Examen PSP: Hilos, Sincronización y Sockets. Tipo 2.

- Entregar un proyecto de IntelliJ IDEA exportado en zip con **un módulo por cada uno de los ejercicios**.
- Asegúrate de que los proyectos **compilan**.
- Utiliza el **JDK 17**.

Ejercicio 1. Hilos (3 ptos)

Escribe un programa que mediante el uso de la clase `Thread` y la clase `URL` permita descargar **en paralelo** los siguientes ficheros:

- https://httpbin.org/image/jpeg
- https://httpbin.org/image/png
- https://httpbin.org/image/svg
- https://httpbin.org/image/webp

Consideraciones

1. Utiliza el **paquete** `com.[nombreapellido].psp.hilos`
2. Se deben descargar los ficheros utilizando las clases apropiadas para leer y escribir **en modo binario**.
3. La descarga debe realizarse en paralelo, no de forma secuencial.
4. Se deben **cerrar los recursos** utilizados.
5. Se debe gestionar correctamente el **ciclo de vida de los hilos**.
6. Se deben guardar los ficheros descargados en la carpeta `downloads` del usuario, cada fichero se guardará con el nombre de `file.extension`, donde la extensión es jpeg,png, svg y webp.

Ejercicio 2. Sincronización (4 ptos)

Utiliza las clases `Thread` y `Semaphore` para escribir un programa que simule el funcionamiento de una carrera de caballos (HorseRace). En la carrera participan una serie de caballos(Horse) que tardan un tiempo aleatorio en completar la carrera y publican este tiempo.

La carrera debe esperar a que los caballos terminen de correr, obtiene el tiempo que ha tardado cada caballo, y lo saca por pantalla.

Los caballos, una vez han generado el valor aleatorio, deben esperar a que la carrera les permita correr de nuevo.

Este ciclo se repite una serie predeterminada de veces.

Una posible salida del programa, en la que tenemos 5 caballos corriendo un tiempo aleatorio de entre 1 y 3 segundos y en el que se repite la carrera 3 veces sería:

```
Race 0 results
horse 0 took 1,31 seconds
horse 1 took 3,51 seconds
horse 2 took 2,69 seconds
horse 3 took 3,92 seconds
horse 4 took 1,27 seconds
Race 1 results
horse 0 took 3,84 seconds
horse 1 took 2,33 seconds
horse 2 took 1,63 seconds
horse 3 took 2,34 seconds
horse 4 took 1,36 seconds
Race 2 results
horse 0 took 3,17 seconds
horse 1 took 3,07 seconds
horse 2 took 3,25 seconds
horse 3 took 3,27 seconds
horse 4 took 2,63 seconds
Horse done
Horse done
Horse done
Horse done
Horse done
HorseRace done
```

Programa las clases

- `Horse` en la que se puede configurar por constructor el tiempo máximo de de carrera y en que el tiempo que tarda será aleatorio entre 1 segundo más el tiempo máximo.
- `HorseRace` en la que se puede configurar el número de caballo, el tiempo máximo que estos corren y el número de veces que se repite el juego.
- Sincroniza estas clases mediante el uso de semáforos.
- Puedes formatear la salida del tiempo empleado por los caballos con `%.2f` en `printf`.
- Utiliza el **paquete** `com.[nombreapellido].psp.sincronizacion`

Ejercicio 3. Sockets (3 ptos)

Escribe un servidor **multihilo** que permita a un cliente obtener un número aleatorio.

- Cuando el servidor reciba un mensaje con un número válido, responderá con un número aleatorio entero generado entre cero y el número recibido.
- En caso contrario, debe responder con la cadena `please write a number`
- Utiliza `Integer.parseInt` para comprobar si el número proporcionado es válido.

Escribe un cliente que permita interactuar de forma asíncrona con los datos que proporciona el usuario y con los datos recibidos desde el servidor.

Utiliza el **paquete** `com.[nombreapellido].psp.sockets`