

[illegible]

En la tabla se puede observar que se guarda la contraseña encriptada, sería como la clave pública, se puede ver pero necesitas otra clave para desencriptar y poder entrar en la aplicación (es la clave que introduces en la aplicación).

Al arrancar el servidor, reseteo de la BD 'trivia.db' todos los puntos de los jugadores y el campo de 'jugando' a 0, que es para comprobar si un jugador ya está jugando y no duplicar el mismo jugador.

```
Servidor: Se han actualizado todos los jugadores a 'NO JUGANDO' (0)
Servidor: Se han actualizado todos los puntos de los jugadores a 0
Servidor escuchando...
```

Ahora arranco un cliente:

```
SERVIDOR:
MENU INICIO
Elige una opción:

1 Entrar
2 Registrar Usuario
3 Salir

Opción:
```

Eligiendo la opción 1, entro directamente logueándome (nick y contraseña):

NICK	CONTR.	CONTRASEÑA ENCRYPTADA EN LA BD
jonrula	1234	03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
txus	1234	03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
luismi	1234	03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
elen	1234	03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4

```
SERVIDOR:
MENU INICIO
Elige una opción:

1 Entrar
2 Registrar Usuario
3 Salir

Opción:1
SERVIDOR Nick: jonrula
Generando par de claves
Enviamos la clave publica cuyo valor es: RSA
SERVIDOR Contraseña: 1234
Contraseña encriptada sha256:03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
SERVIDOR: Usuario validado ...

MENU JUGAR
Elige una opción:

1 Aceptar reglas y jugar
2 Volver

Opción:
```

El nick , se manda encriptado asimétricamente al servidor, y la contraseña, que es "1234", se manda encriptada, con una librería 'Digeutils'.

Se comprueban los datos y luego se verifica la contraseña encriptada en la BD que corresponde a la contraseña introducida:

SERVIDOR:

```
Servidor: Se han actualizado todos los jugadores a 'NO JUGANDO' (0)
Servidor: Se han actualizado todos los puntos de los jugadores a 0
Servidor escuchando...
Establecida conexión desde el cliente 1 ...
RECIBIENDO DATOS DESDE LA BASE DE DATOS TRIVIAL ...

Recibiendo opción del menú del cliente ...
SERVIDOR: Opción 1
Leemos la clave
La clave recibida es: RSA
0006000000*0fI);0"0r o000rirI00;/o0:
000w000000:004!0o0B0 P000.00 0000m{V70 00000s0z83.03c000 0>0E!0MxD0o0F0evg0P0000hN0HV00p2090zt0 0000
Mensaje descifrado con clave privada: jonrula
Contraseña encriptada recibida del cliente jonrula: 03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
Se ha actualizado el jugador JONRULA a 'Jugando':1 ...
Recibiendo opción del menú del cliente ...
```

El servidor valida que el usuario es correcto, comprobando el nick , la contraseña (comprobando que el mismo valor encriptado que se guarda en la BD) y también el estado del jugador, que tiene que ser 0, para que pueda jugar, luego lo actualizo en la BD a 1, para saber que está jugando.

```
// Compruebo Usuario y contraseña y que no este jugando --> 0
for (Jugador jugador : jugadores) {
    if (jugador.getNick().equalsIgnoreCase(nombre) &&
        jugador.getContrasena().equalsIgnoreCase(contrasena) &&
        jugador.getJugando() == 0) {

        correcto = true;

        // actualizamos el valor de 'jugando' en la BD para que no juegue otra vez, mientras está jugando
        PreguntarBD.actualizarJugando( jugando: 1, nombre);
    }
}
```

Para encriptar la contraseña, utilizo una librería, 'DigestUtils' que me da la opción de encriptar en diferentes algoritmos y me crea una contraseña cifrada que mando al servidor y guardo en la BD.

