

JUEGO SAMURAI

PÉREZ SELLERS, ALEJANDRO
ALEJANDRO PÉREZ SELLERS 1º de Daw

1. Introducción al juego de La batalla de los Samuráis.

En esta práctica he desarrollado un juego en Java llamado “Batalla de Samuráis”, donde dos jugadores se enfrentan con sus equipos de guerreros. Cada uno tiene 7 samuráis y debe repartir entre ellos 30 puntos de potencia, de este modo se debe de tener en cuenta los números introducidos para intentar ganar la batalla, ya que, si un jugador reparte mal los puntos, puede quedarse con samuráis muy débiles y perder.

Una vez los equipos están completados, empieza la batalla. Los combates se realizan en orden, empezando por un samurái aleatorio, y van luchando uno contra uno. En cada ronda gana el samurái de mayor potencia, si empatan los dos mueren. La partida termina cuando uno de los equipos pierde a más de la mitad de sus samuráis, y el otro se declara ganador.

La idea es que los samuráis de cada equipo se enfrenten uno a uno y el que tenga más fuerza gana la ronda. Al final, el equipo que tenga menos bajas es el que se lleva la victoria.

2. Explicación del código.

```
package org.example;

import java.util.Arrays;
import java.util.Random;
import java.util.Scanner;

public class practica_samurais {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Random aleatorio = new Random();
        int contadorjugador1=0;
        int contadorjugador2=0;
        //int []equipo1 = new int[7];
        //int []equipo2 = new int[7];
        //int []vector1 = {4,6,1,3,7,8,1};
        //int []vector2 = {5,6,1,7,8,2,3};
        // String equipo_1 = "1,3,5,5,7,8,2";
        // String equipo_2 = "1,3,5,5,7,8,1";
        // String [] subequipo_1 = equipo_1.split(",");
        // String [] subequipo_2 = equipo_1.split(",");
        boolean repetir1 = false;
        boolean repetir2 = false;
        int [] equiponumero_1 = new int[7];
        int [] equiponumero_2 = new int[7];
    }
}
```

Se comienza el punto de entrada declarando las variables **random aleatorio = new random** donde se usará para elegir al azar con que samurái se empieza la batalla.

Mediante **int contadorjugador1=0;** y **int contadorjugador2=0;** se almacenan el número de combates ganados por cada jugador.

Después **boolean repetir1 = false; y boolean repetir2 = false;** controlan si se deben repetir las introducciones de datos de los equipos, en caso de que se introduzca cifras incorrectas, por ejemplo, suma distinta de 30.

También se crea dos arrays de tamaño 7 para cada equipo, ya que cada equipo debe de estar formado por 7 guerreros.

```
do {
    System.out.println("Equipo 1");
    System.out.println("introduce potencia de los samurais separado
por espacios:");
    String equipo1 = sc.nextLine();
    String[] subequipo_1 = equipo1.split(" ");
    System.out.println(Arrays.toString(subequipo_1));
    repetir1 = false;
    int suma1 = 0;
```

Se pide al jugador que introduzca las potencias de los 7 samuráis separadas por espacios, se usa el método Split para dividir la cadena en trozos, separando los valores por espacio y guardándolos en un array string[].

```
for (int i = 0; i < subequipo_1.length; i++) {
    int numerol = Integer.parseInt(subequipo_1[i]);
    suma1 += numerol;
    equiponumero_1[i] = numerol;
}
```

Se recorre el array, convirtiendo cada valor a numero entero con **Integer.parseInt()**. Además, se suman los valores suma1 y se va guardando cada numero dentro del array **equiponumero_1**.

```
if (subequipo_1.length != 7) {
    System.out.println("Error. El equipo no tiene 7 samurais.");
    repetir1 = true;
} else if (suma1 != 30) {
    System.out.println("Error. El equipo no suma 30");
    repetir1 = true;
} else {
    System.out.println("El equipo esta completo");
    //repetir = false;
}
```

En este bloque se validan las entradas, si el jugador no introduce exactamente 7 números, se muestra un error, si la suma total no es 30 también se muestra un error y si todo esta correcto, aparece un mensaje donde confirma que el equipo esta completo.

Por ultimo se aplica un bucle do-while donde repite la entrada hasta que los datos sean correctos.

```

do {
    System.out.println("Equipo 2");
    System.out.println("introduce potencia de los samurais");
    String equipo_2 = sc.nextLine();
    String[] subequipo_2 = equipo_2.split(" ");
    repetir2 = false;
    int suma2 = 0;
    for (int i = 0; i < subequipo_2.length; i++) {
        int numero2 = Integer.parseInt(subequipo_2[i]);
        suma2 += numero2;
        equiponumero_2[i] = numero2;
    } if (subequipo_2.length != 7) {
        System.out.println("Error. El equipo no tiene 7 samurais");
        repetir2=true;
    } else if (suma2 != 30) {
        System.out.println("Error. La potencia total no suma 30.");
        repetir2 = true;
    } else {
        System.out.println("El equipo esta completo");
        //repetir = false;
    }
} while (repetir2 == true);

```

El bloque para el equipo 2 repite exactamente la misma lógica que el equipo 1, así que no hace falta comentarlo.

