



Argentina  
programa  
4.0



# JAVA

## Exposición Final

### Grupo N°6

Lourdes Aybar - Matias Ruben Candia - Franco Agustin Del For - Jefferson Gutierrez -

Carlos Antonio Patron - Franco Alanoca - Macarena Delgado Velez

# El Proceso

## ? **¿Qué problemas se encontraron al trabajar el código? ¿Cómo los solucionaron?**

En general, se nos dificultaron los procesos de manipulación y lectura de los archivos, también surgieron algunas dudas respecto a cómo realizar la conexión de SQL. Pudimos solucionarlo viendo videos, buscando información adicional, y por supuesto haciendo uso de las grabaciones de las clases junto a las referencias propuestas.

## ? **Al presentarse distintas opiniones dentro del grupo ¿Cómo las resolvieron?**

Se resolvieron implementando toda sugerencia hasta determinar en conjunto la opción más óptima, la que se acercaba al objetivo.

# El Proceso

- **Comunicación del grupo**

Para trabajar y desarrollar el proyecto, el grupo trabajó comunicándose mediante muchos medios, por un lado se creó un grupo de **WhatsApp**, se organizaron reuniones por google meet y el uso de Github como repositorio.

- **Trabajo en el grupo**

Se desarrolló el proyecto tratando de mantener una comunicación constante, las reuniones sirvieron para aclarar y resolver los problemas que se presentaron, cabe aclarar que como cada integrante tiene sus actividades personales aparte del curso, se programaban a partir de la disponibilidades de cada uno, para muchos fue la primera vez que se trabajó en un **grupo de trabajo**.

# Base de Datos



Argentina  
programa  
4.0

id	RONDA	EQUIPO1	EQUIPO2	GOLES1	GOLES2
1	1	ARGENTINA	ARABIA SAUDITA	1	2
2	1	POLONIA	MEXICO	0	0
3	1	ARGENTINA	MEXICO	2	0
5	1	ARABIA SAUDITA	POLONIA	0	2

4 rows in set (0.00 sec)

```
mysql> SELECT * FROM PRONOSTICOS;
```

id	PARTICIPANTE	EQUIPO1	GANA1	EMPATA	EQUIPO2	GANA2
1	MARIANA	ARGENTINA	X		ARABIA SAUDITA	
2	MARIANA	POLONIA		X	MEXICO	
3	MARIANA	ARGENTINA	X		MEXICO	
4	MARIANA	ARABIA SAUDITA			POLONIA	X
5	PEDRO	ARGENTINA	X		ARABIA SAUDITA	
6	PEDRO	POLONIA			MEXICO	X
7	PEDRO	ARGENTINA	X		MEXICO	
8	PEDRO	ARGENTINA		X	MEXICO	

8 rows in set (0.00 sec)

```
mysql>
```

# Desarrollo: Main

Este código se conecta a una base de datos MySQL y realiza una consulta a la tabla "RESULTADOS" de esa base de datos.

Tiene una sección "try" del código, se carga el controlador JDBC de MySQL. Luego se crea un objeto Statement, que es un objeto utilizado para enviar consultas SQL a la base de datos. Además de un ciclo "while", que itera sobre cada fila de "resultQuery" utilizando el método "next()" y se imprimen los valores de las columnas.

Y se tuvo en cuenta que en caso de que se produzca alguna excepción durante el proceso de conexión o de consulta a la base de datos, se capture la excepción y se imprima un mensaje en la consola.



# Clase: Lector de Archivo

Este código es un lector de archivos CSV, el primer archivo CSV contiene los pronósticos de los usuarios sobre los partidos y el segundo archivo CSV contiene los resultados oficiales de los partidos.

El programa comienza declarando algunas variables, como una matriz de String para almacenar los pronósticos de los usuarios y variables enteras para dar puntajes a los usuarios. También se define una cadena "X" para comparar con los pronósticos de los usuarios y dos rutas de archivo para los archivos CSV.

Luego, el programa usa un bloque try-catch para leer el primer archivo CSV con un **BufferedReader**. Así recorre cada línea del archivo CSV utilizando un bucle while y asignar variables a cada uno de los campos del archivo CSV.

