

# CALCULADORA

---

Mauro Hernández Cardenal  
1ºDAW | PROGRAMACIÓN

## Informe

<https://github.com/Maurohc91/Calculadora>

En este informe vamos a pasar a detallar qué hemos hecho, cómo lo hemos hecho y el por qué.

Para ello, voy a usar capturas de pantalla del código y voy a ir explicándolo poco a poco. Después, adjuntare también capturas para ver el funcionamiento o no funcionamiento de dicho código.

En primer lugar y como casi siempre tenemos un fragmento de código que define un paquete llamado org.example y contiene una clase pública llamada Calculadora. Dentro de esta clase, el método main es el punto de entrada del programa, donde empezaremos con la ejecución del código.

```
1 package org.example;
2
3 import java.util.InputMismatchException;
4 import java.util.Scanner;
5
6
7
8
9 public class Calculadora {
10
11
12     public static void main(String[] args) {
13
14         Scanner entrada = new Scanner(System.in);
15
```

A continuación, tenemos la definición de las variables.

Los operandos en double para que saque los decimales, el signo como string y un booleano para más adelante poder controlar el try and catch.

```
16     double operando1 = 0;
17     double operando2 = 0;
18     String signo = "x";
19     boolean numerovalido = false;
```

Como dice el ejercicio, primero daremos la bienvenida.

```
21     System.out.println("BIENVENIDO A MI CALCULADORA.");
```

Aquí viene el primer try and catch dentro del while.

El while nos indica que el bucle seguirá ejecutándose mientras la variable numerovalido sea falsa.

Dentro del bucle, tenemos el try, y dentro saltará una frase para que metamos el primer operando.

Si el valor escrito es correcto, numerovalido pasará a ser true y seguiremos con el programa, pero si lo que introducimos no es válido, saltará al catch que nos mostrará un mensaje para que metamos bien el operando de nuevo.

```
22
23     while (!numerovalido) {
24         try {
25
26
27             System.out.println("Introduce el primer operando: ");
28             operando1 = entrada.nextDouble();
29             numerovalido = true;
30
31         } catch (InputMismatchException er) {
32             System.out.println("Introduce el primer operando bien.");
33             entrada.nextLine();
34         }
35     }
```

Aquí llega el segundo while y con él otro booleano para control que el signo sea válido.

Para ello he creado la variable booleana signovalido = false.

El while nos indica que el bucle seguirá ejecutándose mientras la variable signovalido sea falsa.

A continuación, y antes del if, el programa nos preguntará que metamos un signo para hacer la operación.

En el if, ponemos que si el signo no es igual a algún operando, que salte un mensaje poniendo que no es correcto y en caso de que el signo introducido sea correcto, pase directamente al else, que estará signovalido = true, para poder salir del bucle.

```
36     boolean signovalido = false;
37
38     while (!signovalido) {
39         System.out.println("Introduce el signo a aplicar (+,-,x,/,R");
40         signo = entrada.next();
41         signo = signo.toLowerCase();
42
43         if (!signo.equals("+") && !signo.equals("-") &&
44             !signo.equalsIgnoreCase("x") && !signo.equals("/") && !signo.equalsIgnoreCase("R")) {
45             System.out.println("no es correcto.");
46             entrada.nextLine();
47         } else {
48             signovalido = true;
49         }
50     }
51 }
```

Aquí tenemos algo importante en este if, y es que si el signo NO es una R (raíz cuadrada), el programa nos tendrá que preguntar por el segundo operando.

Y lo haremos de la siguiente manera.

Primero, creamos otro booleano `numerovalido2 = false`, para que, igual que con el primer operando, no nos puedan poner letras u otros caracteres.

El while nos indica que el bucle seguirá ejecutándose mientras la variable `numerovalido2` sea falsa.

