

Práctica 3.4 Depuración (CE3.c & CE3.d & CE3.e)

1. El ejercicio. En el siguiente ejercicio vamos a trabajar con la herramienta de depuración de NetBeans.

Nuestra tarea va a consistir en realizar pruebas sobre la siguiente aplicación:

_La Gran Calculadora

El programa comenzará dando la bienvenida al usuario.

Tras ello le pedirá los dos operandos, que serán de tipo entero.

Las operaciones disponibles serán: Sumar, restar, multiplicar, dividir, y elevar el primer número al segundo.

Además, también existirá una opción de “Salir”.

En la clase “Calculadora”, disponemos de dos atributos: operando1 y operando2.

También tendremos definidas las operaciones “getter” y “setter” de los dos operandos, y las operaciones matemáticas “sumar”, “restar”, “multiplicar”, “dividir” y “elevar”.

Ahora, corresponde al equipo de pruebas realizar las pruebas pertinentes sobre el proyecto.

¿Qué encontraremos?

```

1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5  package ejercicio4_3_ed;
6
7  /**
8   *
9   * @author violi
10  */
11 public class Calculadora {
12
13     private int operando1;
14     private int operando2;
15
16     public Calculadora() {
17         operando1=0;
18         operando2=0;
19     }
20
21     public int sumar() {
22         return operando1 + operando2;
23     }
24
25     public int restar() {
26         return operando2-operando1;
27     }
28
29     public int multiplicar(){
30         return operando1 * operando2;
31     }
32 }

```

Nos encontramos el proyecto, con dos ficheros .java uno llamado Calculadora.java y otro llamado Ejercicio3.java

Establecemos puntos de ruptura en nuestro código main.

Que vendría a ser, Ejercicio3.java nuestro archivo main de nuestro proyecto.

```

45 // Realizar la operación correspondiente
46 switch (opcion) {
47     case 1:
48         System.out.println("Resultado: " + miCalculadora.sumar());
49         break;
50     case 2:

```

Y lo enmarcamos, este punto de ruptura lo que hace es establecer un punto de ruptura que al ejecutar el programa para en esa línea.

```

47
48         System.out.println("Resultado: " + miCalculadora.sumar());
49     case 2:

```

ejercicio4_3_ed.Ejercicio3 > main > do ... while (opcion != 0) > switch (opcion) > case 3: >

Output > Variables

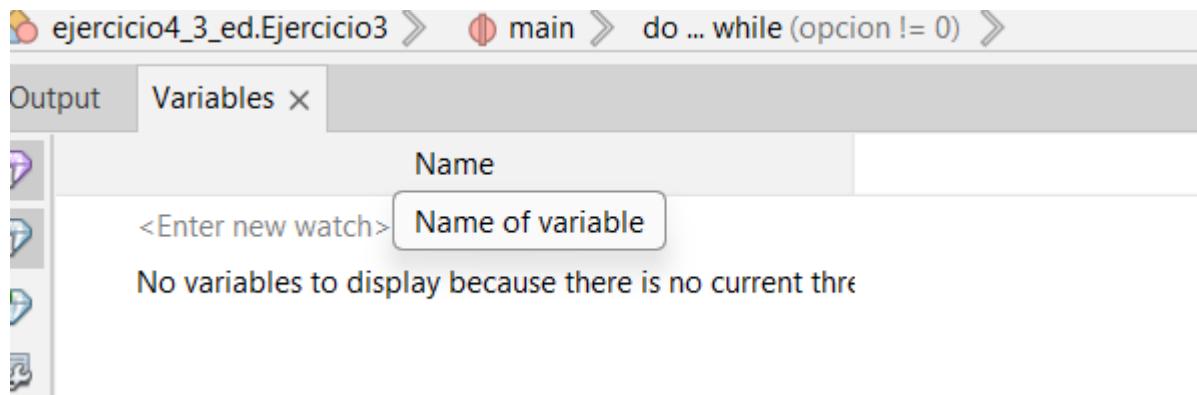
Ejercicio4_3_ED (debug) > Debugger Console >

Listening on javadebug
User program running
LineBreakpoint Ejercicio3.java : 48 successfully submitted.