Después de que se han leído y asignado todas las variables, se imprimen los pronósticos del usuario en la consola y se almacenan en la matriz de String resultUsuario. Además, se compara el pronóstico del usuario con la cadena "X" para determinar quién ganó el partido y se asigna el resultado correspondiente a la matriz resultUsuario.

# Clase: Lector de Archivo

El segundo archivo CSV se lee de manera similar al primero, después de que se han leído y asignado todas las variables, se compara el resultado oficial del partido con los pronósticos de los usuarios almacenados en la matriz `resultUsuario`. Si el pronóstico del usuario es correcto, se le otorga un punto al usuario. Además, se almacena el resultado oficial del partido en una matriz de String `resultOficial` para su posterior comparación con los pronósticos del usuario.

El programa también imprime el resultado oficial del partido en la consola y utiliza las mismas comparaciones que en la sección anterior para determinar quién ganó el partido y asignar el resultado correspondiente a la matriz `resultOficial`.

En resumen se leen los dos archivos CSV y almacena los datos leídos en matrices de String. Luego, compara los datos almacenados en las matrices para determinar el puntaje de los usuarios.

# Clase: equipo

La clase tiene dos atributos privados: “**nombre**” y “**descripción**”, que se establecen mediante un constructor que recibe dos argumentos. La clase también tiene métodos “**get**” y “**set**” para cada uno de los atributos.



# clase: pronóstico

Este código define una clase llamada “Pronóstico”. La clase tiene tres variables miembro privadas: “**nombre**”, “**jugado**” y “**ganador**”.

- nombre: (cadena de caracteres) representa el nombre del pronosticador.
- jugado: es un objeto de la clase “Partido” que representa el partido en el que se hace el pronóstico.
- ganador (cadena de caracteres): representa el equipo que el usuario cree que ganará el partido.

Los métodos “**get**” y “**set**” se utilizan para acceder y modificar las variables miembro.

El método “**mostrarDatos**” devuelve una cadena de caracteres que contiene el nombre del pronosticador y el equipo que cree que ganará el partido.

# clase: partido



La clase tiene cuatro atributos privados: “**equipo1**”, “**equipo2**”, “**golesEquipo1**” y “**golesEquipo2**”.

Hay dos constructores definidos en la clase. El primer constructor toma dos cadenas de caracteres que representan los nombres de los equipos. El segundo constructor toma los nombres de los equipos y los goles marcados por cada equipo.

La clase tiene varios métodos públicos para acceder (ej: `getEquipo1()` y `getEquipo2()`) y modificar (ej: `setGolesEquipo1()` y `setGolesEquipo2()`) los valores de los atributos.

El método **getEquipoGanador()** devuelve el nombre del equipo que ganó el partido, (o la cadena "Empate" en caso de empate).

Los métodos `getEquipoLocal()` y `getEquipoVisitante()` devuelven el nombre del equipo local y visitante, respectivamente.

El método `registrarGoles()` permite establecer los goles marcados por cada equipo.

# Clase: resultado

Este código define la clase "Resultado", que representa el resultado de un partido entre dos equipos.

La clase tiene cuatro atributos privados: "**ganador**", "**perdedor**", "**empate**" y "**resultado**". Los tres primeros son cadenas de texto que indican qué equipo ganó, perdió o empató el partido, respectivamente. El último es también una cadena de texto que indica el resultado del partido en formato "goles equipo1 - goles equipo2".

La clase tiene un constructor que recibe como parámetros el nombre del equipo ganador, el nombre del equipo perdedor y la cantidad de goles que marcó cada uno. Al momento de crear el objeto, se establece el valor del atributo "resultado" con la cantidad de goles de cada equipo.

La clase también tiene varios métodos de acceso y modificación (get y set) para cada uno de los atributos.

# Clase: ronda

La clase “Ronda” que tiene dos atributos: “**NumeroRonda**” y “**partido**”.

- El atributo “NumeroRonda” es de tipo “Integer” y representa el número de ronda.
- “partido” es un objeto de tipo “Partido” que representa el partido correspondiente a esta ronda.