CLIENTE:

```
// Comprobar Usuario
System.out.print(reciboDato.readUTF());
String nombre = br.readLine();

// Envío mensaje cifrado al servidor
cifradoMensajeCliente(clienteSSL, nombre);
//envioDato.writeUTF(nombre);

// Comprobar contraseña
System.out.print(reciboDato.readUTF());
String contraseñaPrivada = br.readLine();

// Contraseña encriptada
String contraseñaEncriptsha256 = DigestUtils.sha256Hex(contraseñaPrivada);
System.out.println("Contraseña encriptada sha256:" + contraseñaEncriptsha256);

envioDato.writeUTF(contraseñaEncriptsha256);

// Recibo la validación del servidor , para poder entrar a jugar
boolean validarUsuario = reciboDato.readBoolean();

// Si usuario y contraseña SI son correctos, entro a jugar
if (validarUsuario) {
```

Ahora accedo al siguiente menú:

```
MENU JUGAR
Elige una opción:

    1 Aceptar reglas y jugar
    2 Volver

Opción:
```

Ahora hay que aceptar las reglas (firma digital), en la opción 1:

```
MENU JUGAR
Elige una opción:

    1 Aceptar reglas y jugar
    2 Volver

Opción:
1
Leemos la clave:
La clave recibida es: RSA

SERVIDOR:
Estas son las reglas del juego TRIVIAL :
- Solo puede jugar un jugador a la vez (no hay duplicados de jugadores).
- Hay un mínimo de 2 jugadores.
- Cada pregunta se contesta solo 1 vez, luego se borra.
- Solo hay una respuesta correcta por cada pregunta.
- Se puede salir de la aplicación sin contestar todas las preguntas.
- Se empieza de nuevo si se sale de la aplicación, no se almacenan los resultados.
- Cada respuesta correcta son 10 puntos.
- Gana el que más puntos obtenga, al finalizar todos (hilos) los jugadores que salgan del juego

Estas de acuerdo con las reglas del juego (s/n) ?
```

Le doy la opción al usuario de aceptar o no:

- Si acepta, va directamente a las preguntas del trivial:

```
Este mensaje va a ser firmado...
Verifico firma
El mensaje es auténtico

***** TRIVIAL *****

Bienvenido JONRULA a Trivial

SERVIDOR: Elige una pregunta:

    1 ¿Cuál es el monte más alto del sistema solar ?
    2 ¿Cuál es la parte más profunda de la corteza terrestre?
    3 ¿Cuál es la capital de Marruecos?
    4 ¿En qué país se encuentra el río Po?
    5 ¿Cuál es el país más visitado del mundo?
    6 ¿Quién desarrolló originalmente el desarrollo del lenguaje de programación JAVA?
    7 ¿Qué significa los acrónimos de JAVA?
    8 ¿Qué compañía desarrolló la implementación de referencia original para los compiladores de Java?
    9 ¿En qué año fue comercializada por primera vez el lenguaje de programación Java?
    10 ¿Cómo se llamó el proyecto para el desarrollo de Java ?
    11 ¿Cuáles son los cuatro primeros bytes (el número mágico) de los archivos.class que genera el compilador en JAVA?
    12 ¿Cómo se llamó el prototipo de navegador Web para el desarrollo de Java?
    13 ¿Cuál era la promesa inicial en el desarrollo de Java?
    14 ¿Cuál es el origen del logo de Java?
    15 ¿Cómo se denominó inicialmente el lenguaje de programación Java?

OPCIÓN:
```

- Si no acepta, no puede acceder a jugar y va al menú principal:

```
Estas de acuerdo con las reglas del juego (s/n) ?
n
SERVIDOR: Sentimos que no estes de acuerdo con las reglas de juego ...
SERVIDOR:
MENU INICIO
Elige una opción:

    1 Entrar
    2 Registrar Usuario
    3 Salir

Opción:|
```

El juego funciona eligiendo una pregunta, salen las 4 opciones y se elige una respuesta, Tanto la pregunta elegida como la respuestas van cifradas al servidor que comprueban si es un número y si esté está entre el rango de preguntas o respuestas. Por cada opcion elegida, ya sea pregunta o respuesta que se manda cifrada al servidor, se crean las dos claves pública y privada, podía haber generado solo un par de claves para todo el tiempo de ejecución del juego y mandarle al servidor solo la clave pública y que descrypte todas las opciones de preguntas y respuestas, pero lo he hecho que lo haga por cada opción, así introduzco un nivel más de seguridad.