Dentro del try, nos pide que metamos el segundo operando y si metemos un número válido, la condición será true y podremos seguir con nuestro código. En caso de que no sea válido, saltaría al catch, que imprimiría por pantalla que metamos el segundo operando bien.

```
52     if (!signo.equalsIgnoreCase( anotherString: "r")) {
53
54         boolean numerovalido2 = false;
55         while (!numerovalido2) {
56             try {
57
58                 System.out.println("Introduce el segundo operando: ");
59                 operando2 = entrada.nextDouble();
60                 numerovalido2 = true;
61
62             } catch (InputMismatchException er2) {
63                 System.out.println("Introduce el segundo operando bien.");
64                 entrada.nextLine();
65             }
66         }
67     }
```

Aquí empezamos con el switch, que en primer lugar igualaremos las mayúsculas y las minúsculas para la X y la R.

En los tres primeros casos, no hay ningún problema. Si es case + se hará una suma, si es case -, se hará una resta y si es case x, se hará una multiplicación.

Importante poner el break al final de cada case.

```
69         switch (signo.toLowerCase()) {
70             case "+":
71                 System.out.println(operando1 + operando2);
72                 break;
73
74             case "-":
75                 System.out.println(operando1 - operando2);
76                 break;
77
78             case "x":
79                 System.out.println(operando1 * operando2);
80                 break;
81         }
```

En la división es cuando nos puede surgir algún problema, por eso creamos un if, y ponemos que, si operando2 es igual a 0, tiene que saltar un mensaje con un error, ya que no se puede dividir entre 0, pero si no es 0, el programa imprimirá el resultado de operando1 entre operando2.

```
82         case "/":
83             if (operando2 == 0) {
84                 System.out.println("No se puede dividir entre 0.");
85             } else {
86                 System.out.println(operando1 / operando2);
87             }
```

En el caso de la raíz cuadrada, tenemos que asegurarnos que el primer operando introducido es mayor a 0 y eso lo conseguimos con el if y el else.

En primer lugar, si el operando es mayor que 0, que haga la raíz cuadrada del primer operando y, si el operando1 fuera 0 o negativo, saltaría un error.

```
89         case "r":
90             if (operando1 > 0) {
91
92                 System.out.println(Math.sqrt(operando1));
93
94             } else {
95                 System.out.println("Introduce un número correcto para este formato.");
96             }
97             break;
```

Para finalizar el código, tenemos el default, que es lo que pasaría si no se cumple ningún de los casos anteriores.

```
99         default:
100             System.out.println("Operación inválida");
101             break;
102     }
103 }
104 }
105 }
```

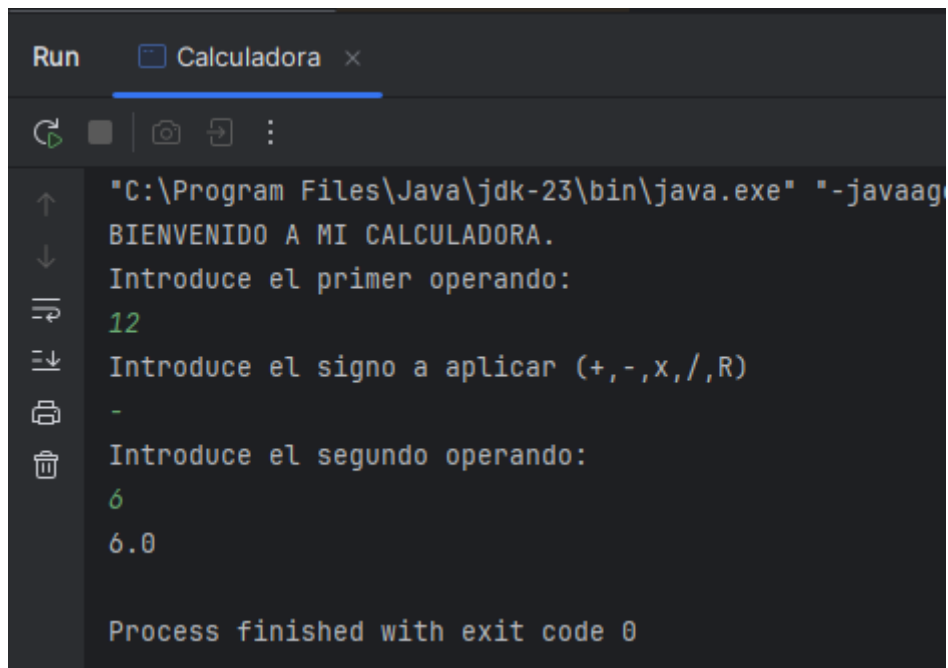
## Pruebas

Aquí voy a dejar las pruebas con las diferentes operaciones y los fallos que pueda dar.

