

CLEAN CODE

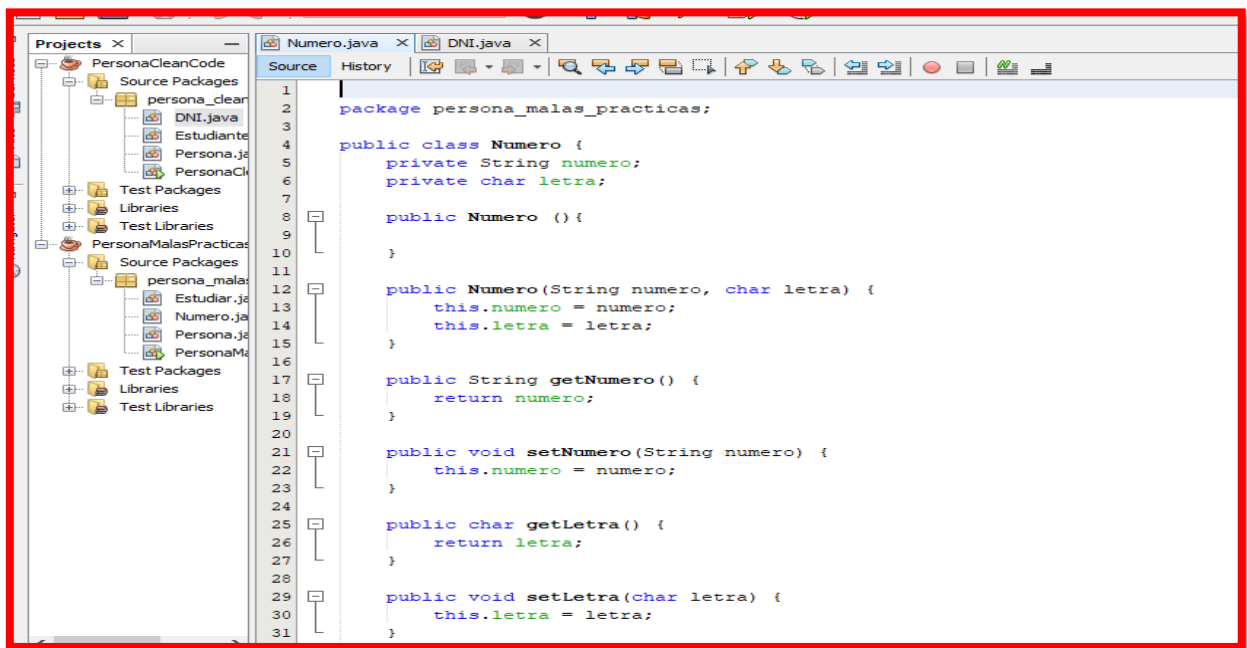
Se han creado dos proyectos en java, uno que contiene todas las malas prácticas que se llama **PersonaMalasPracticas** (de ahora en adelante proyecto1) y otro que tiene todo como realmente debería hacerse siguiendo las reglas del Clean Code, llamado **PersonaCleanCode** (de ahora en adelante proyecto 2).

Vamos a ordenar la práctica en el mismo orden que hemos dado los bloques y dando ejemplos de cada caso.

Bloque 1: Nombres

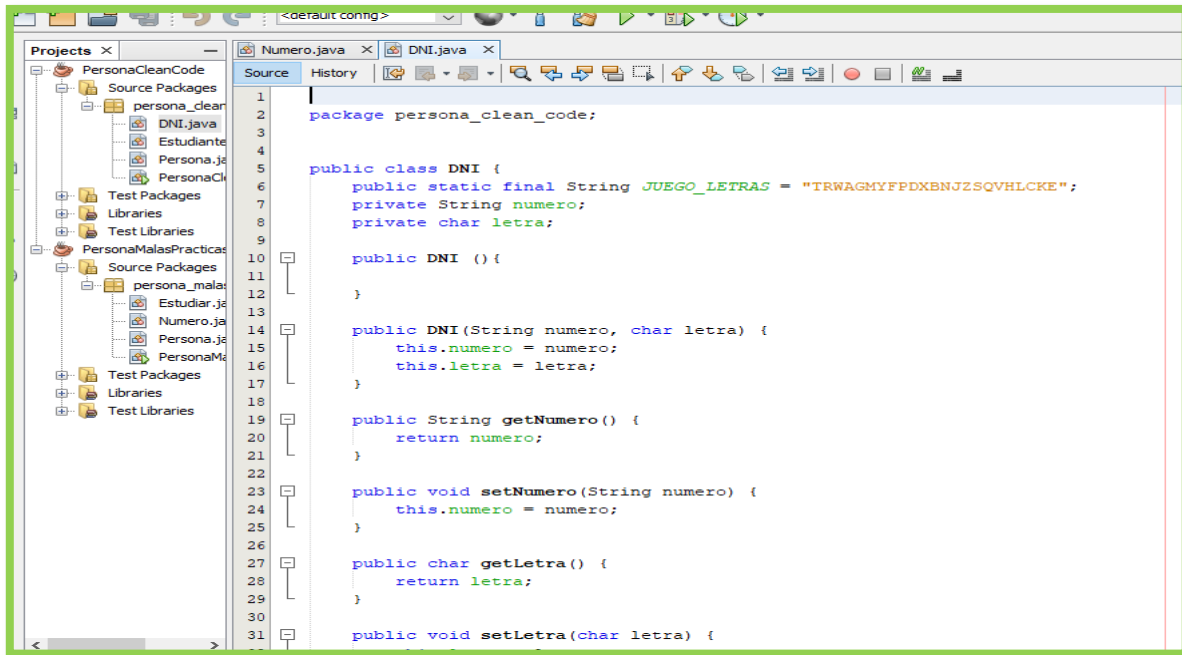
Usar nombre con significado

En el proyecto 1 tenemos el objeto **Numero.java** y su nombre en este caso no es suficientemente representativo, ya que lo que contiene es un DNI y toda su funcionalidad.



```
1 package persona_malas_practicas;
2
3
4 public class Numero {
5     private String numero;
6     private char letra;
7
8     public Numero () {
9
10    }
11
12    public Numero(String numero, char letra) {
13        this.numero = numero;
14        this.letra = letra;
15    }
16
17    public String getNumero() {
18        return numero;
19    }
20
21    public void setNumero(String numero) {
22        this.numero = numero;
23    }
24
25    public char getLetra() {
26        return letra;
27    }
28
29    public void setLetra(char letra) {
30        this.letra = letra;
31    }
32 }
```

Sería más correcto tenerlo como en el proyecto 2, llamándolo **DNI.java**. Si nosotros leemos Numero.java no tendríamos claro de manera directa, cuál va a ser la finalidad del objeto. Por el contrario, si leemos DNI.java, nos hacemos una idea completa fácilmente.



Usa nombres fáciles de pronunciar

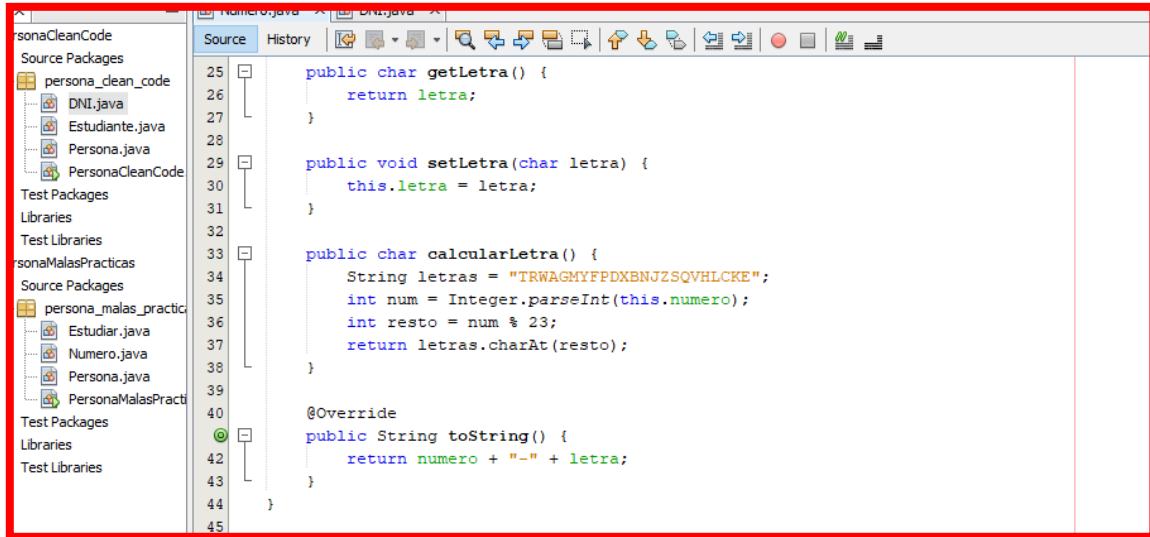
En la primera línea dentro del main del proyecto 1, hemos creado una variable que es la siguiente ***asdfjiasdfkljahsdf***, esta variable es errónea por varias cosas. Por una parte, como en el caso anterior, su nombre no nos aporta ningún tipo de información útil, pero, además es un nombre imposible de pronunciar. Lo correcto sería como se hace en el proyecto 2, que en esa misma línea se declara la variable como ***dni1***.

```
7  * @version 25/11/2023
8  */
9  public class PersonaMalasPracticas {
10
11     public static void main(String[] args) {
12         Numero asdfkljasdfkljahsdf = new Numero("12345678", 'A');
13         Numero dni2 = new Numero();
14         dni2.setNumero("06421478");
15         dni2.setLetra(dni2.calcularLetra());
16
17         Persona personal = new Persona("Helenca", "Rábano", "Fernández", 41, 'F', asdfkljasdfkljahsdf, "España");
18         Persona persona2 = new Persona("Tania", "García", "Martínez", 28, 'F', dni2, "EEUU");
19
20         nombre(personal);
21
22         int suma = sumar(1,2);
23         System.out.println(suma);
24         suma = agregar(2,1);
25         System.out.println(suma);
26
27
28         if (personal.getEdad() <= 5 || personal.getEdad() == 10 || personal.getEdad() == 17 || personal.getEdad() == 20 || personal.getEdad() == 25 || personal.getEdad() == 30 || personal.getEdad() == 35 || personal.getEdad() == 40 || personal.getEdad() == 45 || personal.getEdad() == 50 || personal.getEdad() == 55 || personal.getEdad() == 60 || personal.getEdad() == 65 || personal.getEdad() == 70 || personal.getEdad() == 75 || personal.getEdad() == 80 || personal.getEdad() == 85 || personal.getEdad() == 90 || personal.getEdad() == 95 || personal.getEdad() == 100) {
29             System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
30         }
31
32         Estudiar est = new Estudiar("aerdsD4", 1);
33         personal.setEstudiar(est);
34
35         boolean personaEsAlumno = false;
36         if (personal.getEstudiar() != null) {
37             personaEsAlumno = true;
```

```
5
6  * @author Helen
7  * @version 25/11/2023
8  */
9  public class PersonaCleanCode {
10
11     public static void main(String[] args) {
12         DNI dni1 = new DNI("12345678", 'A');
13         DNI dni2 = new DNI();
14         dni2.setNumero("06421478");
15         dni2.setLetra(dni2.calcularLetra());
16
17         Persona personal = new Persona("Helenca", "Rábano", "Fernández", 41, 'F', dni1, "España");
18         Persona persona2 = new Persona("Tania", "García", "Martínez", 28, 'F', dni2, "EEUU");
19
20         imprimir(personal);
21
22         double suma = sumar(1,2);
23         System.out.println(suma);
24         suma = sumar(2,1);
25         System.out.println(suma);
26
27         /*En este condicional valoramos si se cumplen las condiciones necesarias para
28         recibir un regalo de una promoción*/
29         if (personal.getEdad() <= 5 || personal.getEdad() == 10 || personal.getEdad() == 17 ||
30             personal.getEdad() == 20 || personal.getEdad() == 30 || personal.getEdad() == 40) {
31             System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
32         }
33
34         Estudiante estudiante = new Estudiante("aerdsD4", 1);
35         personal.setEstudiante(estudiante);
36     }
```