CLIENTE:

```
OPCIÓN:
1
Generando par de claves
Enviamos la clave publica cuyo valor es: RSA

SERVIDOR: Elige una respuesta:

¿Cuál es el monte más alto del sistema solar ?

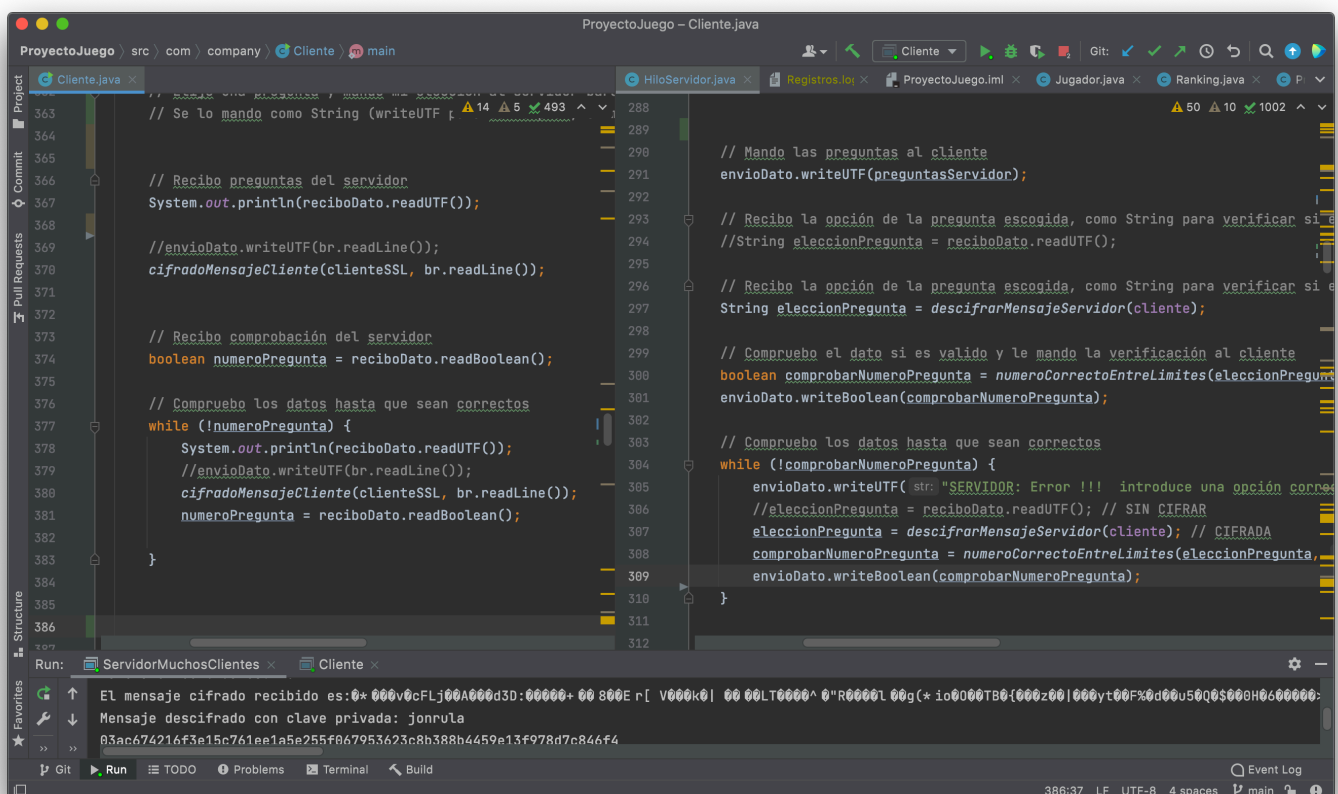
1 Doom Mons en Titán
2 Monte Everest
3 Cumbre central de Rheasilvia, en Vesta
4 Monte Olimpo, en Marte

Respuesta:
```

SERVIDOR:

```
Leemos la clave
La clave recibida es: RSA
00~0000T}0{u000C0뵆u ) a00>k 0뵆0Jh000U!08_뵆`S^0\0000$;t00;,(0Ac0m#00k+U 000 0xJ#0 +00M000hR00A0hf0 tL 00J004 0000-R0V 000
Mensaje descifrado con clave privada: 1
```

