ADIVINA MI PERSONAJE

En que consiste el juego:
Pensar un personaje real o ficticio famoso. Pensar 3 pistas para poder adivinarlo, cada una de las pistas será más certera que la anterior. ¡OJO! EL personaje es vuestro secreto, nadie debe saberlo.

Cada uno levantará una API Rest que podrán consultar sus compañeros. En ella, se podrán consultar las 3 pistas que has pensado y además vuestros compañeros tendrán la posibilidad de probar suerte e intentar adivinar el personaje.

Por último, cuando todo el mundo tenga su API levantada y operativa, cada uno programará en IntelliJ un programa que recorra todas las IPs de clase y vaya intentando resolver todas las adivinanzas de la clase.

Parte servidor (API REST):
Para esta parte vamos a usar SpringBoot. Crearemos un nuevo proyecto llamado "juego" en el puerto 8080 (por defecto) que constará de los siguientes EndPoints:

@GetMapping("/") : devuelve boolean, true si la API está lista, false si no.
@GetMapping("/pista1") : devuelve String con la pista más complicada
@GetMapping("/pista2") : devuelve String con la pista de dificultad media
@GetMapping("/pista3") : devuelve String con la pista de dificultad más baja
@GetMapping("/resolver/{name}") → : devuelve boolean

- {name} es el string que enviarán los jugadores para intentar acertar el personaje
- Es conveniente hacer el equalsIgnoreCase.

@GetMapping("/statistics") : devuelve String con las estadísticas del servicio.

- Número de veces que se ha consultado la API
- Número de veces que se ha acertado el personaje
- Número de veces que se ha fallado

@GetMapping("/score") : devuelve int

- Este apartado es para consultar vuestra puntuación

@GetMapping("/score/password/{score}")

- Este apartado es para que añadáis a vuestra API una nueva puntuación máxima obtenida. Se usará cuando tengáis diseñada la parte de cliente y empecéis a jugar con las APIs de vuestros compañeros.
- La password es básicamente para que ninguno de vuestros compañeros pueda cambiaros vuestra puntuación máxima a 0. De esta manera, solo vosotros tendréis acceso a modificar ese valor en vuestro servicio.

Parte cliente:

Esta parte vamos a usar como hasta ahora IntelliJ. Debemos crear un programa que haga lo siguiente:

- Tendremos en un array de String[] todas las Ips de nuestros compañeros de clase, excluyendo la nuestra.
- El programa deberá recorrer todas esa IPs.
- Por cada IP el programa se deberá comportar de la siguiente manera.
    - Hacer un bluce con 3 iteraciones (una por cada pista), empezando desde 1 y acabando en 3.
        - Mostrar la pista 1: (Responder s/n) → Si se responde y se acierta son 10 puntos y salta a la siguiente IP. Si se responde y no se acierta se ha perdido y salta a la siguiente. Si no se responde salta a la siguiente pista.
        - Mostrar pista 2: (Responder s/n) → Mismo comportamiento que la anterior, pero en este caso si se acierta, son solo 5 puntos.
        - Mostrar pista 3: (Responder s/n) → Mismo comportamiento que la anterior, pero en este caso si se acierta, son solo 3 puntos.
        - Fijaos que si se divide 10 entre la i(número de iteración) ya nos da la puntuación obtenida en la pregunta.

- Cuando terminemos de recorrer todas las IPs se devolverá la puntuación obtenida y se hará una llamada a nuestra API /score. Si la puntuación que hemos obtenido es mayor que la que tenemos en nuestra API, se modificará la variable score en nuestra API a través del siguiente EndPoint:
    - /score/nuestraPassword/nurestraNuevaPuntuación

Función para consumir APIs:

```
public static String consumeAPI(String ip, String path) throws
URISyntaxException, IOException, InterruptedException {
        URI targetURI = new URI("http://" + ip + ":8080/" + path);
        HttpRequest httpRequest =
HttpRequest.newBuilder().uri(targetURI).GET().build();
        HttpClient httpClient = HttpClient.newHttpClient();
        HttpResponse<String> response = httpClient.send(httpRequest,
HttpResponse.BodyHandlers.ofString());
        return response.body();
}
```