Ejercicio Opcional
Bingo

Alumno
Sergio Gárgoles Carballo

Contenidos

Planteamiento 3
Boleto.java 3
Método constructor 3
Función mostrarBoleto 4
Función comprobarLinea 5
Función comprobarBingo 6
Función marcarNumero 6
Bingo.java 7
Función numAleatorio 7
Función partidaBingo 7
Ejecutar.java 9
Dificultades encontradas 10

Planteamiento

- *Bingo (generación de los cartones para un número de jugadores y sorteo posterior)*

Inicialmente planteé un menú donde podías elegir mostrar tu boleto, pasar a la siguiente bola o saltar al final del sorteo, pero después de crearlo, me supuso bastante dificultad añadirlo todo, así que lo simplifiqué en que solo eliges los jugadores y mostrase los boletos y los ganadores de la línea y el bingo.

Boleto.java

Primero, pensé en la generación del boleto, para el cual hice una clase con 3 variables miembro, una para el array del boleto de dos dimensiones y otra para las columnas y filas.

```
package src;
public class Boleto {

    private int boleto[][];
    private final int filas = 3;
    private final int columnas = 5;
```

Método constructor

Generé el método constructor donde se iguala la variable que se va a utilizar a un array que tenga las filas y columnas establecidas:

```
public Boleto() {
    this.boleto = new int[filas][columnas];
    for (int i = 0; i < filas; i++) {
        for (int j = 0; j < columnas; j++) {
            this.boleto[i][j] = Bingo.numAleatorio();
        }
    }

}
```

En cada columna se genera un número aleatorio el cual se verá más adelante en la clase Bingo

Función mostrarBoleto

En esta función se recorren todas las posiciones del array, escribiendo XX si en esta posición se encuentra un -1, el cual significa que este número ha salido en el sorteo, sino escribe el número con formato “%2d ” con el que se escriben dos decimales, si uno de estos está vacío lo sustituye por un cero y escribe un espacio después:

```
public void mostrarBoleto() {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 5; j++) {
            if (this.boleto[i][j] == -1) {
                System.out.print("XX" + " ");
            }else {
                System.out.printf("%2d ", boleto[i][j]);
            }
        }
        System.out.println();
    }
}
```

Función comprobarLinea

Aquí se comprueba si se ha hecho una línea en el boleto, para ello creo una variable para comparar con el número de columnas que hay, la cual inicializo a 0 en cada fila, para que esta se compruebe en todas las filas.
Como se ha dicho antes, si hay un número marcado significa que hay un -1 en esa posición, con lo cual sumará uno a la variable cantidadNum.
Si al finalizar la línea esta variable se iguala a las columnas, devolverá un true, si pasa por todas las filas y no encuentra ninguna fila completa, devolverá un false.

```
public boolean comprobarLinea() {

    int cantidadNum = 0;
    for (int i = 0; i < filas; i++) {
        cantidadNum = 0;
        for (int j = 0; j < columnas; j++) {
            if (this.boleto[i][j] == -1) {
                cantidadNum++;
            }
        }
        if (cantidadNum == columnas) {
            return true;
        }
    }
    return false;


}
```

Función comprobarBingo

Similar a la anterior función, se recorre el boleto sumando 1 a cantidadNum, si al terminar esto es igual a multiplicar las filas por las columnas, devolverá un verdadero (de tal manera que, si es necesario cambiar las filas o las columnas, esta parte del código no será necesaria cambiarla (Sinceramente esto me lo ha dicho un amigo, a mí no se me ocurrió, pero lo utilizaré porque es muy útil))

```
public boolean comprobarBingo() {

    int cantidadNum = 0;
    for (int i = 0; i < filas; i++) {
        for (int j = 0; j < columnas; j++) {
            if (this.boleto[i][j] == -1) {
                cantidadNum++;
            }
        }
    }

    return (cantidadNum == filas * columnas);


}
```

Función marcarNumero

Por último y probablemente la más importante, esta función recorre el boleto y marca con -1 si se iguala a el número sacado, el cual es una variable la cual hay que introducir desde fuera, esto saldrá de la clase Bingo.

```
public void marcarNumero(int numeroSacado){

    for (int i = 0; i < filas; i++) {
        for (int j = 0; j < columnas; j++) {
            if (this.boleto[i][j] == numeroSacado) {
                this.boleto[i][j] = -1;
            }
        }
    }

}
```

Bingo.java

En esta clase se realizará el sorteo de números y se utilizarán partes de Boleto.java.
En esta clase no se creará ningún objeto como tal sino que utilizará información que depende de otras clases, por lo cual no necesita de variables método ni clases constructoras.