Un ejemplo de como se comprueba un dato entre el cliente y el servidor: tienen que estar los dos coordinados:



```
ProyectoJuego - Cliente.java
src \ com \ company \ Cliente \ main

// Se lo manda como String (writeUTF)
// Recibo preguntas del servidor
System.out.println(reciboDato.readUTF());
//envioDato.writeUTF(br.readLine());
cifradoMensajeCliente(clienteSSL, br.readLine());

// Recibo comprobación del servidor
boolean numeroPregunta = reciboDato.readBoolean();

// Compruebo los datos hasta que sean correctos
while (!numeroPregunta) {
    System.out.println(reciboDato.readUTF());
    //envioDato.writeUTF(br.readLine());
    cifradoMensajeCliente(clienteSSL, br.readLine());
    numeroPregunta = reciboDato.readBoolean();
}

// Mando las preguntas al cliente
envioDato.writeUTF(preguntasServidor);

// Recibo la opción de la pregunta escogida, como String para verificar si es
//String eleccionPregunta = reciboDato.readUTF();

// Recibo la opción de la pregunta escogida, como String para verificar si es
String eleccionPregunta = descifrarMensajeServidor(cliente);

// Compruebo el dato si es valido y le mando la verificación al cliente
boolean comprobarNumeroPregunta = numeroCorrectoEntreLimites(eleccionPregunta);
envioDato.writeBoolean(comprobarNumeroPregunta);

// Compruebo los datos hasta que sean correctos
while (!comprobarNumeroPregunta) {
    envioDato.writeUTF(str: "SERVIDOR: Error !!! introduce una opción correcta");
    //eleccionPregunta = reciboDato.readUTF(); // SIN CIFRAR
    eleccionPregunta = descifrarMensajeServidor(cliente); // CIFRADA
    comprobarNumeroPregunta = numeroCorrectoEntreLimites(eleccionPregunta);
    envioDato.writeBoolean(comprobarNumeroPregunta);
}

Run: ServidorMuchosClientes x Cliente x
El mensaje cifrado recibido es:0* 000v0cFLj00A000d3D:00000+ 00 800E n[ V000k0| 00 00LT00000 0"R0000L 00g(* io0000TB0:000z00|000yt00FX0d00u50Q0$000H0600000
Mensaje descifrado con clave privada: jonrula
83ac674216f3a15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
```

También tengo funciones para comprobar que es un número correcto, entre unos límites, por ejemplo para usar en los menús y en las preguntas y respuestas:

```
public static boolean numeroCorrectoEntreLmites(String respuesta, int numAbajo, int numArriba) {
    // Creo un objeto br de tipo BufferedReader para coger las respuestas por teclado
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

    boolean correcto = true;

    int numero;

    try {
        numero = Integer.parseInt(respuesta);
        if (numero < numAbajo || numero > numArriba) {
            System.out.println("Error !! ... introduce un número entre 1 y " + numArriba);
            correcto = false;
        }
    } catch (NumberFormatException e) {
        System.out.println("Error !! ... introduce un número");
        correcto = false;
    }

    return correcto;
}
```

También otras funciones para comprobar que los datos introducidos cumplen un patrón:

Comprobar nombres y apellidos (nombres compuestos y tildes entre 3 y 20 caracteres)

```
public static boolean comprobarNombreoApellidos(String nombre) {

    // Comprueba nombres compuestos tildes de entre 3 y 20 caracteres en total
    boolean correcto = false;
    String regexp = "(?=.{3,20}$)[a-zA-ZáéíóúüñÁÉÍÓÚÑ]+(?:[\\s][a-zA-ZáéíóúüñÁÉÍÓÚÑ]+)*$";

    if (Pattern.matches(regexp, nombre)) {
        correcto = true;
    } else {
        System.out.println("Introduce un formato correcto de nombre o apellido entre 3 y 20 caracteres !!");
    }

    return correcto;
}
```

Comprobar Nick (letras y números entre 4 y 20 caracteres)

```
public static boolean comprobarNick(String nick) {  
  
    // Comprueba nombres sin espacios, ni tildes, solo letras y números entre 4 y 20 caracteres en total  
  
    boolean correcto = false;  
    String regexp = "^(?=.{4,20}$)[a-zA-Z0-9]*$";  
  
    if (Pattern.matches(regexp, nick)) {  
        correcto = true;  
    } else {  
        System.out.println("Introduce un formato correcto de nick, solo letras y números entre 4 y 20 caracteres !!");  
    }  
  
    return correcto;  
}
```