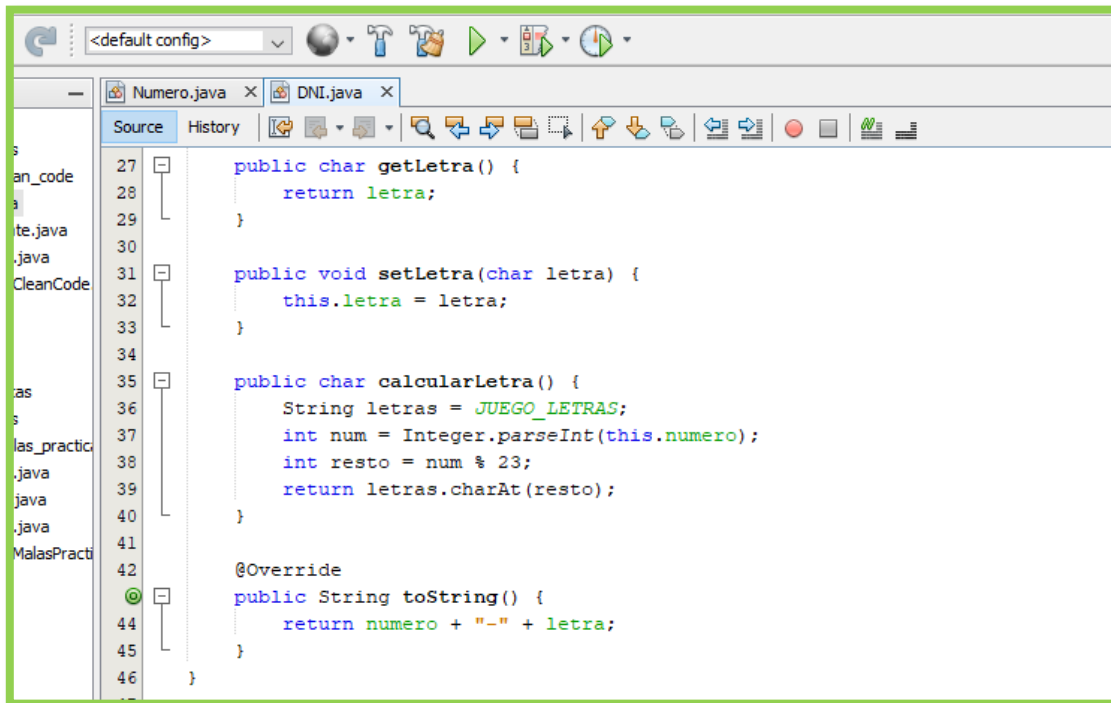
Usa nombres que puedan buscarse

En el proyecto 1, el método **calcularLetra()** de Numero.java podemos ver como directamente tenemos un String con la cadena de letras necesarias para realizar el cálculo de letra de un determinado DNI. Esta cadena tiene un valor constante y de hecho bastante ininteligible.



```
25 public char getLetra() {
26     return letra;
27 }
28
29 public void setLetra(char letra) {
30     this.lettra = letra;
31 }
32
33 public char calcularLetra() {
34     String letras = "TRWAGMYFPDXBNJZSQVHLCKE";
35     int num = Integer.parseInt(this.numero);
36     int resto = num % 23;
37     return letras.charAt(resto);
38 }
39
40 @Override
41 public String toString() {
42     return numero + "-" + letra;
43 }
44
45 }
```

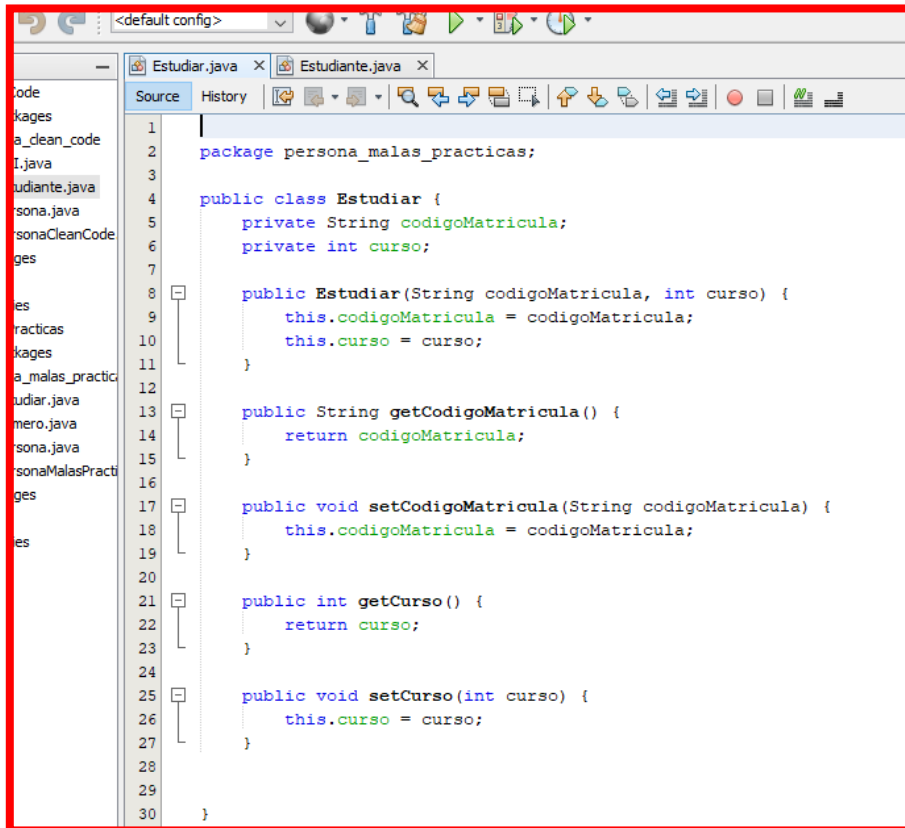
En el proyecto 2, el método **calcularLetra()** de DNI.java, vemos que llama a una constante denominada **"JUEGO_LETRAS"** lo cual es mucho más sencillo de buscar y comprender.



```
27 public char getLetra() {
28     return letra;
29 }
30
31 public void setLetra(char letra) {
32     this.lettra = letra;
33 }
34
35 public char calcularLetra() {
36     String letras = JUEGO_LETRAS;
37     int num = Integer.parseInt(this.numero);
38     int resto = num % 23;
39     return letras.charAt(resto);
40 }
41
42 @Override
43 public String toString() {
44     return numero + "-" + letra;
45 }
46
47 }
```

Nombres de clases y métodos (o funciones)

