



Fundamentos de POO

Ingeniería en
Tecnologías de la
Información

EDER LÓPEZ VILLARREAL

ERIC JHONATHAN ANAYA
MARQUEZ

5H T/V

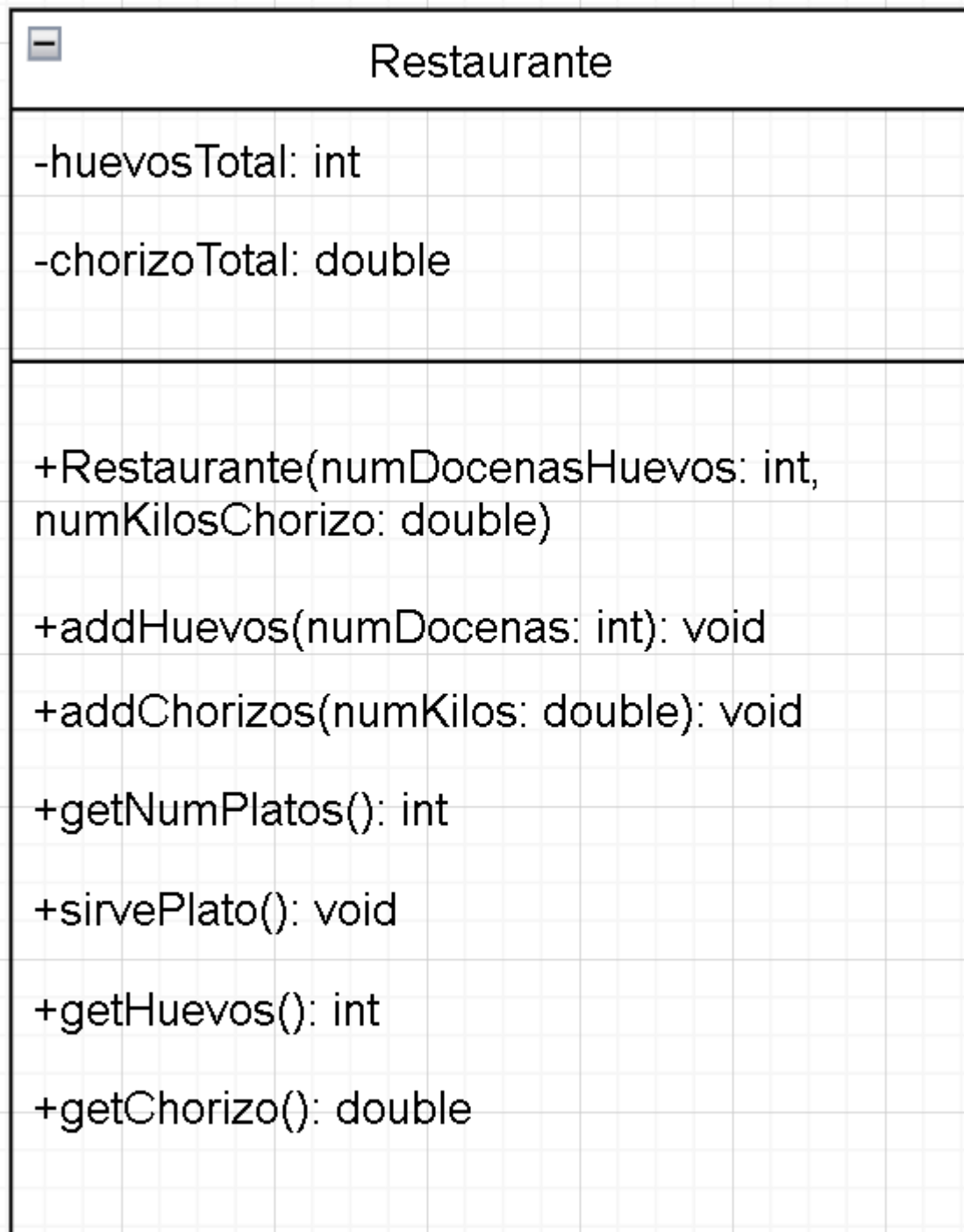
DOCENTE

MTRO. SAUL OLAF LOAIZA

Desarrollo clase 1 Clase
Restaurante

25/01/2024

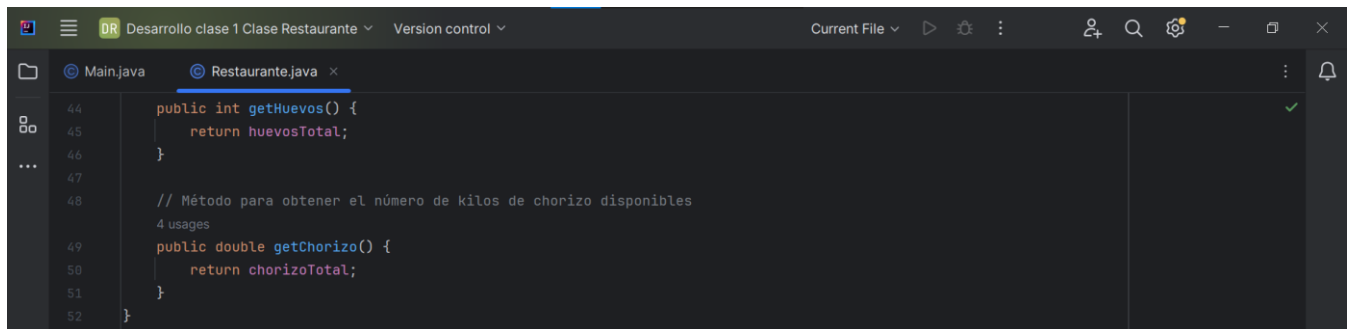
Diagrama UML del clasificador de clase Restaurante



Código de clase Restaurante

```
1 package POO;
2
3 public class Restaurante {
4     private int huevosTotal; // Total de huevos, incluyendo las docenas parcialmente utilizadas
5     private double chorizoTotal; // Total de kilos de chorizo
6
7     // Constructor
8     public Restaurante(int numDocenasHuevos, double numKilosChorizo) {
9         huevosTotal = numDocenasHuevos * 12; // Convertimos las docenas de huevos a huevos individuales
10        chorizoTotal = numKilosChorizo;
11    }
12
13    // Método para incrementar el número de docenas de huevos
14    public void addHuevos(int numDocenas) {
15        huevosTotal += numDocenas * 12; // Convertimos las docenas de huevos a huevos individuales
16    }
17
18    // Método para incrementar el número de kilos de chorizo
19    public void addChorizos(double numKilos) {
20        chorizoTotal += numKilos;