Comprobar Contraseña

- Cualquier texto con símbolos, números y letras ente 4 y 70 caracteres.

```
public static boolean comprobarContrasena(String contrasena) {  
  
    // Comprueba la contraseña sin espacios, con todos los símbolos y que tenga entre 4 y 70 caracteres en total  
  
    boolean correcto = false;  
    String regexp = "^(?=.{4,70}$)[a-zA-Z0-9~!|_|.$%&/()=?_`' |<>`*+`,`;.:#@$-]*$";  
  
    if (Pattern.matches(regexp, contrasena)) {  
        correcto = true;  
    } else {  
        System.out.println("Introduce un formato correcto de contraseña , solo letras y números entre 4 y 20 caracteres si  
    }  
  
    return correcto;  
}
```


Si por ejemplo, queremos instanciar un nuevo jugador (hilo) , hay que verificar que no se instancie un jugador que ya está jugando, eso lo compruebo a través de la BD, mirando si el jugador tiene valor 0 ó 1 en la columna 'JUGANDO' de la tabla JUGADOR'.

Me logueo como 'jonrula' --> 1234

```
SERVIDOR:
MENU INICIO
Elige una opción:

1 Entrar
2 Registrar Usuario
3 Salir

Opción:1
SERVIDOR Nick: jonrula
Generando par de claves
Enviamos la clave publica cuyo valor es: RSA
SERVIDOR Contraseña: 1234
Contraseña encriptada sha256:03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
SERVIDOR: Usuario validado ...

MENU JUGAR
Elige una opción:

1 Aceptar reglas y jugar
2 Volver

Opción:
```

Instancio un nuevo jugador (hilo) e intento loguearme de nuevo como 'jonrula' --> 1234
No me deja y me manda al menú principal de nuevo:

```
SERVIDOR:
MENU INICIO
Elige una opción:

1 Entrar
2 Registrar Usuario
3 Salir

Opción:1
SERVIDOR Nick: jonrula
Generando par de claves
Enviamos la clave publica cuyo valor es: RSA
SERVIDOR Contraseña: 1234
Contraseña encriptada sha256:03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
SERVIDOR Usuario NO validado o ya está jugando
SERVIDOR:
MENU INICIO
Elige una opción:

1 Entrar
2 Registrar Usuario
3 Salir

Opción:
```

Desde el menú principal, puedo crear un nuevo usuario, donde introduzco los nuevos datos: nombre, apellidos, nick, edad, contraseña.

Pero primero hago una comprobación para que no se registre un nuevo usuario que ya existe en la BD:

```
SERVIDOR:
MENU INICIO
Elige una opción:

    1 Entrar
    2 Registrar Usuario
    3 Salir


Opción:2
REGISTRO USUARIO NUEVO
SERVIDOR Nombre: Juanan
SERVIDOR Apellido: Ruiz
SERVIDOR: EL usuario: 'Juanan Ruiz' ya existe !!
REGISTRO USUARIO NUEVO
SERVIDOR Nombre:
```

Vuelvo a introducir nuevos datos, cumpliendo los patrones de verificación:

```
SERVIDOR:
MENU INICIO
Elige una opción:

    1 Entrar
    2 Registrar Usuario
    3 Salir

Opción:2
REGISTRO USUARIO NUEVO
SERVIDOR Nombre: Juanan
SERVIDOR Apellido: Ruiz
SERVIDOR: EL usuario: 'Juanan Ruiz' ya existe !!
REGISTRO USUARIO NUEVO
SERVIDOR Nombre: Lucas
SERVIDOR Apellido: Ramirez
SERVIDOR Edad: 34
SERVIDOR Nick: lucas1
SERVIDOR Contraseña: 12589
Contraseña nueva encriptada sha256:b30bb16071db22210dc01c26f5b9d0d4e6de9f696f69af29d9a4f31816b4b053
```