La clase tiene dos constructores, uno sin argumentos y otro que recibe el número de ronda como parámetro. También tiene métodos getter y setter para ambos atributos, permitiendo acceder y modificar el número de ronda y el partido correspondiente.

# Clase: usuario

La clase “**Usuario**” representa a un usuario que participa en una competición de predicción de resultados de fútbol.

La clase tiene dos atributos: “usuario” que es el nombre del usuario y “id” que es un identificador único para el usuario.

El constructor de la clase recibe el nombre del usuario y lo almacena en el atributo “**usuario**”.

La clase también tiene dos métodos:

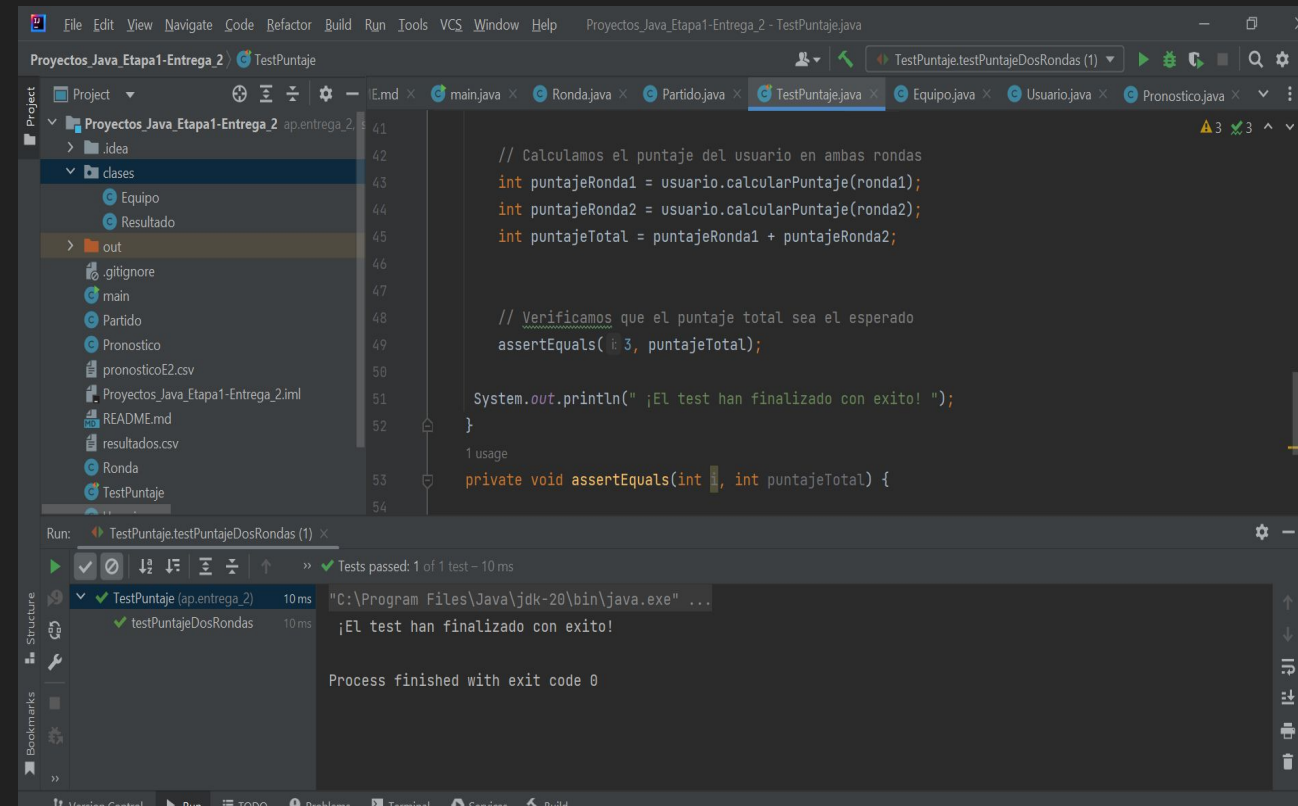
- **agregarPrediccion**: recibe como parámetros un objeto de la clase “Partido” y una predicción del resultado del partido para ese usuario. Este método no hace nada en este momento, pero al momento de mejorarlo se esperaría que este método almacenará la predicción del usuario en algún lugar por ejemplo.
- **calcularPuntaje**: recibe como parámetro un objeto de la clase “Ronda” y devuelve un entero que representa el puntaje del usuario en esa ronda. En este momento, el método simplemente devuelve el número de la ronda, pero al momento de mejorarlo se esperaría que este método calcule el puntaje del usuario comparando las predicciones del usuario con los resultados reales de los partidos en la ronda.