21    }
22}
```

```
21 }
22
23 // Método para obtener el número de platos de huevos con chorizo disponibles
24 public int getNumPlatos() {
25     // Calcula el número de platos basado en los huevos y el chorizo disponibles
26     int numPlatosHuevos = huevosTotal / 2; // Cada plato necesita 2 huevos
27     int numPlatosChorizo = (int)(chorizoTotal / 0.2); // Cada plato necesita 0.2 kg de chorizo
28     // Devuelve el número mínimo entre el número de platos de huevos y el número de platos de chorizo
29     return Math.min(numPlatosHuevos, numPlatosChorizo);
30 }
31
32 // Método para servir un plato y disminuir las existencias
33 public void sirvePlato() {
34     if (huevosTotal >= 2 && chorizoTotal >= 0.2) { // Comprobamos si hay suficientes huevos y chorizo
35         huevosTotal -= 2; // Se consumen 2 huevos
36         chorizoTotal -= 0.2; // Se consumen 0.2 kg de chorizo
37         System.out.println("Plato servido. Quedan " + huevosTotal + " huevos y " + chorizoTotal + " kg de chorizo.");
38     } else {
39         System.out.println("No se puede servir el plato. No hay suficientes ingredientes.");
40     }
41 }
42
43 // Método para obtener el número de huevos disponibles
44 }
```