En proyecto 1 tenemos un objeto llamado **Estudiar.java**, que es incorrecto, porque debería ser un nombre. Como lo que realmente queremos representar es un estudiante, lo lógico es tenerlo como en el proyecto 2 **Estudiante.java**.



```
1 package persona_malas_practicas;
2
3
4 public class Estudiar {
5     private String codigoMatricula;
6     private int curso;
7
8     public Estudiar(String codigoMatricula, int curso) {
9         this.codigoMatricula = codigoMatricula;
10        this.curso = curso;
11    }
12
13    public String getCodigoMatricula() {
14        return codigoMatricula;
15    }
16
17    public void setCodigoMatricula(String codigoMatricula) {
18        this.codigoMatricula = codigoMatricula;
19    }
20
21    public int getCurso() {
22        return curso;
23    }
24
25    public void setCurso(int curso) {
26        this.curso = curso;
27    }
28
29
30 }
```

```

1  package persona_clean_code;
2
3
4  public class Estudiante {
5      private String codigoMatricula;
6      private int curso;
7
8      public Estudiante(String codigoMatricula, int curso) {
9          this.codigoMatricula = codigoMatricula;
10         this.curso = curso;
11     }
12
13     public String getCodigoMatricula() {
14         return codigoMatricula;
15     }
16
17     public void setCodigoMatricula(String codigoMatricula) {
18         this.codigoMatricula = codigoMatricula;
19     }
20
21     public int getCurso() {
22         return curso;
23     }
24
25     public void setCurso(int curso) {
26         this.curso = curso;
27     }
28
29

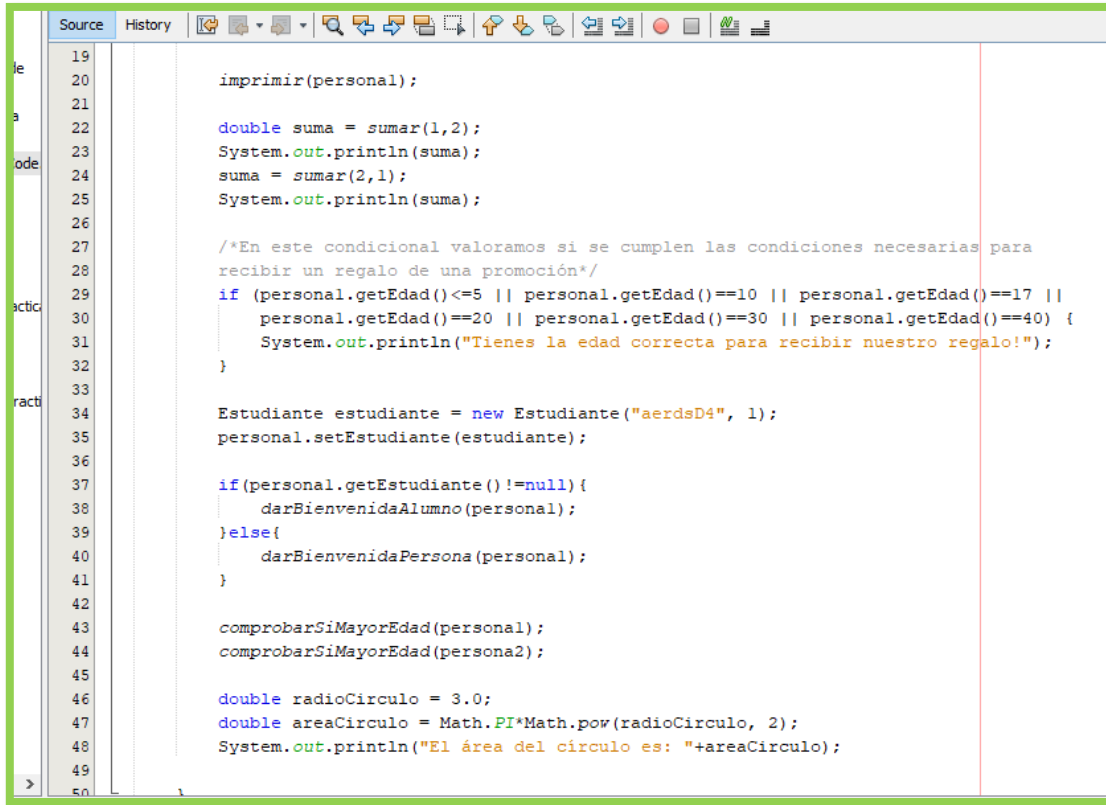
```

Por otro lado, tenemos en proyecto 1 una función que hemos denominado **nombre**, y recibiendo por parámetro un objeto de tipo persona, devuelve su nombre por consola. En el proyecto 2 tenemos la misma funcionalidad, pero lo hemos llamado **imprimir**, dado que es una función, que realiza una acción y con esto queda claro.

```

19  nombre(personal);
20
21
22  int suma = sumar(1,2);
23  System.out.println(suma);
24  suma = agregar(2,1);
25  System.out.println(suma);
26
27
28
29  if (personal.getEdad() <= 5 || personal.getEdad() == 10 || personal.getEdad() == 17 || personal.getEdad() == 20 || personal.getEdad() == 25) {
30      System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
31  }
32
33  Estudiar est = new Estudiar("aerdsD4", 1);
34  personal.setEstudiar(est);
35
36  boolean personaEsAlumno = false;
37  if (personal.getEstudiar() != null) {
38      personaEsAlumno = true;
39  }
40
41  darBienvenida(personal, personaEsAlumno);
42
43  //Este if comprueba si la edad de la persona es >= 18 y entonces dice que es mayor de edad, si no entonces dice que es menor de edad.
44  if (personal.getEdad() >= 18) {
45      System.out.println(personal.getNombre() + " " + personal.getApellido1() + " " + personal.getApellido2() + " es mayor de edad.");
46  } else {
47      System.out.println(personal.getNombre() + " " + personal.getApellido1() + " " + personal.getApellido2() + " es menor de edad.");
48  }
49  if (persona2.getEdad() >= 18) {

```



```
19
20     imprimir(personal);
21
22     double suma = sumar(1,2);
23     System.out.println(suma);
24     suma = sumar(2,1);
25     System.out.println(suma);
26
27     /*En este condicional valoramos si se cumplen las condiciones necesarias para
28     recibir un regalo de una promoción*/
29     if (personal.getEdad()<=5 || personal.getEdad()==10 || personal.getEdad()==17 ||
30         personal.getEdad()==20 || personal.getEdad()==30 || personal.getEdad()==40) {
31         System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
32     }
33
34     Estudiante estudiante = new Estudiante("aerdsD4", 1);
35     personal.setEstudiante(estudiante);
36
37     if(personal.getEstudiante()!=null){
38         darBienvenidaAlumno(personal);
39     }else{
40         darBienvenidaPersona(personal);
41     }
42
43     comprobarSiMayorEdad(personal);
44     comprobarSiMayorEdad(persona2);
45
46     double radioCirculo = 3.0;
47     double areaCirculo = Math.PI*Math.pow(radioCirculo, 2);
48     System.out.println("El área del círculo es: "+areaCirculo);
49
50
```

Elige una sola palabra por concepto

En proyecto 1 tenemos dos funciones que se llaman **sumar** y **agregar** respectivamente. Internamente hacen lo mismo (cambiando simplemente el orden de los números). Lo correcto es darse cuenta de ello y dejar la funcionalidad en una sola, como está en el proyecto 2 → **sumar**

(se ven en las capturas anteriores)