La suma:

```
"C:\Program Files\Java\jdk-23\bin\java.exe"  
BIENVENIDO A MI CALCULADORA.  
Introduce el primer operando:  
4  
Introduce el signo a aplicar (+,-,x,/,R)  
+  
Introduce el segundo operando:  
6  
10.0  
  
Process finished with exit code 0
```

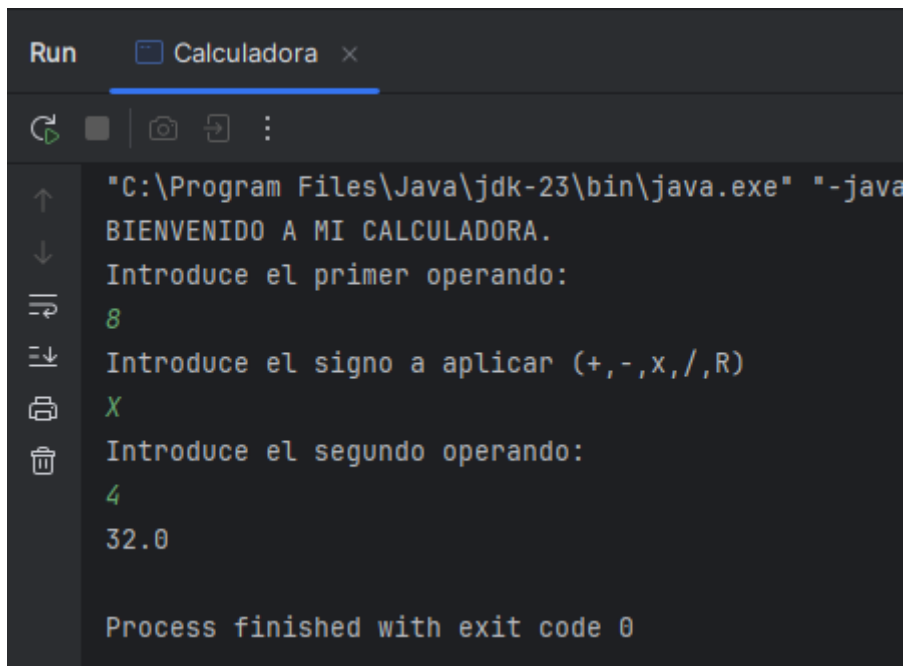
La resta:



The screenshot shows an IDE window titled "Run" with a tab for "Calculadora". The console output is as follows:

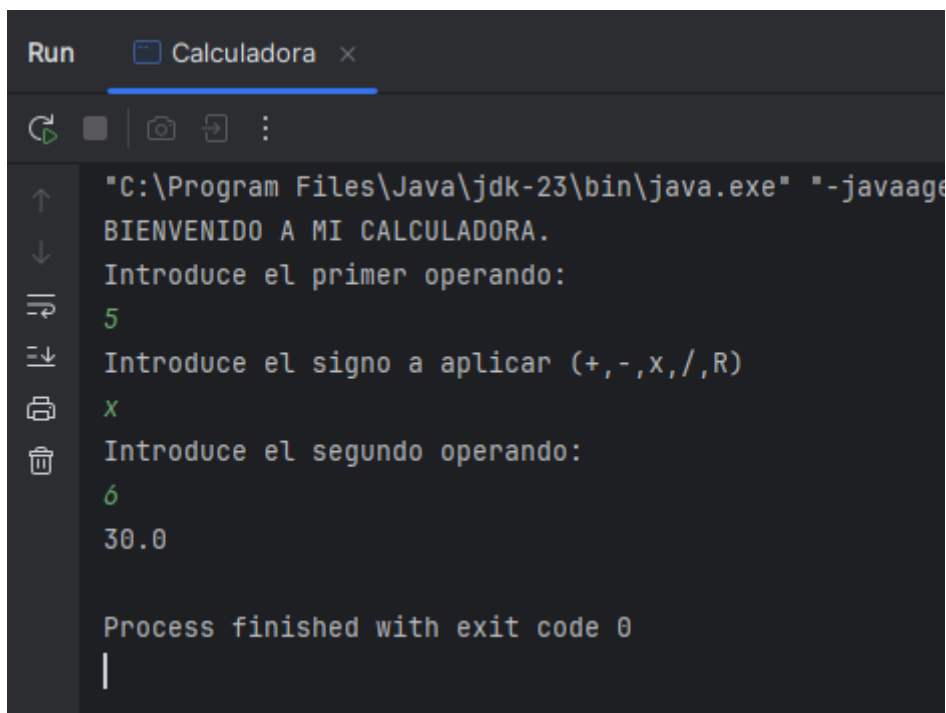
```
"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaag  
BIENVENIDO A MI CALCULADORA.  
Introduce el primer operando:  
12  
Introduce el signo a aplicar (+,-,x,/,R)  
-  
Introduce el segundo operando:  
6  
6.0  
  
Process finished with exit code 0
```