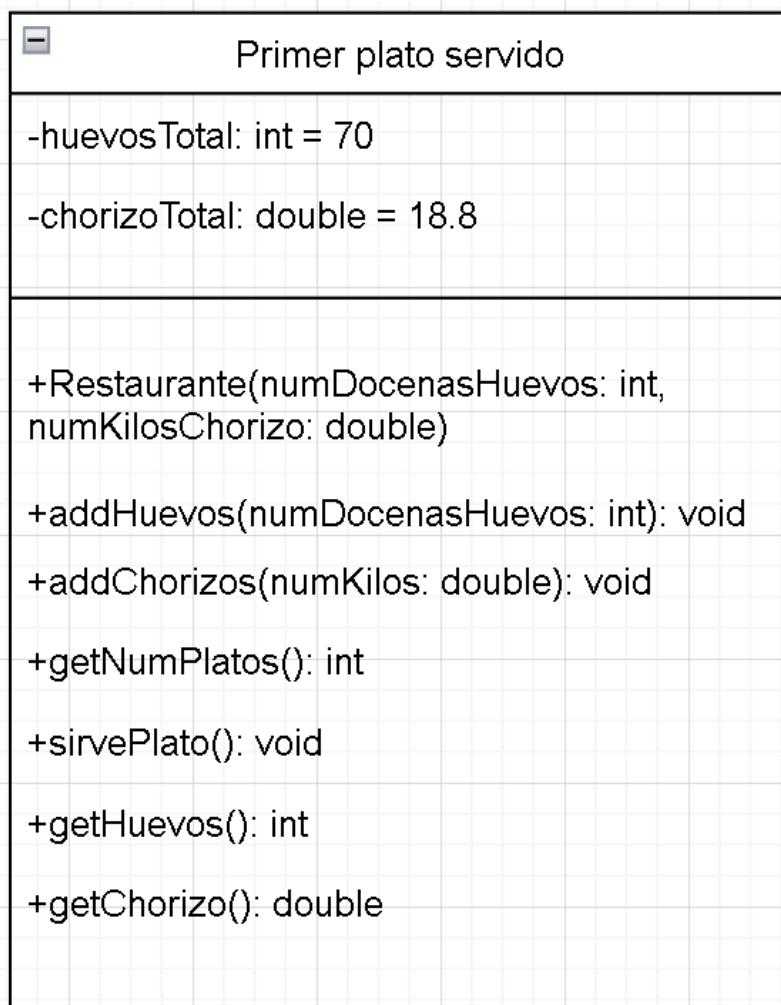


The screenshot shows an IDE window with a dark theme. The title bar at the top reads "DR Desarrollo clase 1 Clase Restaurante" and "Version control". The menu bar includes "Current File", a play button, a settings gear, a vertical ellipsis, a search icon, a plus icon, a magnifying glass, a gear with a plus, a minus, a square, and a close button. The editor has two tabs: "Main.java" and "Restaurante.java", with the latter being active. On the left, a file explorer shows a folder icon and a list of files including "Main.java" and "Restaurante.java". The code in the editor is as follows:

```
44 public int getHuevos() {
45     return huevosTotal;
46 }
47
48 // Método para obtener el número de kilos de chorizo disponibles
49 // 4 usages
50 public double getChorizo() {
51     return chorizoTotal;
52 }
```

A green checkmark is visible in the right margin next to line 45.