Bloque 2: Funciones

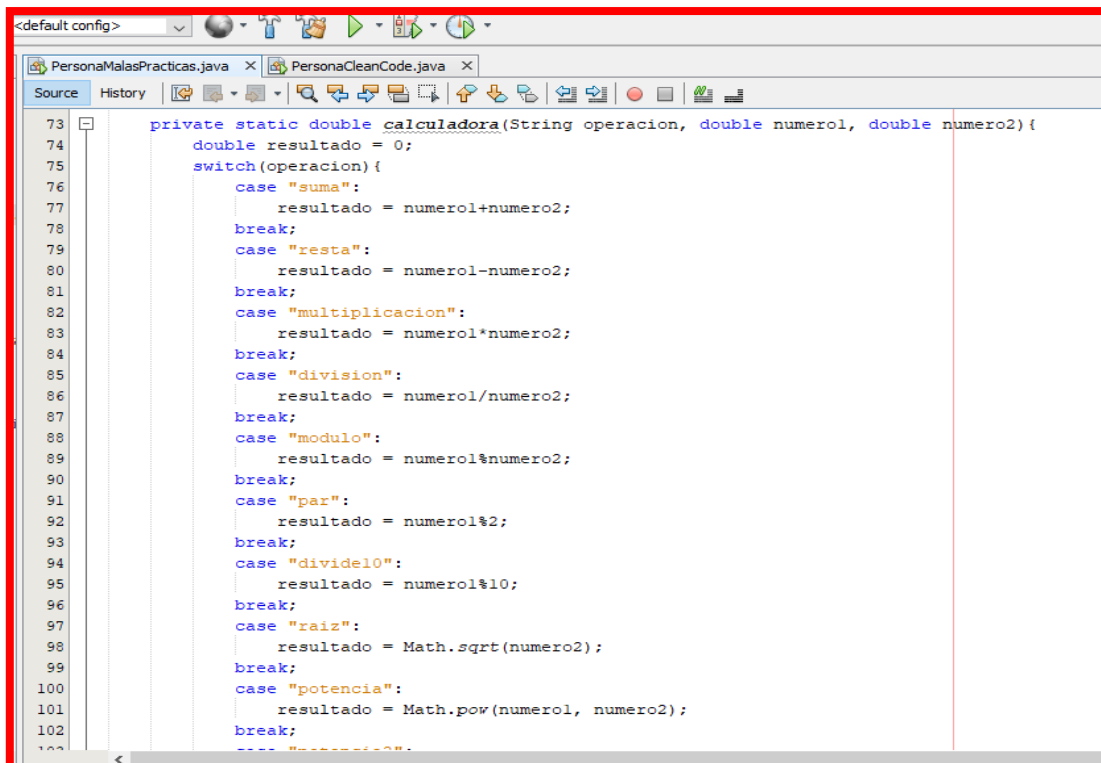
Las funciones deben ser pequeñas

Haz una única cosa

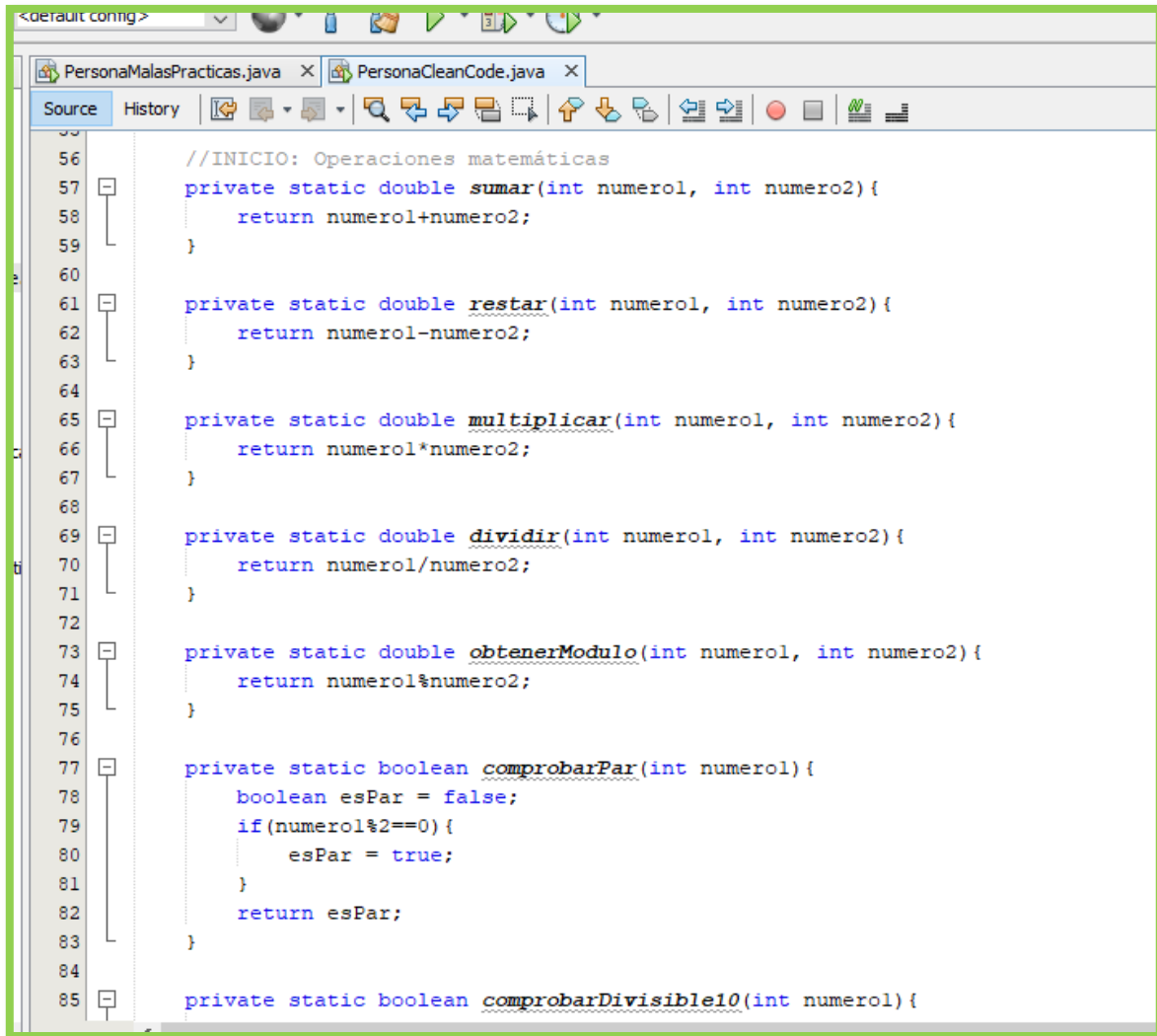
No abuses de los switch/when

En proyecto 1 tenemos una función bastante larga llamada **calculadora**, que podría ser mucho más aún si queremos implementar una calculadora científica. La solución es tenerlo como en el proyecto 2, en el que se ha dividido por funcionalidades más pequeñas. Con esto tenemos cada operación separada, haciendo las funciones mucho más cortas y además conseguimos que cada una de ellas sólo haga una única cosa y eliminamos un switch demasiado largo.

Para comprobar el largo de las líneas, en el main del proyecto 1 hay un if que es demasiado ancho y en el proyecto 2 el mismo if, como debería verse de manera correcta.



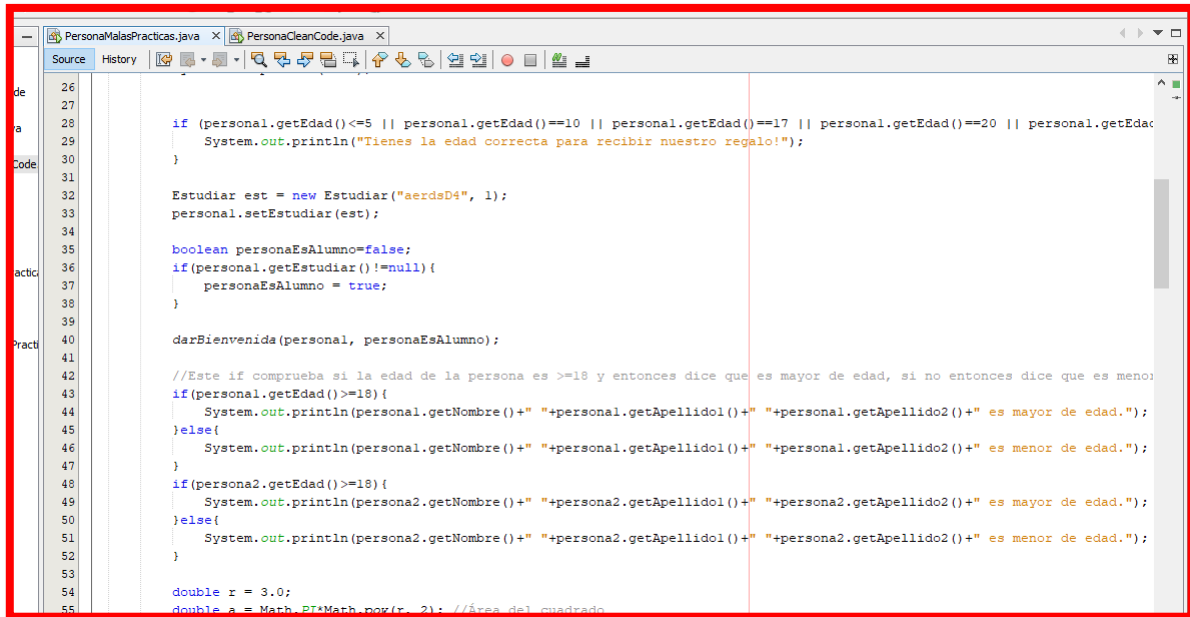
```
73 private static double calculadora(String operacion, double numero1, double numero2){
74     double resultado = 0;
75     switch(operacion){
76         case "suma":
77             resultado = numero1+numero2;
78             break;
79         case "resta":
80             resultado = numero1-numero2;
81             break;
82         case "multiplicacion":
83             resultado = numero1*numero2;
84             break;
85         case "division":
86             resultado = numero1/numero2;
87             break;
88         case "modulo":
89             resultado = numero1%numero2;
90             break;
91         case "par":
92             resultado = numero1%2;
93             break;
94         case "dividelo":
95             resultado = numero1%10;
96             break;
97         case "raiz":
98             resultado = Math.sqrt(numero2);
99             break;
100        case "potencia":
101            resultado = Math.pow(numero1, numero2);
102            break;
103        case "potencia2":
```

The screenshot shows an IDE window with two tabs: 'PersonaMalasPracticas.java' and 'PersonaCleanCode.java'. The 'PersonaCleanCode.java' tab is active, displaying a Java class with several static methods for mathematical operations. The code is as follows:

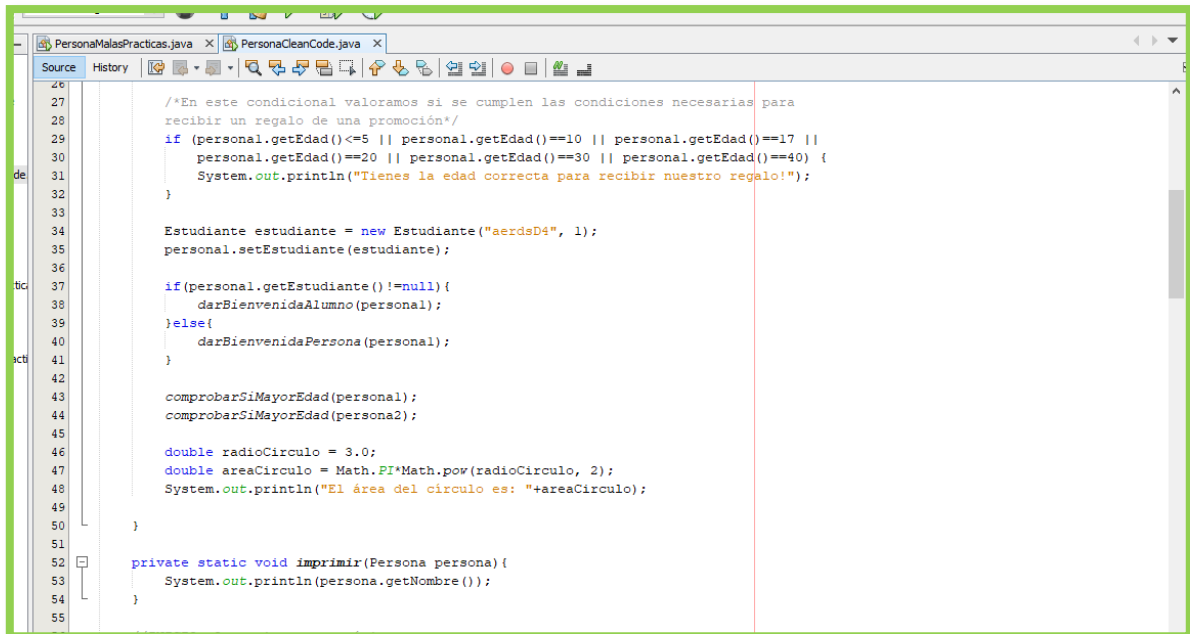
```
//INICIO: Operaciones matemáticas
56 private static double sumar(int numerol, int numero2){
57     return numerol+numero2;
58 }
59
60
61 private static double restar(int numerol, int numero2){
62     return numerol-numero2;
63 }
64
65 private static double multiplicar(int numerol, int numero2){
66     return numerol*numero2;
67 }
68
69 private static double dividir(int numerol, int numero2){
70     return numerol/numero2;
71 }
72
73 private static double obtenerModulo(int numerol, int numero2){
74     return numerol%numero2;
75 }
76
77 private static boolean comprobarPar(int numerol){
78     boolean esPar = false;
79     if(numerol%2==0){
80         esPar = true;
81     }
82     return esPar;
83 }
84
85 private static boolean comprobarDivisible10(int numerol){
```

Mostramos el "If" de ambos proyectos:



This screenshot shows the 'PersonaCleanCode.java' file in an IDE. The code is organized into a single 'if' block that checks multiple age conditions. The conditions are: `personal.getEdad() <= 5`, `personal.getEdad() == 10`, `personal.getEdad() == 17`, and `personal.getEdad() == 20`. If any of these conditions are met, it prints a message: `System.out.println("Tienes la edad correcta para recibir nuestro regalo!");`. Following this, it creates an `Estudiante` object, sets it on the `personal` object, and checks if the `personal` object has an `Estudiante` set. If so, it calls `darBienvenidaAlumno`; otherwise, it calls `darBienvenidaPersona`. It then checks the age of `persona2` and prints a message indicating if they are older or younger than 18. Finally, it calculates the area of a circle with a radius of 3.0.

```
26
27
28     if (personal.getEdad() <= 5 || personal.getEdad() == 10 || personal.getEdad() == 17 || personal.getEdad() == 20 || personal.getEdad() == 30 || personal.getEdad() == 40) {
29         System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
30     }
31
32     Estudiante estudiante = new Estudiante("aerdsD4", 1);
33     personal.setEstudiante(estudiante);
34
35     boolean personaEsAlumno = false;
36     if (personal.getEstudiante() != null) {
37         personaEsAlumno = true;
38     }
39
40     darBienvenida(personal, personaEsAlumno);
41
42     //Este if comprueba si la edad de la persona es >= 18 y entonces dice que es mayor de edad, si no entonces dice que es menor de edad.
43     if (personal.getEdad() >= 18) {
44         System.out.println(personal.getNombre() + " " + personal.getApellido1() + " " + personal.getApellido2() + " es mayor de edad.");
45     } else {
46         System.out.println(personal.getNombre() + " " + personal.getApellido1() + " " + personal.getApellido2() + " es menor de edad.");
47     }
48
49     if (persona2.getEdad() >= 18) {
50         System.out.println(persona2.getNombre() + " " + persona2.getApellido1() + " " + persona2.getApellido2() + " es mayor de edad.");
51     } else {
52         System.out.println(persona2.getNombre() + " " + persona2.getApellido1() + " " + persona2.getApellido2() + " es menor de edad.");
53     }
54
55     double r = 3.0;
56     double a = Math.PI * Math.pow(r, 2); //Área del cuadrado
```



This screenshot shows the 'PersonaMalasPracticas.java' file in an IDE. The code is more verbose than the clean version. It has a comment: `/*En este condicional valoramos si se cumplen las condiciones necesarias para recibir un regalo de una promoción*/`. The 'if' statement checks the same age conditions as the clean version. It then creates an `Estudiante` object and sets it. Next, it checks if the `personal` object has an `Estudiante` set. If so, it calls `darBienvenidaAlumno`; otherwise, it calls `darBienvenidaPersona`. It then checks the age of `persona2` and prints a message indicating if they are older or younger than 18. Finally, it calculates the area of a circle with a radius of 3.0. A private static method `imprimir` is also defined at the bottom.

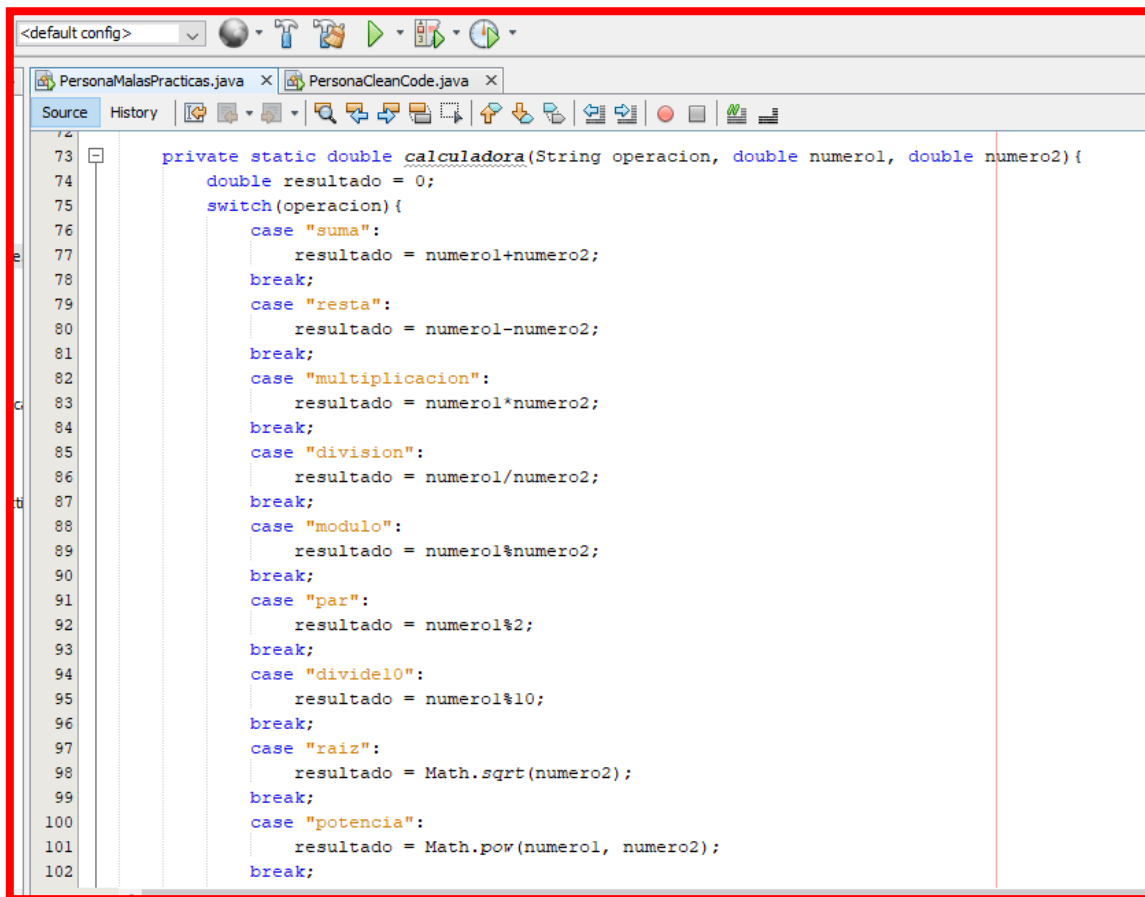
```
26
27     /*En este condicional valoramos si se cumplen las condiciones necesarias para
28     recibir un regalo de una promoción*/
29     if (personal.getEdad() <= 5 || personal.getEdad() == 10 || personal.getEdad() == 17 ||
30         personal.getEdad() == 20 || personal.getEdad() == 30 || personal.getEdad() == 40) {
31         System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
32     }
33
34     Estudiante estudiante = new Estudiante("aerdsD4", 1);
35     personal.setEstudiante(estudiante);
36
37     if (personal.getEstudiante() != null) {
38         darBienvenidaAlumno(personal);
39     } else {
40         darBienvenidaPersona(personal);
41     }
42
43     comprobarSiMayorEdad(personal);
44     comprobarSiMayorEdad(persona2);
45
46     double radioCirculo = 3.0;
47     double areaCirculo = Math.PI * Math.pow(radioCirculo, 2);
48     System.out.println("El área del círculo es: " + areaCirculo);
49
50 }
51
52 private static void imprimir(Persona persona) {
53     System.out.println(persona.getNombre());
54 }
55
```

¿Cuántos argumentos debe tener una función?