La multiplicación con las minúsculas y mayúsculas:



```
Run  Calculadora x
" C:\Program Files\Java\jdk-23\bin\java.exe" "-java
BIENVENIDO A MI CALCULADORA.
Introduce el primer operando:
8
Introduce el signo a aplicar (+,-,x,/,R)
X
Introduce el segundo operando:
4
32.0

Process finished with exit code 0
```



```
Run  Calculadora x
" C:\Program Files\Java\jdk-23\bin\java.exe" "-java
BIENVENIDO A MI CALCULADORA.
Introduce el primer operando:
5
Introduce el signo a aplicar (+,-,x,/,R)
x
Introduce el segundo operando:
6
30.0

Process finished with exit code 0
|
```

La división entre 0 y normal:

```
Run  Calculadora x
" C:\Program Files\Java\jdk-23\bin\java.exe" "-javaage
BIENVENIDO A MI CALCULADORA.
Introduce el primer operando:
8
Introduce el signo a aplicar (+,-,x,/,R)
/
Introduce el segundo operando:
0
No se puede dividir entre 0.

Process finished with exit code 0
```

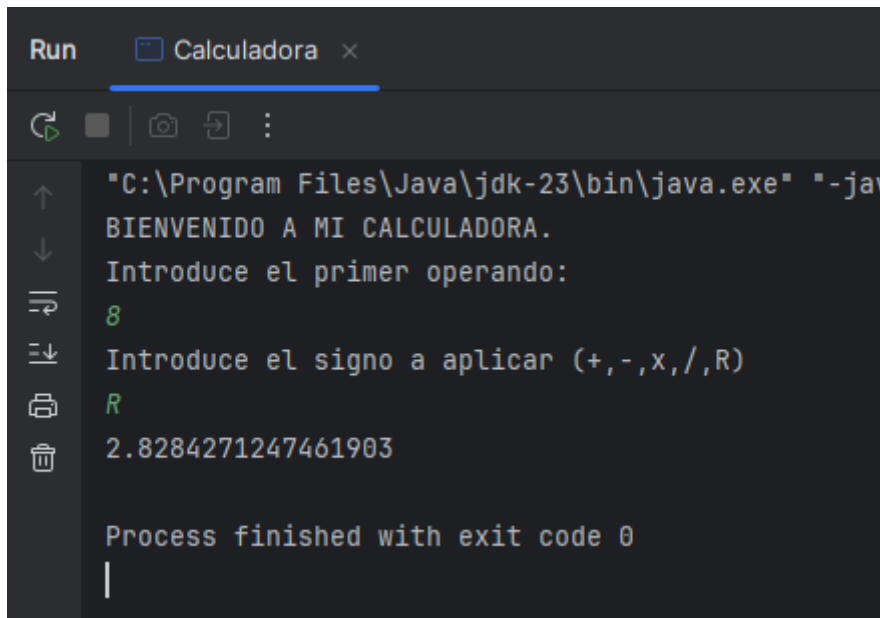
```
Run  Calculadora x
" C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagen
BIENVENIDO A MI CALCULADORA.
Introduce el primer operando:
8
Introduce el signo a aplicar (+,-,x,/,R)
/
Introduce el segundo operando:
1
8.0

Process finished with exit code 0
|
```