... se inserta el nuevo usuario en la BD, hago un listado de los nuevos usuarios y entra directamente al siguiente menú para aceptar las reglas y jugar:

```
SERVIDOR:
MENU INICIO
Elige una opción:

    1 Entrar
    2 Registrar Usuario
    3 Salir

Opción:2
REGISTRO USUARIO NUEVO
SERVIDOR Nombre: Juanan
SERVIDOR Apellido: Ruiz
SERVIDOR: EL usuario: 'Juanan Ruiz' ya existe !!
REGISTRO USUARIO NUEVO
SERVIDOR Nombre: Lucas
SERVIDOR Apellido: Ramirez
SERVIDOR Edad: 34
SERVIDOR Nick: lucas1
SERVIDOR Contraseña: 12589
Contraseña nueva encriptada sha256:b30bb16071db22210dc01c26f5b9d0d4e6de9f696f69af29d9a4f31816b4b053

SERVIDOR: Lista actualizada de jugadores:
ID NOMBRE APELLIDOS NICK EDAD CONTRASEÑA RECORD
0 Juanan Ruiz jonrula 50 03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4 0
1 Luis Romero luismi 48 03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4 0
2 Txus Gonzalez txus 52 03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4 0
3 Elena Zamora elen 24 03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4 0
7 sergio perez 6543 50 ytre 0
8 luisito lop 45hj 34 03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4 0
12 pepe lopez pepito 23 b30bb16071db22210dc01c26f5b9d0d4e6de9f696f69af29d9a4f31816b4b053 0
13 Lucas Ramirez lucas1 34 b30bb16071db22210dc01c26f5b9d0d4e6de9f696f69af29d9a4f31816b4b053 0

SERVIDOR: Usuario nuevo añadido ...

MENU JUGAR
Elige una opción:

    1 Aceptar reglas y jugar
    2 Salir

Opción:
```

Se ha insertado el nuevo usuario en la BD, con la contraseña encriptada (que corresponde a '12589' y ya le pongo el valor de 'JUGANDO' a 1)

DB Browser for SQLite - /Users/juanan/Documents/PROGRAMACION TERCERO/PROGRAMACION SERVICIO Y PROCESOS/Actividades/SEGURIDAD/ProyectoEncriptacion/ProyectoJuego/...

Nueva base de datos

Abrir base de datos

Guardar cambios

Deshacer cambios

Abrir proyecto

Anexar base de datos

Estructura

Hoja de datos

Editar pragmas

Ejecutar SQL

Tabla: JUGADOR

Filtrar en cualquier columna

ID	NOMBRE	RECORD	APELLIDOS	NICK	EDAD	CONTRASEÑA	JUGANDO	
Filtrar	Filtrar	Filtrar	Filtrar	Filtrar	Filtrar	Filtrar	Filtrar	
1	0	Juanan	0	Ruiz	jonrula	50	0 3 ac 6 7 4 2 1 6 f 3 e 1 5 c 7 6 1 ee 1 a 5 e 2 5 5 f 0 6 7 9 5 3 6 2 3 c 8 b 3 8 8 b 4 4 5 9 e 1 3 f 9 7 8 d 7 c 8 4 6 f 4	0
2	1	Luis	0	Romero	luismi	48	0 3 ac 6 7 4 2 1 6 f 3 e 1 5 c 7 6 1 ee 1 a 5 e 2 5 5 f 0 6 7 9 5 3 6 2 3 c 8 b 3 8 8 b 4 4 5 9 e 1 3 f 9 7 8 d 7 c 8 4 6 f 4	0
3	2	Txus	0	Gonzalez	txus	52	0 3 ac 6 7 4 2 1 6 f 3 e 1 5 c 7 6 1 ee 1 a 5 e 2 5 5 f 0 6 7 9 5 3 6 2 3 c 8 b 3 8 8 b 4 4 5 9 e 1 3 f 9 7 8 d 7 c 8 4 6 f 4	0
4	3	Elena	0	Zamora	elen	24	0 3 ac 6 7 4 2 1 6 f 3 e 1 5 c 7 6 1 ee 1 a 5 e 2 5 5 f 0 6 7 9 5 3 6 2 3 c 8 b 3 8 8 b 4 4 5 9 e 1 3 f 9 7 8 d 7 c 8 4 6 f 4	0
5	7	sergio	0	perez	6 5 4 3	50	ytre	0
6	8	luisito	0	lop	4 5 hj	34	0 3 ac 6 7 4 2 1 6 f 3 e 1 5 c 7 6 1 ee 1 a 5 e 2 5 5 f 0 6 7 9 5 3 6 2 3 c 8 b 3 8 8 b 4 4 5 9 e 1 3 f 9 7 8 d 7 c 8 4 6 f 4	0
7	12	pepe	0	lopez	pepito	23	b3 0 bb 1 6 0 7 1 db 2 2 2 1 0 dc 0 1 c 2 6 f 5 b 9 d 0 d 4 e 6 de 9 f 6 9 6 f 6 9 af 2 9 d 9 a 4 f 3 1 8 1 6 b 4 b 0 5 3	0
8	13	Lucas	0	Ramirez	lucas1	34	b3 0 bb 1 6 0 7 1 db 2 2 2 1 0 dc 0 1 c 2 6 f 5 b 9 d 0 d 4 e 6 de 9 f 6 9 6 f 6 9 af 2 9 d 9 a 4 f 3 1 8 1 6 b 4 b 0 5 3	1