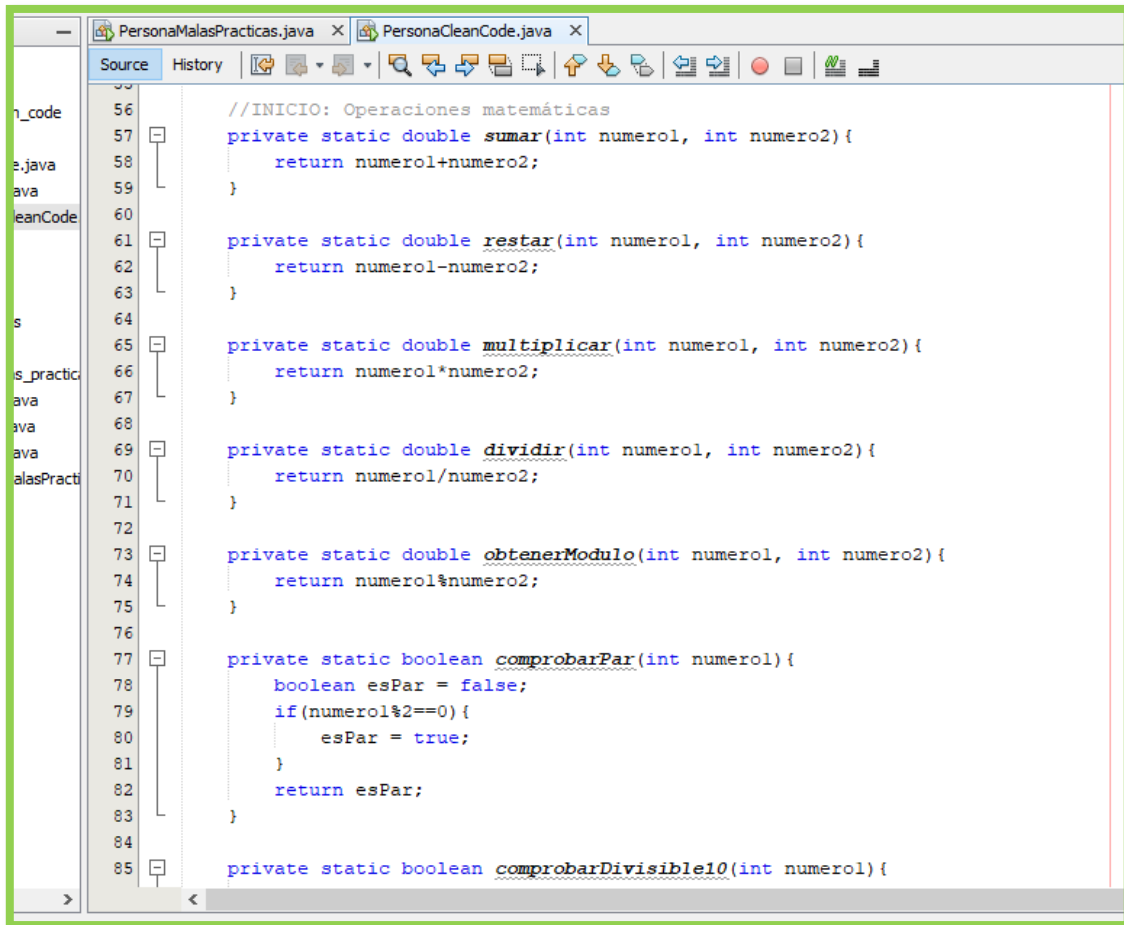
Usando de nuevo como ejemplo la función de **calculadora** en el proyecto1, se observa que, aunque no parezcan muchos, en este caso 3 ya está entorpeciendo bastante el entendimiento de cómo funcionará por dentro y será difícil de entender incluso a la hora de probarlo.

Por una parte, tenemos un primer parámetro en que se pasa como String el nombre de la operación a realizar, esto es bastante peligroso, porque si la cadena no coincide exactamente en la llamada, no realizará la operación que queremos ("suma" no es lo mismo que "sumar" o que "SUMA"). Más allá de lo lioso que pueda ser esto, también tiene sentido que el nombre de la función sea quien determine lo que se va a hacer en cada momento, como se puede ver en las funciones equivalentes del proyecto 2.

Por otro lado, hay operaciones que sólo requieren un operando y otras requieren dos, con lo cual esperar dos números de entrada para cualquier caso genérico, no es lo mejor. Lo correcto es lo que se hace en el proyecto 2, donde cada función recibe únicamente lo que necesita.



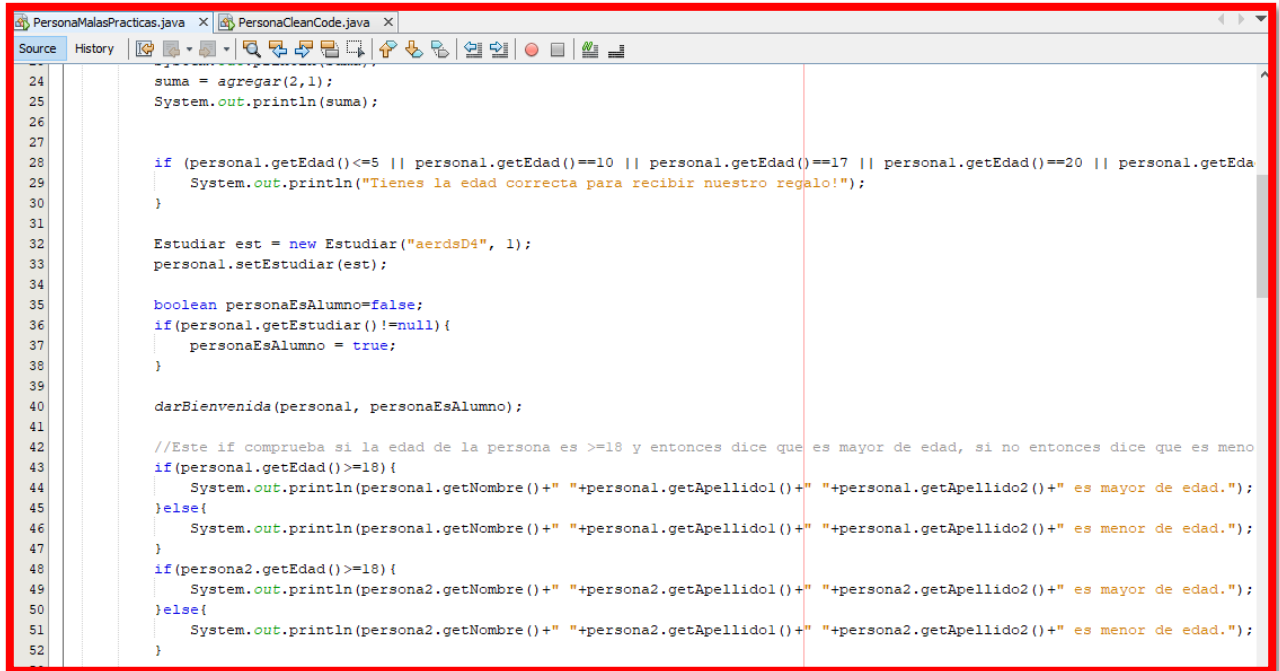
```
73 private static double calculadora(String operacion, double numero1, double numero2){
74     double resultado = 0;
75     switch(operacion){
76         case "suma":
77             resultado = numero1+numero2;
78             break;
79         case "resta":
80             resultado = numero1-numero2;
81             break;
82         case "multiplicacion":
83             resultado = numero1*numero2;
84             break;
85         case "division":
86             resultado = numero1/numero2;
87             break;
88         case "modulo":
89             resultado = numero1%numero2;
90             break;
91         case "par":
92             resultado = numero1%2;
93             break;
94         case "divide10":
95             resultado = numero1%10;
96             break;
97         case "raiz":
98             resultado = Math.sqrt(numero2);
99             break;
100        case "potencia":
101            resultado = Math.pow(numero1, numero2);
102            break;
```



```
56 //INICIO: Operaciones matemáticas
57 private static double sumar(int numero1, int numero2){
58     return numero1+numero2;
59 }
60
61 private static double restar(int numero1, int numero2){
62     return numero1-numero2;
63 }
64
65 private static double multiplicar(int numero1, int numero2){
66     return numero1*numero2;
67 }
68
69 private static double dividir(int numero1, int numero2){
70     return numero1/numero2;
71 }
72
73 private static double obtenerModulo(int numero1, int numero2){
74     return numero1%numero2;
75 }
76
77 private static boolean comprobarPar(int numero1){
78     boolean esPar = false;
79     if(numero1%2==0){
80         esPar = true;
81     }
82     return esPar;
83 }
84
85 private static boolean comprobarDivisible10(int numero1){
```