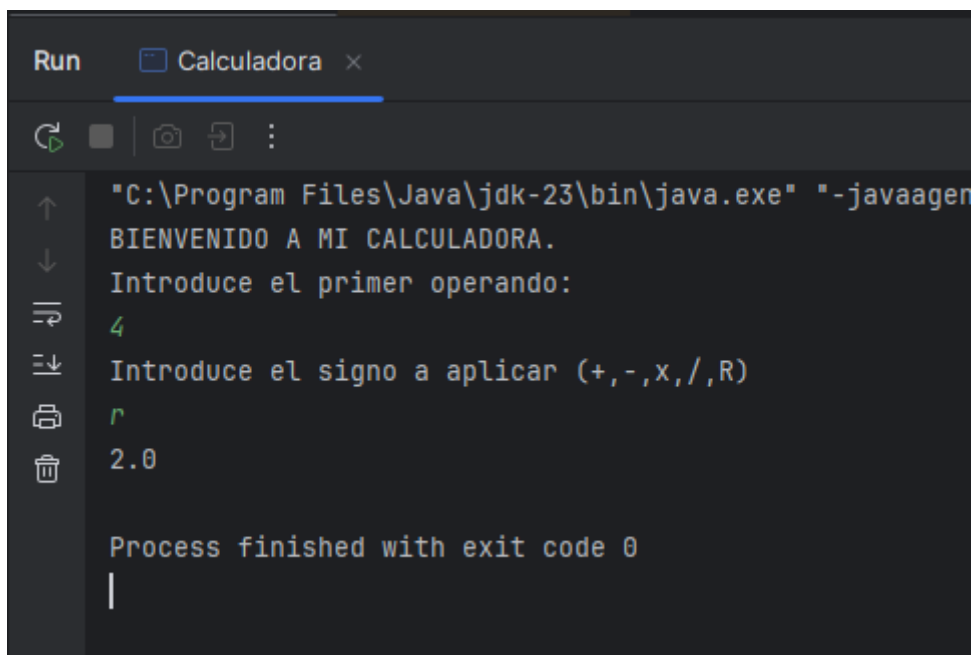
Aquí están las capturas con la raíz cuadrada con la R mayúscula y minúscula y cuando introduces un número negativo.

Como se puede observar, no pide segundo operando.



```
Run  Calculadora x
" C:\Program Files\Java\jdk-23\bin\java.exe" "-jav
BIENVENIDO A MI CALCULADORA.
Introduce el primer operando:
8
Introduce el signo a aplicar (+,-,x,/,R)
R
2.8284271247461903

Process finished with exit code 0
|
```



```
Run  Calculadora x
" C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagen
BIENVENIDO A MI CALCULADORA.
Introduce el primer operando:
4
Introduce el signo a aplicar (+,-,x,/,R)
r
2.0

Process finished with exit code 0
|
```

```
Run  Calculadora x
"\"C:\Program Files\Java\jdk-23\bin\java.exe\" \"-javaage
BIENVENIDO A MI CALCULADORA.
Introduce el primer operando:
-4
Introduce el signo a aplicar (+,-,x,/,R)
r
Introduce un número correcto para este formato.

Process finished with exit code 0
```

Aquí el programa con sus diferentes fallos y cómo los corrige:

```
Run  Calculadora x
"\"C:\Program Files\Java\jdk-23\bin\java.exe\"
BIENVENIDO A MI CALCULADORA.
Introduce el primer operando:
df
Introduce el primer operando bien.
Introduce el primer operando:
6
Introduce el signo a aplicar (+,-,x,/,R)
ddddd
no es correcto.
Introduce el signo a aplicar (+,-,x,/,R)
x
Introduce el segundo operando:
adfasd
Introduce el segundo operando bien.
Introduce el segundo operando:
54
324.0

Process finished with exit code 0
```