1 - 8 de 8

Ir a: 1

1 fila, 8 columnas. Suma: 48; Media: 6; Mín: 0; Máx: 34

UTF-8

Para finalizar el juego, ya sea como usuario ya logueado o usuario nuevo, mientras no tecleemos 'fin' no salimos del bucle y siguen apareciendo las preguntas, pero borrando las que has elegido previamente, para no repetir puntos.

En este caso estan dos jugadores (hilos) 'luismi' y 'elen'.

Al teclear 'fin' salen los puntos obtenidos en el juego en cada jugador (hilo).

'luismi':

```
SERVIDOR: Respuesta correcta !!
Quieres seguir jugando ? (intro o 'fin' para salir)
fin
SERVIDOR: Finalización del juego, puntos totales de luismi: 20
SERVIDOR:
MENU INICIO
Elige una opción:

    1 Entrar
    2 Registrar Usuario
    3 Salir

Opción:
```

'elen':

```
SERVIDOR: Respuesta correcta !!
Quieres seguir jugando ? (intro o 'fin' para salir)
fin
SERVIDOR: Finalización del juego, puntos totales de elen: 10
SERVIDOR:
MENU INICIO
Elige una opción:

    1 Entrar
    2 Registrar Usuario
    3 Salir

Opción: 3
SERVIDOR: Adios ELEN
```

... y al salir de la aplicación, opción 3, sale un ranking con la puntuación de todos los jugadores, que esten jugando mientras el servidor esté arrancado, se comprueba esos datos en la BD y cuando se pare y vuelva arrancar se resetean todos los jugadores a 0 puntos y 'JUGANDO' a 0, de la BD.

Para ello hago una nueva clase 'Ranking' con dos atributos, nombre y puntos, que se los paso de la tabla 'JUGADOR' y con la clase 'Collections.sort' puedo ordenarlos ...

SERVIDOR:

```
***** RANKING PUNTOS TRIVIAL *****

      PUESTO      NOMBRE      PUNTOS
      1           Luis        20
      2           Elena        10
      3           Juanan         0
      4           Txus          0
      5          sergio          0
      6          luisito          0
      7           pepe          0
      8           Lucas          0

*****
```

Le mando los datos del ranking al cliente:

CLIENTE:

```
SERVIDOR: LISTA DE PUNTOS ACTUALIZADA:
```

***** RANKING PUNTOS TRIVIAL *****		
PUESTO	NOMBRE	PUNTOS
1	Luis	20
2	Elena	10
3	Juanan	0
4	Txus	0
5	sergio	0
6	luisito	0
7	pepe	0
8	Lucas	0