Función numAleatorio

Esta función en principio estaba ideada para Boleto.java, pero en esta clase tenía más sentido que estuviese, porque es parte del juego.
Inicializo una variable en -1 para que entre al while de debajo el cual solo funciona si esta variable "numeroDePrueba" es menor o igual a 0 o mayor de 100.
Dentro de este bucle, se genera un número aleatorio multiplicado por 100 para que salgan todos los números posibles del 1 al 100, se trunca este número y se pasa a integer mediante (int).
Esta función devolverá este número.

```
public static int numAleatorio() {
    int numeroDePrueba = -1;
    while (numeroDePrueba <= 0 || numeroDePrueba > 100) {
        numeroDePrueba = (int) Math.floor(Math.random() * 100);
    }

    return numeroDePrueba;
}
```

Función partidaBingo

La función que desarrolla el juego, para esta función es necesario introducirle un array de boletos, que será introducido en Ejecutar.java.
Empieza creando un array de booleanos el cual tiene 100 posiciones, lo cual significa los 100 números que pueden salir en el bingo, dos booleanos más para decir si ha salido bingo o línea y para introducirnos en el bucle principal ponemos todo en falso para entrar en este.

La primera parte genera un numero aleatorio, el cual comprobará si esa posición en el array de booleanos ha salido, si es así, lo cual significa que esta posición es true, genera otro número aleatorio. Si este número no ha salido, se guarda en la variable numeroDeMomento y se cambia esa posición en el array a true para simbolizar que ya ha salido.

```
public static void partidaBingo(Boleto[] boletos) {

    boolean[] todosLosNumeros = new boolean[100];
    boolean linea = false;
    boolean bingo = false;
    while (!bingo) {
        //Crea un número aleatorio y comprueba si está repetido, si es así genera otro
        //Posteriormente marca este número como verdadero
        int numeroDeMomento;

        do {
            numeroDeMomento = numAleatorio();
        } while (todosLosNumeros[numeroDeMomento]);

        todosLosNumeros[numeroDeMomento] = true;
```

Habiendo sacado el número, este debe marcarse en todos los boletos de los jugadores, para ello recorreremos la longitud del array, utilizando la propiedad del objeto Boleto marcarNumero pasándole este número que ha salido.

```
for (int i = 0; i < boletos.length; i++) {
    boletos[i].marcarNumero(numeroDeMomento);
}
```

Ahora pasaremos a las comprobaciones, empezando por la línea, la cual mientras no haya habido una línea antes, inicializado en false previamente, recorre todos los boletos, utilizando la propiedad del objeto Boleto, comprobarLinea, la cual, si devuelve un true, cambia el estado de línea a true para que no vuelva a introducirse en este if e indica de quién ha sido la línea y muestro el boleto de ese jugador.

```
if (!linea) {
    for (int i = 0; i < boletos.length; i++) {
        if (boletos[i].comprobarLinea()) {
            linea = true;
            System.out.println();
            System.out.println("Línea del jugador " + (i+1));
            boletos[i].mostrarBoleto();
        }
    }
}
```

Similar al anterior punto, recorremos el array de boletos, utilizando la propiedad del objeto Boleto, comprobarBingo, la cual, si devuelve un true, cambia bingo a true para salir de este bucle principal e imprime el jugador que ha sacado bingo y su boleto.

```
for (int i = 0; i < boletos.length; i++) {
    if (boletos[i].comprobarBingo()) {
        bingo = true;
        System.out.println();
        System.out.println("Bingo del jugador " + (i+1));
        boletos[i].mostrarBoleto();
    }
}
```

Ejecutar.java

Ahora pasamos a la ejecución del programa, para ello recogemos de la consola el número de jugadores, he intentado meter control de errores con try catch y con el scanner de java, porque tiene métodos para saber si has metido un integer u otra cosa, pero no lo he conseguido, así que se hace una entrada de datos simple en la que se asume que el usuario no introduce un dato inválido. Posteriormente se crea un array del objeto Boleto con el número que introducimos anteriormente.

Inicializamos los objetos, recorriendo el array que hemos generado y en cada posición, le indicamos que genere un objeto en esta, lo imprimimos para mostrar el boleto en la consola utilizando la propiedad mostrarBoleto.

Después de haberlos generado, le pasamos al método Bingo.partidaBingo el array que hemos creado para que ejecute este método.

```
int numBoletos = Teclado.leerInt("Introduzca el número de jugadores: ");
Boleto[] boletos = new Boleto[numBoletos];


for (int i = 0; i < boletos.length; i++) {
    boletos[i] = new Boleto();
    System.out.println();
    System.out.println("Boleto " + (i + 1));

    boletos[i].mostrarBoleto();
}
System.out.println();
Bingo.partidaBingo(boletos);
```

Dificultades encontradas

Me ha sido muy complicado el entender como hacer funcionar un array dentro del objeto y posteriormente sacarlo.

Utilizar varias clases era lioso de entender porque si con una ya me confundo, utilizar dos ya me perdía constantemente, lo he conseguido gracias a bastantes horas y a mi amigo anteriormente mencionado (no es chatgpt por si acaso), que tiene bastante idea del tema y consiguió que lo comprendiese, aun así, me sigue pareciendo como estar en una cuerda floja.

Saber qué tipos de variable y de métodos utilizar es muy complicado, porque pensaba que void se utilizaba poco y no le había dado demasiada vuelta, tras ver esto, lo he utilizado mucho más y es el que más sencillo veía de utilizar.