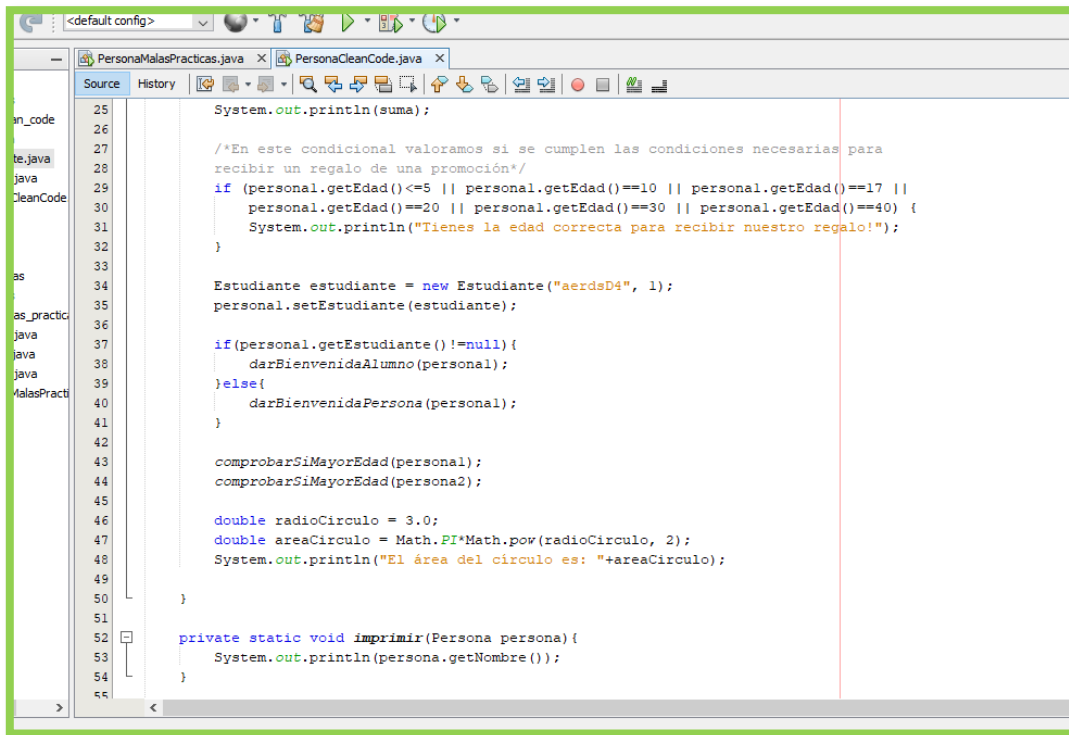
Evita los flag arguments

En proyecto 1 hay una función que es **darBienvenida**, le pasamos como parámetros una Persona y un booleano que nos dice si es alumno o no y dependiendo de este flag, le mostramos un mensaje u otro.



```
24 suma = agregar(2,1);
25 System.out.println(suma);
26
27
28 if (personal.getEdad()<=5 || personal.getEdad()==10 || personal.getEdad()==17 || personal.getEdad()==20 || personal.getEdad()==25)
29     System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
30 }
31
32 Estudiar est = new Estudiar("aerdsD4", 1);
33 personal.setEstudiar(est);
34
35 boolean personaEsAlumno=false;
36 if(personal.getEstudiar()!=null){
37     personaEsAlumno = true;
38 }
39
40 darBienvenida(personal, personaEsAlumno);
41
42 //Este if comprueba si la edad de la persona es >=18 y entonces dice que es mayor de edad, si no entonces dice que es menor de edad
43 if(personal.getEdad()>=18){
44     System.out.println(personal.getNombre()+" "+personal.getApellido1()+" "+personal.getApellido2()+" es mayor de edad.");
45 }else{
46     System.out.println(personal.getNombre()+" "+personal.getApellido1()+" "+personal.getApellido2()+" es menor de edad.");
47 }
48
49 if(persona2.getEdad()>=18){
50     System.out.println(persona2.getNombre()+" "+persona2.getApellido1()+" "+persona2.getApellido2()+" es mayor de edad.");
51 }else{
52     System.out.println(persona2.getNombre()+" "+persona2.getApellido1()+" "+persona2.getApellido2()+" es menor de edad.");
53 }
```

En el proyecto 2 se muestra como habría que hacerlo, como queremos diferenciar el saludo o mensaje, si la persona es o no alumno, creamos dos funciones, una que será **darBienvenidaAlumno** y otra **darBienvenidaPersona** y el control de si es una cosa u otra lo dejamos fuera de la lógica.



```
25      System.out.println(suma);
26
27      /*En este condicional valoramos si se cumplen las condiciones necesarias para
28      recibir un regalo de una promoción*/
29      if (personal.getEdad() <= 5 || personal.getEdad() == 10 || personal.getEdad() == 17 ||
30          personal.getEdad() == 20 || personal.getEdad() == 30 || personal.getEdad() == 40) {
31          System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
32      }
33
34      Estudiante estudiante = new Estudiante("aerdsD4", 1);
35      personal.setEstudiante(estudiante);
36
37      if (personal.getEstudiante() != null) {
38          darBienvenidaAlumno(personal);
39      } else {
40          darBienvenidaPersona(personal);
41      }
42
43      comprobarSiMayorEdad(personal);
44      comprobarSiMayorEdad(persona2);
45
46      double radioCirculo = 3.0;
47      double areaCirculo = Math.PI * Math.pow(radioCirculo, 2);
48      System.out.println("El área del círculo es: " + areaCirculo);
49
50  }
51
52  private static void imprimir(Persona persona) {
53      System.out.println(persona.getNombre());
54  }
55  }
```

No te repitas

En el proyecto 1 tenemos una lógica que sirve para saber mostrar por pantalla si una persona es mayor o menor de edad, en el caso de la mala práctica, estamos repitiendo dos veces el código (una vez por cada persona). En el proyecto 2, hemos encapsulado la lógica en una función denominada **comprobarSiMayorEdad** de esta forma, la lógica de comprobación y de imprimir el mensaje, sólo aparece una vez en nuestro código y le llamamos tantas veces como sea necesario.

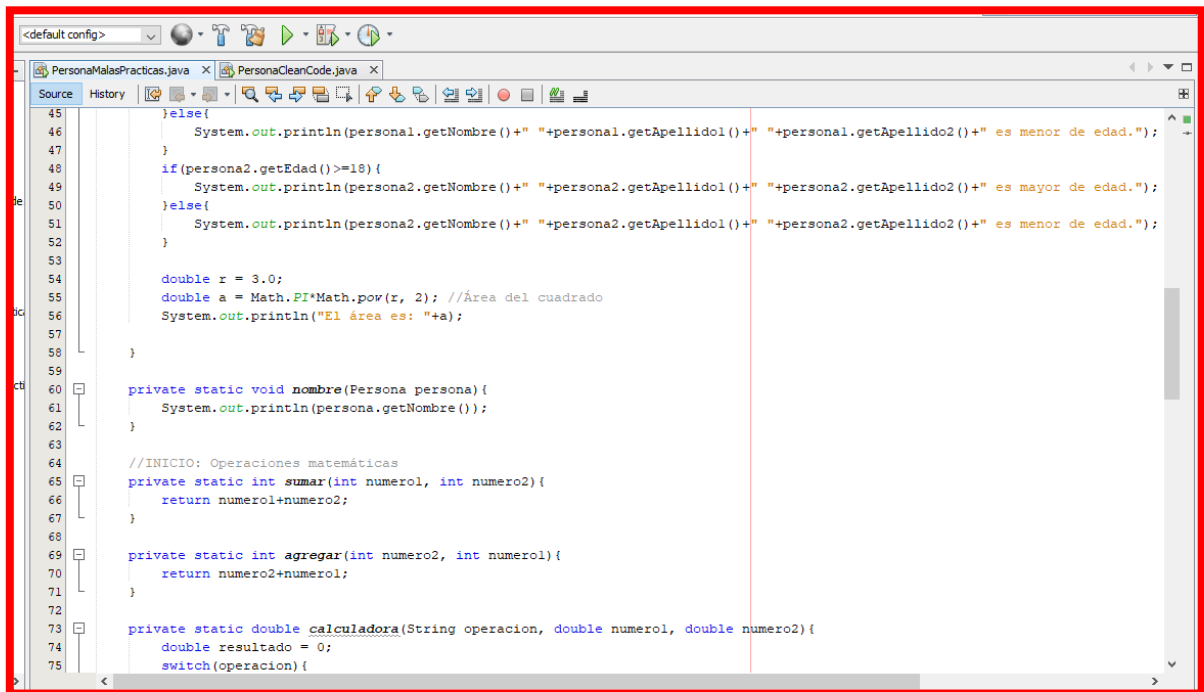
(aparece en las capturas anteriores)

Bloque 3: Comentarios

Los comentarios mienten

Usa código autoexplicativo

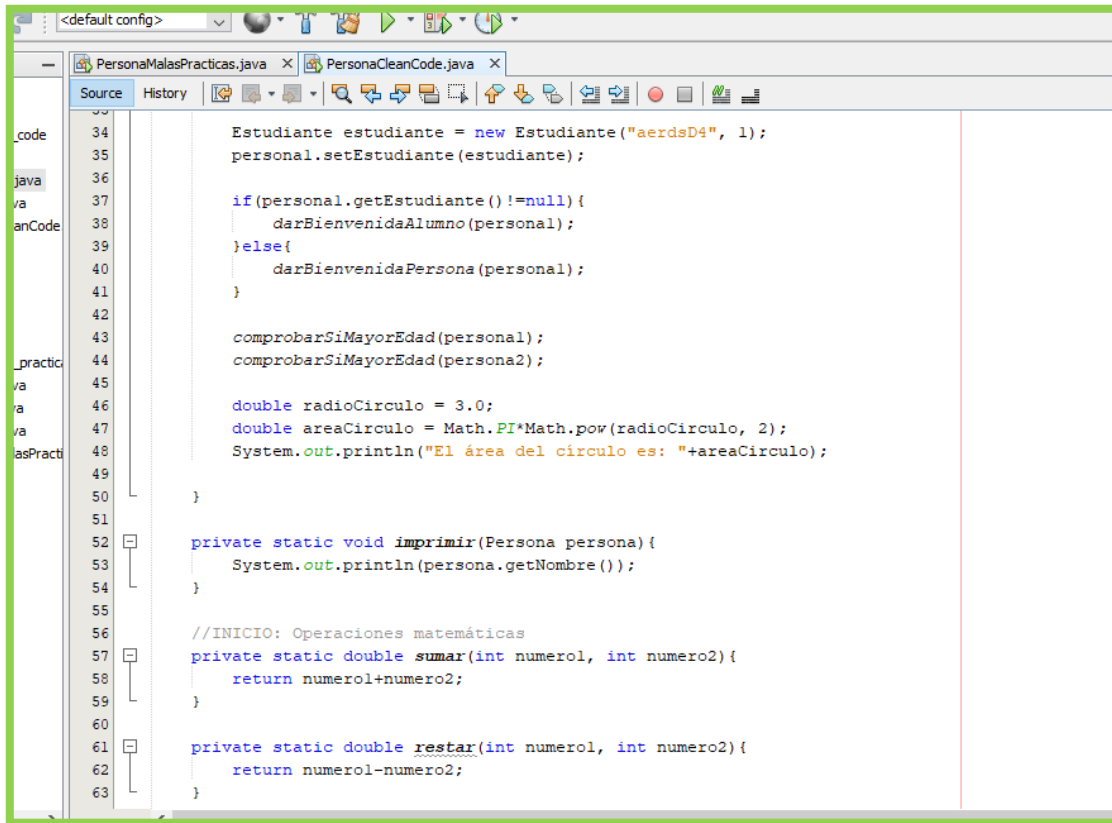
Entre las líneas 54-56 del main del proyecto 1, podemos ver un comentario erróneo que pone: **//Área del cuadrado**. Realmente lo que estamos calculando es el área del círculo, por lo que el comentario está llevándonos a error (mente).