Diagrama UML del objeto para verificar la prueba indicada



Actualización huevosTotal

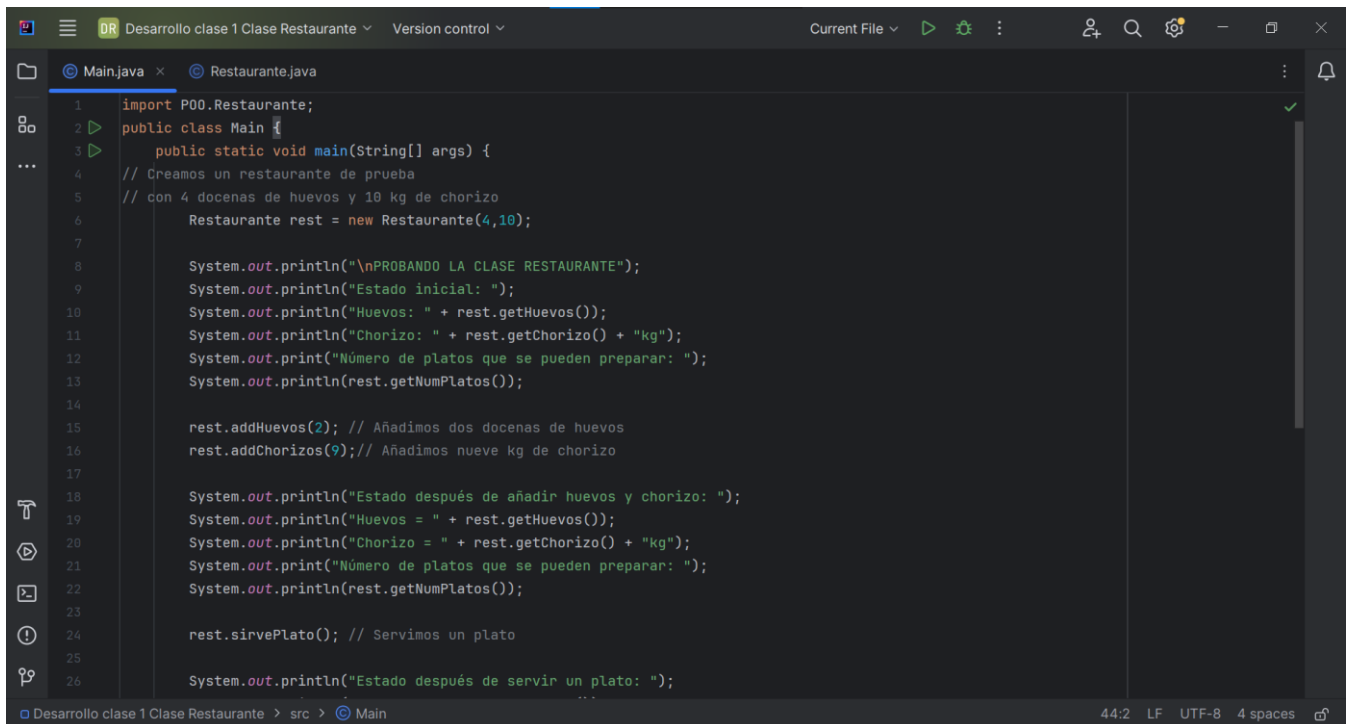
En el constructor, `numDocenasHuevos` representa el número de docenas de huevos, que se multiplica por 12 para obtener el total de huevos. Por lo tanto, si se pasa 4 como `numDocenasHuevos`, se calcula $4 * 12 = 48$, lo que establece `huevosTotal` en 48. Sin embargo, luego se incrementa `huevosTotal` en 2 docenas más utilizando el método `addHuevos(2)` en el método `main()`. Por lo tanto, el valor total de `huevosTotal` se convierte en $48 + 2 * 12 = 72$. Además, se sirve un plato en el método `sirvePlato()`, lo que reduce `huevosTotal` en otros 2, llegando a $72 - 2 = 70$. Y así son las mismas operaciones para cuando se sirven los 5 platos restantes.

Actualización chorizoTotal

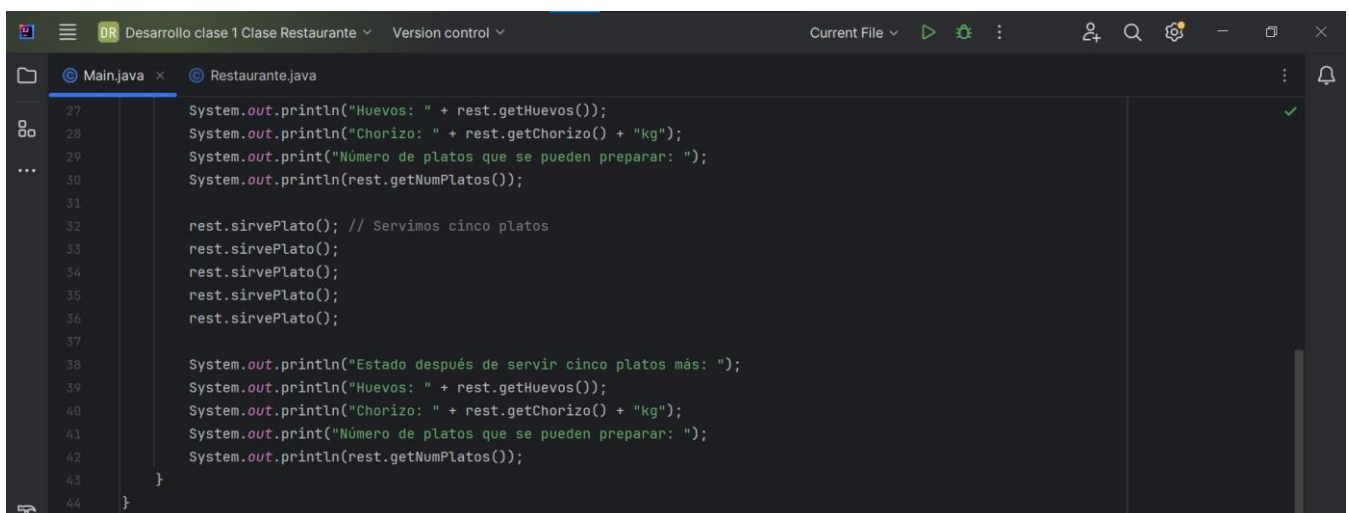
El constructor establece el valor de `chorizoTotal` según el parámetro `numKilosChorizo` que se pasa. Luego, durante la ejecución del método `main()`, se añaden 9 kilos adicionales de chorizo mediante el método `addChorizos(9)`. Después de servir un plato, se consume 0.2 kg de chorizo. Por lo tanto, se reduciría en 0.2 kg. Constructor: `chorizoTotal` se establece en 10 (valor pasado como argumento). Método `addChorizos(9)`: Se añaden 9 kilos adicionales, por lo que `chorizoTotal` se convierte en $10 + 9 = 19$.