- Inicio de la batalla -

```

- int inicioguerra = aleatorio.nextInt(7)+1;
  System.out.println("la batalla se inicia con el samurai numero "
    +inicioguerra);

```

Se genera un numero aleatorio entre 1 y 7 para decidir con que samurái comienza la guerra, por lo tanto, cada guerra es distinta.

```

for (int i =0; i<7; i++){
    int turno = (inicioguerra -1 + i)%7;
    int samurai_1 = equiponumero_1[turno];
    int samurai_2 = equiponumero_2[turno];
}

```

Este bucle recorre los 7 combates, empezando desde el samurái elegido de manera arbitraria. El operador módulo % permite que al llegar al final del array, vuelva al inicio sin pasarse de índice.

```

int samurai_1 = equiponumero_1[turno];
int samurai_2 = equiponumero_2[turno];

```

Se guardan las potencias de los samuráis que van a enfrentarse en esa ronda.

```

if (samurai_1>samurai_2){
    System.out.println("Samurai "+(turno+1)+" gana equipo 1
"+samurai_1+" vs "+samurai_2);
    contadorjugador1++;
} else if (samurai_2>samurai_1) {
    System.out.println("samurai "+(turno+1)+" gana equipo 2
"+samurai_1+" vs "+samurai_2);
    contadorjugador2 ++;

}else {
    System.out.println("Empaque, los dos samurais caen en batalla");
}

```

En este bloque se comparan las potencias, si el samurai del equipo 1 tiene mas potencia gana y se suma un punto a su contador, si el del equipo 2 tiene más potencia se suma un punto al contador, sin embargo si empatan ambos mueren.

```

}
if (contadorjugador1>contadorjugador2){
    System.out.println("El equipo 1 ha ganado");
} else if (contadorjugador1<contadorjugador2) {
    System.out.println("el equipo 2 ha ganado");
}else{
    System.out.println("Ha habido un empate");
}

```

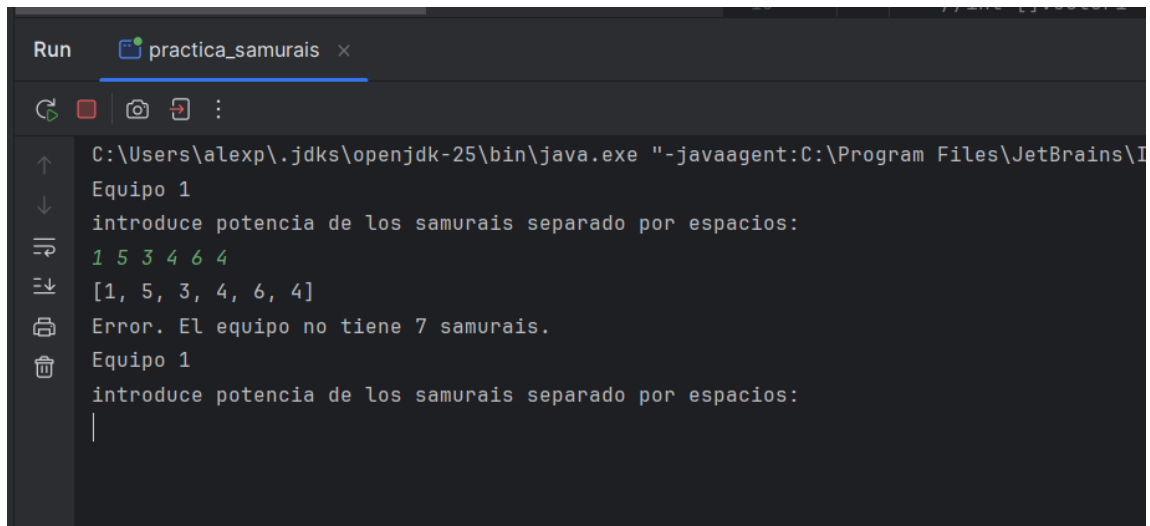
Al final del programa se comparan las victorias acumuladas por cada equipo, el que tenga mas combates ganados es el vencedor, si los dos tiene la misma cantidad se declara como empate.

3. Conclusión

Para concluir la memoria, he tenido una serie de problemas con la variable **int sum1=0; y int suma2=0;** que he podido solventar sacando la variable del bucle for de ambos equipos, lo que me ocurría es que no empezaba en cero cada vez que me saltaba el aviso de la potencia total debe ser 30, se quedaba con el ultimo sumatorio. También he tenido otro problema, ya que no se me ocurría como poder recorrer del inicio aleatorio y dar la vuelta al final **int turno = (inicioguerra -1 + i)%7;** esto tuve que buscarlo por internet, lo que hace es que los duelos empiezan en inicioguerra y siguen en orden, pero cuando se llega al final del equipo se debe volver al principio (posición 0), es decir, cuando llega a una posición donde es divisible entre 7 vuelve al principio. Las demás dudas que he ido teniendo en la practica las he podido resolver con los apuntes y ejercicios de clase.