The screenshot shows an IDE window with two tabs: 'PersonaMalasPracticas.java' and 'PersonaCleanCode.java'. The 'PersonaCleanCode.java' tab is active, displaying Java code. The code includes several methods and a main method. In the main method, around lines 54-56, there is a calculation for the area of a circle using the formula $a = \text{Math.PI} * \text{Math.pow}(r, 2)$. The comment for this line is `//Área del cuadrado`, which is incorrect as it refers to a square instead of a circle. The code also includes methods for checking age, summing numbers, and a calculator function.

```
45 }else{
46     System.out.println(personal.getNombre()+" "+personal.getApellido1()+" "+personal.getApellido2()+" es menor de edad.");
47 }
48 if(persona2.getEdad()>=18){
49     System.out.println(persona2.getNombre()+" "+persona2.getApellido1()+" "+persona2.getApellido2()+" es mayor de edad.");
50 }else{
51     System.out.println(persona2.getNombre()+" "+persona2.getApellido1()+" "+persona2.getApellido2()+" es menor de edad.");
52 }
53
54 double r = 3.0;
55 double a = Math.PI*Math.pow(r, 2); //Área del cuadrado
56 System.out.println("El área es: "+a);
57
58 }
59
60 private static void nombre(Persona persona){
61     System.out.println(persona.getNombre());
62 }
63
64 //INICIO: Operaciones matemáticas
65 private static int sumar(int numerol, int numero2){
66     return numerol+numero2;
67 }
68
69 private static int agregar(int numero2, int numerol){
70     return numero2+numerol;
71 }
72
73 private static double calculadora(String operacion, double numerol, double numero2){
74     double resultado = 0;
75     switch(operacion){
```

Además, como la fórmula tal vez podría ser más descriptiva y no lo es, se puede dar la posible confusión de dar por bueno el comentario. Entre la línea 46-48 del main del proyecto 2, vemos que las variables dejan suficientemente claro el área de que figura estamos calculando y por ello, ni si quiera es necesario poner un comentario para explicarlo.



```
<default config>
PersonaMalasPracticas.java
PersonaCleanCode.java

Source History
34 Estudiante estudiante = new Estudiante("aerdsD4", 1);
35 personal.setEstudiante(estudiante);
36
37 if(personal.getEstudiante() != null){
38     darBienvenidaAlumno(personal);
39 }else{
40     darBienvenidaPersona(personal);
41 }
42
43 comprobarSiMayorEdad(personal);
44 comprobarSiMayorEdad(persona2);
45
46 double radioCirculo = 3.0;
47 double areaCirculo = Math.PI*Math.pow(radioCirculo, 2);
48 System.out.println("El área del círculo es: "+areaCirculo);
49
50 }
51
52 private static void imprimir(Persona persona){
53     System.out.println(persona.getNombre());
54 }
55
56 //INICIO: Operaciones matemáticas
57 private static double sumar(int numero1, int numero2){
58     return numero1+numero2;
59 }
60
61 private static double restar(int numero1, int numero2){
62     return numero1-numero2;
63 }
```

A veces los comentarios son necesarios

En el proyecto 2, en las líneas 27 y 28, se especifica para que se está utilizando una sentencia if. Sin el comentario, como está en el proyecto 1, puede parecer que los datos numéricos de la sentencia son algo arbitrarios, por el contrario, teniendo el comentario, sabemos perfectamente que se trata de las condiciones de edad que tiene que cumplir alguien para poder recibir un premio.

```
15      dni2.setLetra(dni2.calcularLetra());
16
17      Persona personal = new Persona("Helena", "Rábano", "Fernández", 41, 'F', asdfiljasdfkljahsdf, "España");
18      Persona persona2 = new Persona("Tania", "García", "Martínez", 28, 'F', dni2, "EEUU");
19
20      nombre(personal);
21
22      int suma = sumar(1,2);
23      System.out.println(suma);
24      suma = agregar(2,1);
25      System.out.println(suma);
26
27
28      if (personal.getEdad()<=5 || personal.getEdad()==10 || personal.getEdad()==17 || personal.getEdad()==20 || personal.getEdad()>=30)
29      {
30          System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
31      }
32
33      Estudiar est = new Estudiar("aerdsD4", 1);
34      personal.setEstudiar(est);
35
36      boolean personaEsAlumno=false;
37      if(personal.getEstudiar()!=null){
38          personaEsAlumno = true;
```

```
22      double suma = sumar(1,2);
23      System.out.println(suma);
24      suma = sumar(2,1);
25      System.out.println(suma);
26
27      /*En este condicional valoramos si se cumplen las condiciones necesarias para
28      recibir un regalo de una promoción*/
29      if (personal.getEdad()<=5 || personal.getEdad()==10 || personal.getEdad()==17 ||
30          personal.getEdad()==20 || personal.getEdad()==30 || personal.getEdad()==40) {
31          System.out.println("Tienes la edad correcta para recibir nuestro regalo!");
32      }
33
34      Estudiante estudiante = new Estudiante("aerdsD4", 1);
35      personal.setEstudiante(estudiante);
36
37      if(personal.getEstudiante()!=null){
38          darBienvenidaAlumno(personal);
39      }else{
40          darBienvenidaPersona(personal);
41      }
42
43      comprobarSiMayorEdad(personal);
```

Los comentarios dicen qué hace el código, no cómo lo hace

En el proyecto 1 en la línea 42, vemos un comentario que explica algo muy obvio sobre las sentencias de abajo y entra al detalle en cómo funciona la propia lógica, lo cual es incorrecto. Sin embargo en el proyecto 2 en la función **comprobarSiMayorEdad**, tenemos un comentario que es útil para entender, que la mayoría de edad es diferente en España y en EEUU. No entra al detalle de cómo funcionan los if-else del código, pero si aclara algo que no todo el mundo sepa de primeras.

```

37     personaEsAlumno = true;
38 }
39
40 darBienvenida(personal, personaEsAlumno);
41
42 //Este if comprueba si la edad de la persona es >=18 y entonces dice que es mayor de edad, si no entonces dice que es menor
43 if (personal.getEdad() >= 18) {
44     System.out.println(personal.getNombre() + " " + personal.getApellido1() + " " + personal.getApellido2() + " es mayor de edad.");
45 } else {
46     System.out.println(personal.getNombre() + " " + personal.getApellido1() + " " + personal.getApellido2() + " es menor de edad.");
47 }
48 if (persona2.getEdad() >= 18) {
49     System.out.println(persona2.getNombre() + " " + persona2.getApellido1() + " " + persona2.getApellido2() + " es mayor de edad.");
50 } else {
51     System.out.println(persona2.getNombre() + " " + persona2.getApellido1() + " " + persona2.getApellido2() + " es menor de edad.");
52 }
53
54 double r = 3.0;
55 double a = Math.PI * Math.pow(r, 2); //Área del cuadrado
56 System.out.println("El área es: " + a);
57
58 }
59
60 private static void nombre(Persona persona) {
61     System.out.println(persona.getNombre());
62 }

```

```

110     return Math.cos(numero1);
111 }
112
113 private static double calcularTangente(double numero1) {
114     return Math.tan(numero1);
115 }
116 //FINAL: Operaciones matemáticas
117
118 //INICIO: Mensajes
119 private static void darBienvenidaAlumno(Persona persona) {
120     System.out.println("Bienvenido alumno!: " + persona.getNombre());
121 }
122
123 private static void darBienvenidaPersona(Persona persona) {
124     System.out.println("Bienvenido profesor o invitado!: " + persona.getNombre());
125 }
126
127 private static void comprobarSiMayorEdad(Persona persona) {
128     //Dependiendo del país, la edad adulta comienza con diferentes años
129     String nombreCompleto = persona.getNombre() + " " + persona.getApellido1() + " " + persona.getApellido2();
130     if ((persona.getEdad() >= 18 && persona.getPais().equals("España")) || (persona.getEdad() >= 21 && persona.getPais().equals("EE
131     System.out.println(nombreCompleto + " es mayor de edad.");
132 } else {
133     System.out.println(nombreCompleto + " es menor de edad.");
134 }
135 }
136 //Final: Mensajes
137
138

```