Método `sirvePlato()`: Se consume 0.2 kg de chorizo, por lo que `chorizoTotal` se reduce en 0.2, quedando en $19 - 0.2 = 18.8$. Y así son las mismas operaciones para cuando se sirven los 5 platos restantes.

Clase principal Main con la prueba solicitada

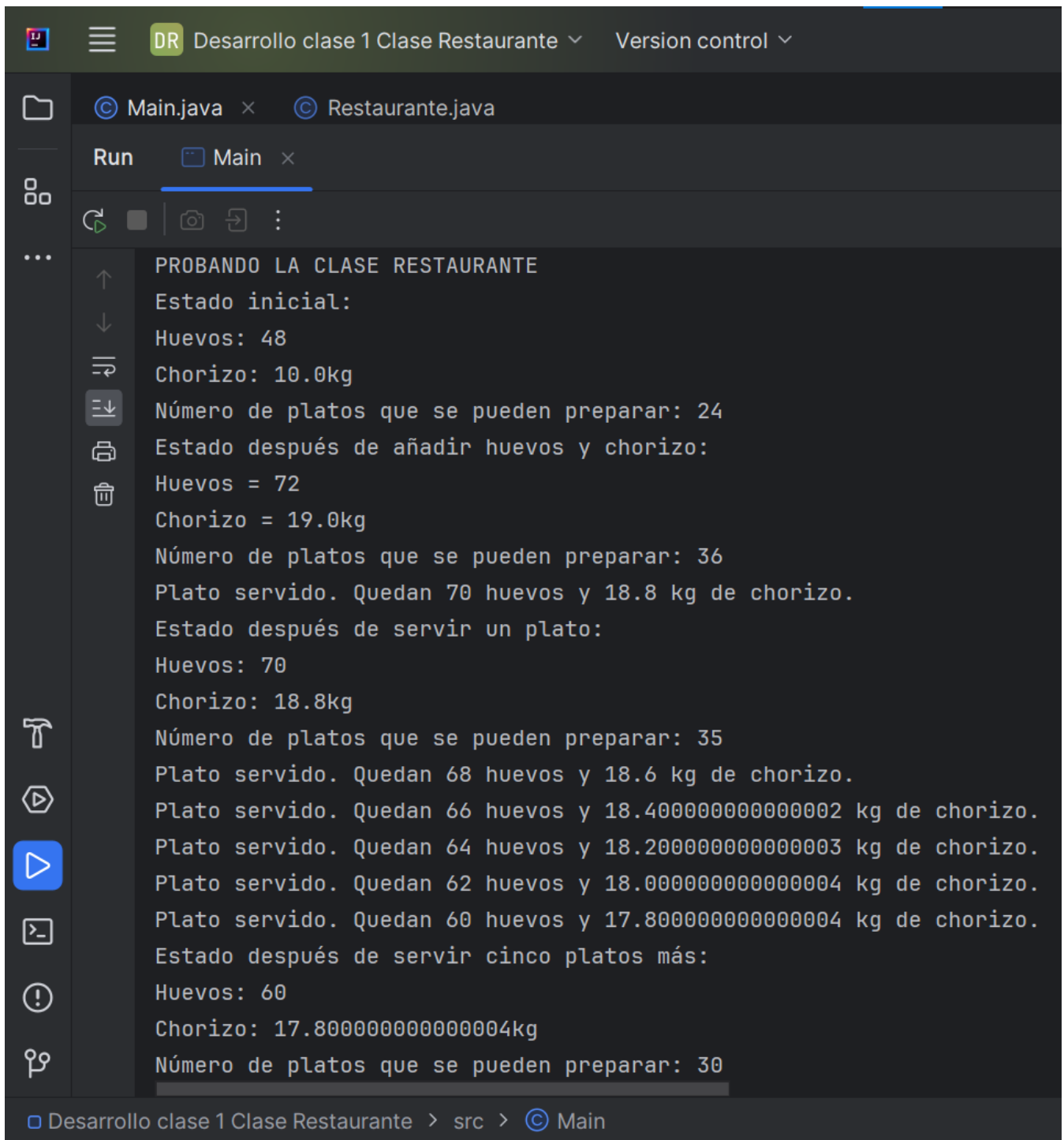


```
1 import P00.Restaurante;
2 public class Main {
3     public static void main(String[] args) {
4         // Creamos un restaurante de prueba
5         // con 4 docenas de huevos y 10 kg de chorizo
6         Restaurante rest = new Restaurante(4,10);
7
8         System.out.println("\nPROBANDO LA CLASE RESTAURANTE");
9         System.out.println("Estado inicial: ");
10        System.out.println("Huevos: " + rest.getHuevos());
11        System.out.println("Chorizo: " + rest.getChorizo() + "kg");
12        System.out.print("Número de platos que se pueden preparar: ");
13        System.out.println(rest.getNumPlatos());
14
15        rest.addHuevos(2); // Añadimos dos docenas de huevos
16        rest.addChorizos(9); // Añadimos nueve kg de chorizo
17
18        System.out.println("Estado después de añadir huevos y chorizo: ");
19        System.out.println("Huevos = " + rest.getHuevos());
20        System.out.println("Chorizo = " + rest.getChorizo() + "kg");
21        System.out.print("Número de platos que se pueden preparar: ");
22        System.out.println(rest.getNumPlatos());
23
24        rest.sirvePlato(); // Servimos un plato
25
26        System.out.println("Estado después de servir un plato: ");
```



```
27        System.out.println("Huevos: " + rest.getHuevos());