# Test

Se realizó un test para verificar el cálculo del puntaje del usuario. Se crean dos rondas con dos partidos cada una, se agregan resultados y predicciones para cada partido, se crea un usuario y se le asignan las predicciones. Luego, se calcula el puntaje del usuario para cada ronda y se suman para obtener el puntaje total. Finalmente, se verifica que el puntaje total sea el esperado utilizando el método `assertEquals`.

Si el test es exitoso, se imprime un mensaje indicando que el test fue exitoso.



The screenshot shows an IDE window titled "Proyectos\_Java\_Etapa1-Entrega\_2 - TestPuntaje.java". The left sidebar displays a project structure with folders like "clases" and "out", and files like "Equipo", "Resultado", "main", "Partido", "Pronostico", "pronosticoE2.csv", "Proyectos\_Java\_Etapa1-Entrega\_2.iml", "README.md", "resultados.csv", "Ronda", and "TestPuntaje". The main editor shows the following Java code:

```
// Calculamos el puntaje del usuario en ambas rondas
int puntajeRonda1 = usuario.calcularPuntaje(ronda1);
int puntajeRonda2 = usuario.calcularPuntaje(ronda2);
int puntajeTotal = puntajeRonda1 + puntajeRonda2;

// Verificamos que el puntaje total sea el esperado
assertEquals(3, puntajeTotal);

System.out.println(" ¡El test han finalizado con exito! ");
}
```

Below the code editor, the "Run" tab shows the test execution results:

```
Run: TestPuntaje.testPuntajeDosRondas (1)
Tests passed: 1 of 1 test - 10 ms
TestPuntaje (ap.entrega_2) 10 ms
  testPuntajeDosRondas 10 ms
  "C:\Program Files\Java\jdk-20\bin\java.exe" ...
  ¡El test han finalizado con exito!

Process finished with exit code 0
```

# Cuestionario



Argentina  
programa  
4.0

- ? ¿Pudieron completar todos los puntos propuestos en el Trabajo Final? Si no, ¿Qué les faltó agregar? ¿Cómo lo implementarían en su código?
- Se pudo completar los puntos.
  - Nos faltó implementar de lleno la estructura correcta del lenguaje, esto lo asociamos a falta de práctica. Es algo por mejorar, lo podríamos implementar de modo que se logrará una estructura más concisa y que establezca la relación entre clases, objetos y métodos de forma que se destaquen las ventajas del uso del lenguaje java.

# Exhibición



Enlace: [github ProyectosJava Etapa1.git](https://github.com/ProyectosJava/Etapa1.git)

# Bibliografia

- Lista de reproducción de youtube sobre programación Java inicial hasta POO:  
<https://www.youtube.com/playlist?list=PLWtYZ2ejMVJkjOuTCzIk61j7XKfpIR74K>
- Video de youtube sobre manipulación de archivos y directorios en Java:  
<https://www.youtube.com/watch?v=TBzGXYqFq3w>
- Lista de reproducción de youtube sobre creación de base de datos y tablas en MySQL:  
<https://www.youtube.com/playlist?list=PL1vDASG8ZwjBZMurvLLVjowpw2FyWFQNt>



Argentina  
programa  
4.0

¡Gracias!

Grupo N°6

Lourdes Aybar - Matias Ruben Candia - Franco Agustin Del For - Jefferson Gutierrez -

Carlos Antonio Patron - Franco Alanoca - Macarena Delgado Velez