También he generado unos certificados SSL para que la conexión al servidor sea más segura, situándome en la carpeta donde tengo el jdk actual y utilizando la herramienta keytool:

```
bin --zsh --180x60
Last login: Fri Dec 3 20:29:49 on ttys000
juanana@Macbook-de-Juanan ~ % /usr/libexec/java_home -v 17
/Users/juanana/Library/Java/JavaVirtualMachines/openjdk-17/Contents/Home
juanana@Macbook-de-Juanan ~ % cd /Users/juanana/Library/Java/JavaVirtualMachines/openjdk-17/Contents/Home
juanana@Macbook-de-Juanan Home % ls
bin  conf  include  jmods  legal  lib  release
juanana@Macbook-de-Juanan Home % cd bin
juanana@Macbook-de-Juanan bin % ls
jar          javadoc      jdb          jhsdb        jmap          jrunscript   jstatd
jarsigner    javap        jdeprscan    jimage        jmod          jshell        keytool
java         jcmd         jdeps        jinfo         jpackage      jstack        rmiregistry
javac        jconsole     jfr          jlink         jps           jstat        serialver
juanana@Macbook-de-Juanan bin % cd java
cd: not a directory: java
juanana@Macbook-de-Juanan bin % keytool -genkey -keyalg RSA -alias claveSSL -keystore /Users/juanana/Desktop/AlmacenSSL.jks
Enter keystore password:
[Re-enter new password:
What is your first and last name?
[Unknown]: Juanan Ruiz
What is the name of your organizational unit?
[Unknown]: Egibide
What is the name of your organization?
[Unknown]: Egibide
What is the name of your City or Locality?
[Unknown]: Vitoria
What is the name of your State or Province?
[Unknown]: Araba
What is the two-letter country code for this unit?
[Unknown]: ES
Is CN=Juanan Ruiz, OU=Egibide, O=Egibide, L=Vitoria, ST=Araba, C=ES correct?
[no]: y

Generating 2048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 90 days
juanana@Macbook-de-Juanan bin % keytool -import -alias claveSSL -file /Users/juanana/Desktop/Certificado.cer -keystore /Users/juanana/Desktop/UsuarioAlmacenSSL -storepass 890123
Owner: CN=Juanan Ruiz, OU=Egibide, O=Egibide, L=Vitoria, ST=Araba, C=ES
Issuer: CN=Juanan Ruiz, OU=Egibide, O=Egibide, L=Vitoria, ST=Araba, C=ES
Serial number: 90c648e180939a16
Valid from: Fri Dec 03 20:38:56 CET 2021 until: Thu Mar 03 20:38:56 CET 2022
Certificate fingerprints:
    SHA1: A1:E7:F0:34:71:06:8A:9D:AB:94:A0:36:E0:5F:BB:43:83:B6:F8:A2
    SHA256: 37:5C:33:C7:04:BB:12:A7:0E:0D:44:77:9F:40:7F:ED:DE:CE:03:C3:C2:10:1F:60:A4:50:CA:7F:43:AD:9B:AF
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 04 DD 10 38 0C 1D 2D 2F 51 42 D7 56 21 E9 CD 2F ...8.../QB.V!../
0010: 32 90 50 28 2.P(
]
]

Trust this certificate? [no]: y
Certificate was added to keystore
juanana@Macbook-de-Juanan bin %
```

SERVIDOR:

```
public class ServidorMuchosClientes {

    // No igualar a cero, para que no salga el mismo valor
    static int contadorServidor;

    public static void main(String[] args) throws IOException {

        // Actualizo a 0 todos los jugadores cada vez que arranco el servidor
        PreguntarBD.actualizarJugandoTodosJugadores();

        // Reseteo todos los puntos de los jugadores a 0, al arrancar el servidor
        PreguntarBD.actualizarPuntosJugadoresACero();

        // Creo un serversocket y solo le paso el puerto
        System.setProperty("javax.net.ssl.keyStore", "AlmacenSSL.jks");
        System.setProperty("javax.net.ssl.keyStorePassword", "12345678");

        SSLServerSocketFactory factory = (SSLServerSocketFactory) SSLServerSocketFactory.getDefault(); // Dejo constantemente abierta la escucha
        SSLServerSocket clienteSSL = (SSLServerSocket) factory.createServerSocket( port: 4999);
        System.out.println("Servidor escuchando...");

        // Aquí solo escucho y derivo a un hilo las peticiones del cliente, cada vez que ejecuto el play del cliente (EN un bucle sin parar)
        // No lo cierro, continuamente escucha
        while (true) {
            // Genero la conexión del cliente hacia el servidor, tengo que instanciar un cliente socket
            contadorServidor++;
            SSLSocket cliente = (SSLSocket) clienteSSL.accept();
            System.out.println("Establecida conexión desde el cliente " + contadorServidor + " ...");
            HiloServidor h = new HiloServidor(cliente, contadorServidor);

            h.start();
        }
    }
}
```

CLIENTE:

```
public class Cliente {

    public static void main(String[] args) throws IOException, NoSuchAlgorithmException, SignatureException, InvalidKeyException, ClassNotFoundException {

        // Creo un objeto br de tipo BufferedReader para coger las respuestas por teclado
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        System.setProperty("javax.net.ssl.trustStore", "UsuarioAlmacenSSL");
        System.setProperty("javax.net.ssl.trustStorePassword", "890123");

        // Me conecto a mi máquina local al puerto 4999
        SSLSocketFactory factory = (SSLSocketFactory) SSLSocketFactory.getDefault();
        SSLSocket clienteSSL = (SSLSocket) factory.createSocket( host: "localhost", port: 4999);
```

... y esto es todo espero que os guste