28        System.out.println("Chorizo: " + rest.getChorizo() + "kg");
29        System.out.print("Número de platos que se pueden preparar: ");
30        System.out.println(rest.getNumPlatos());
31
32        rest.sirvePlato(); // Servimos cinco platos
33        rest.sirvePlato();
34        rest.sirvePlato();
35        rest.sirvePlato();
36        rest.sirvePlato();
37
38        System.out.println("Estado después de servir cinco platos más: ");
39        System.out.println("Huevos: " + rest.getHuevos());
40        System.out.println("Chorizo: " + rest.getChorizo() + "kg");
41        System.out.print("Número de platos que se pueden preparar: ");
42        System.out.println(rest.getNumPlatos());
43    }
44 }
```

Verificación de prueba



```
PROBANDO LA CLASE RESTAURANTE
Estado inicial:
Huevos: 48
Chorizo: 10.0kg
Número de platos que se pueden preparar: 24
Estado después de añadir huevos y chorizo:
Huevos = 72
Chorizo = 19.0kg
Número de platos que se pueden preparar: 36
Plato servido. Quedan 70 huevos y 18.8 kg de chorizo.
Estado después de servir un plato:
Huevos: 70
Chorizo: 18.8kg
Número de platos que se pueden preparar: 35
Plato servido. Quedan 68 huevos y 18.6 kg de chorizo.
Plato servido. Quedan 66 huevos y 18.400000000000002 kg de chorizo.
Plato servido. Quedan 64 huevos y 18.200000000000003 kg de chorizo.
Plato servido. Quedan 62 huevos y 18.000000000000004 kg de chorizo.
Plato servido. Quedan 60 huevos y 17.800000000000004 kg de chorizo.
Estado después de servir cinco platos más:
Huevos: 60
Chorizo: 17.800000000000004kg
Número de platos que se pueden preparar: 30
```