejecución

```
Array inicial
3 6 20 17 13 9 18 4 9 6 9 1 15 18 16
Iniciados los 3 hilos
Final hilo Thread-2
Final hilo Thread-1
Final hilo Thread-0
Otra vez en el hilo main()
Total pares en el array  7
```

La clase TestContarPares incluye el método main() que se completará de la siguiente forma:

- inicializar *array* con valores aleatorios comprendidos entre 1 y 20 (utilizando Math.random())
- mostrar *array* en pantalla
- crear e iniciar tres hilos. Cada uno tratará una tercera parte del *array* (el primer hilos los 5 primeros elementos, el segundo hilo los 5 elementos siguientes, …..)
- hacer que el hilo main() espere a que acaben los tres hilos su trabajo
- escribir en pantalla el total de pares que tenía el *array*

Ejecuta varias veces el programa para observar los resultados que se producen.

3. Importa el proyecto Contador palabras en ficheros con join AL.
Completa la clase ContadorPalabras tal como se indica en el código.

posible ejecución

```
Error en nº de argumentos,
Sintaxis: java TestContadorPalabras  nombre1   nombre2 ......
```

```
Fin del hilo Thread-2 Palabras en fichero fichero3.txt: 7
Fin del hilo Thread-0 Palabras en fichero fichero1.txt: 5
Fin del hilo Thread-1 Palabras en fichero fichero2.txt: 4
Total palabras en todos los ficheros 16
```

La clase TestContadorPalabras se completará de la siguiente forma:

- los nombres de los ficheros se pasan como argumentos desde línea de comandos. Si no se han pasado argumentos se muestra un mensaje de error y la sintaxis adecuada para llamar al programa (ver figura) y el programa termina
- se define un array de tipo ContadorPalabras con tantos elementos como nombres de fichero se hayan pasado (es un array de hilos)
- con un bucle se recogen los nombres de ficheros desde línea de comandos y se crean e inician los hilos
- con un bucle se recorre el array de hilos anterior y se indica que el hilo *main* espere a que todos acaben
- con un bucle se calcula el total de palabras contadas entre todos los hilos
- se muestra un mensaje indicando que el hilo *main* ha terminado

Ejecuta varias veces el programa para observar los resultados que se producen.