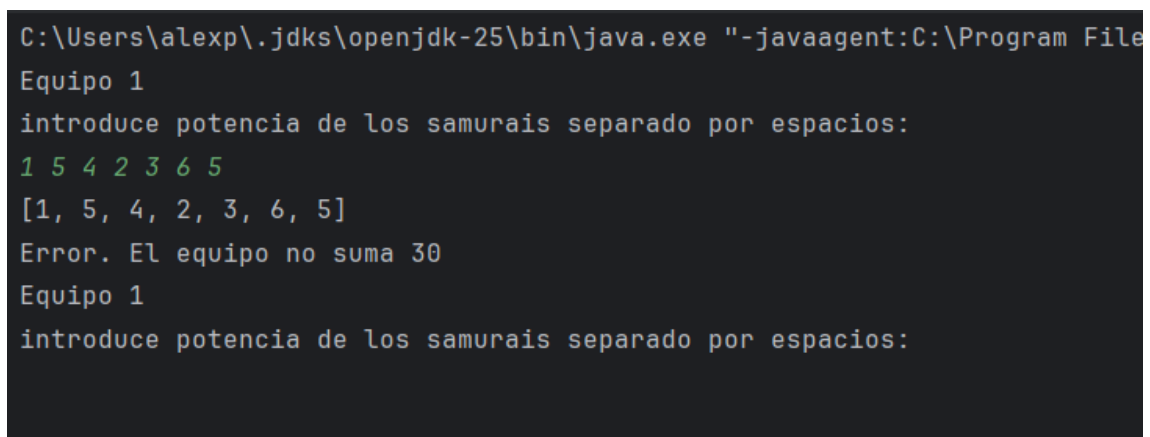
4. Pruebas de funcionamiento.

- Se introducen un numero de samuráis distinto a 7



```
Run practica_samurais x
C:\Users\alexp\.jdk\openjdk-25\bin\java.exe "-javaagent:C:\Program Files\JetBrains\I
Equipo 1
introduce potencia de los samurais separado por espacios:
1 5 3 4 6 4
[1, 5, 3, 4, 6, 4]
Error. El equipo no tiene 7 samurais.
Equipo 1
introduce potencia de los samurais separado por espacios:
|
```

- Suma distinta de 30



```
C:\Users\alexp\.jdk\openjdk-25\bin\java.exe "-javaagent:C:\Program File
Equipo 1
introduce potencia de los samurais separado por espacios:
1 5 4 2 3 6 5
[1, 5, 4, 2, 3, 6, 5]
Error. El equipo no suma 30
Equipo 1
introduce potencia de los samurais separado por espacios:
```

- Se introducen valores no numéricos

```
C:\Users\alexp\.jdk\openjdk-25\bin\java.exe "-javaagent:C:\Progr
Equipo 1
introduce potencia de los samurais separado por espacios:
1 8 7 k 6 8 7
[1, 8, 7, k, 6, 8, 7]
error introduce valores correctos
Error. El equipo no suma 30
Equipo 1
introduce potencia de los samurais separado por espacios:
|
```

- Se introduce 30 y se rellena con 0

```
Equipo 1
introduce potencia de los samurais separado por espacios:
30 0 0 0 0 0 0
[30, 0, 0, 0, 0, 0, 0]
El equipo esta completo
Equipo 2
introduce potentiate de los samurais
0 0 0 0 0 0 30
El equipo esta completo
la batalla se inicia con el samurai numero 1
Samurai 1 gana equipo 1 30 vs 0
Empaque, los dos samurais caen en batalla
Empaque, los dos samurais caen en batalla
Empaque, los dos samurais caen en batalla
Empaque, los dos samurais caen en batalla
Empaque, los dos samurais caen en batalla
samurai 7 gana equipo 2 0 vs 30
Ha habido un empate
```

- Prueba de empate entre samuráis.

```
Equipo 1
introduce potencia de los samurais separado por espacios:
6 5 4 3 2 5 5
[6, 5, 4, 3, 2, 5, 5]
El equipo esta completo
Equipo 2
introduce potencia de los samurais
5 6 5 4 3 5 2
El equipo esta completo
la batalla se inicia con el samurai numero 1
Samurai 1 gana equipo 1 6 vs 5
samurai 2 gana equipo 2 5 vs 6
samurai 3 gana equipo 2 4 vs 5
samurai 4 gana equipo 2 3 vs 4
samurai 5 gana equipo 2 2 vs 3
Empaque, los dos samurais caen en batalla
Samurai 7 gana equipo 1 5 vs 2
el equipo 2 ha ganado

Process finished with exit code 0
```