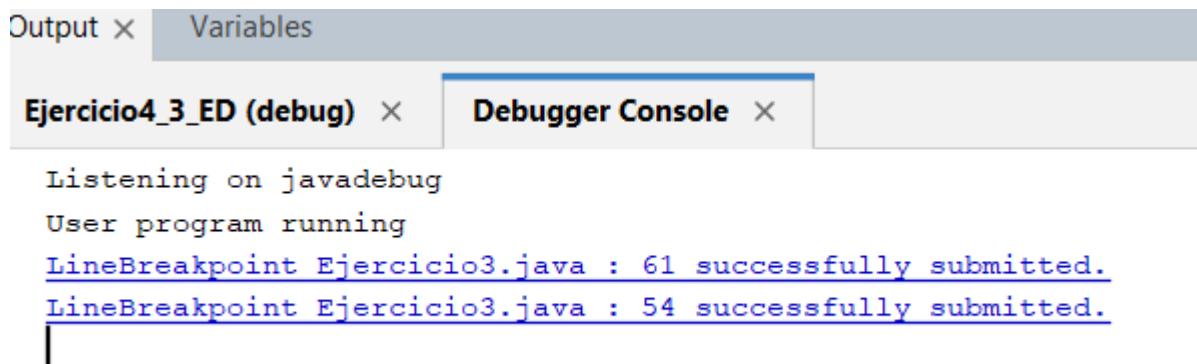
Como podemos ver nuestro programa paro el debug del archivo en la linea 48 que es la que enmarcamos y esta subrayada de color rojo.

Para sacar a punto a que el debug paro en esta linea de nuestro código usamos funciones de window, como por ejemplo variables.

Para ello vamos a la opción window --> debugging --> Variables



Ahora con dos puntos de ruptura en la línea 54 y 61 empezamos nuestra depuración dandole a nuestro archivo main clic derecho y escogemos la opción debug file como se ve a continuación

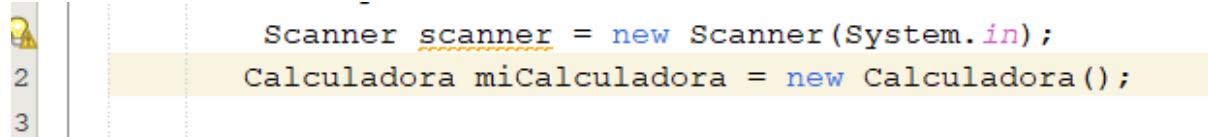


Se nos crean dos ventanas en la que en una está el debug de nuestro archivo y en la otra tenemos el debugger console, donde nos pone línea de break point en la línea 64 y 54 enviada así vemos de forma mas clara como funciona la función debug file.

Ponemos en practica nuestra prueba de caja blanca y detectamos un error en la linea 22 de nuestro codigo, cambiamos esa linea por

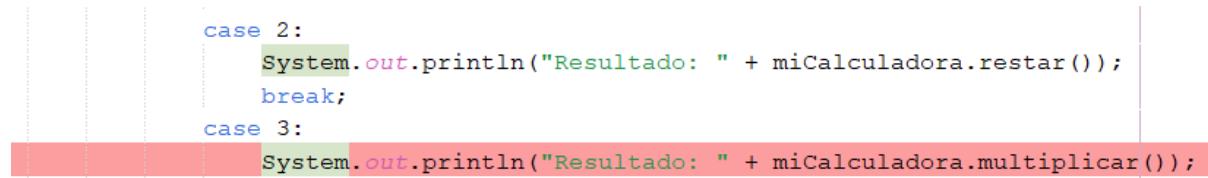
```
Calculadora miCalculadora = new Calculadora();
```

Debido a que si inicializa a null dicho valor ya es null y cuando lo intentamos usar el programa lo detecta como null.

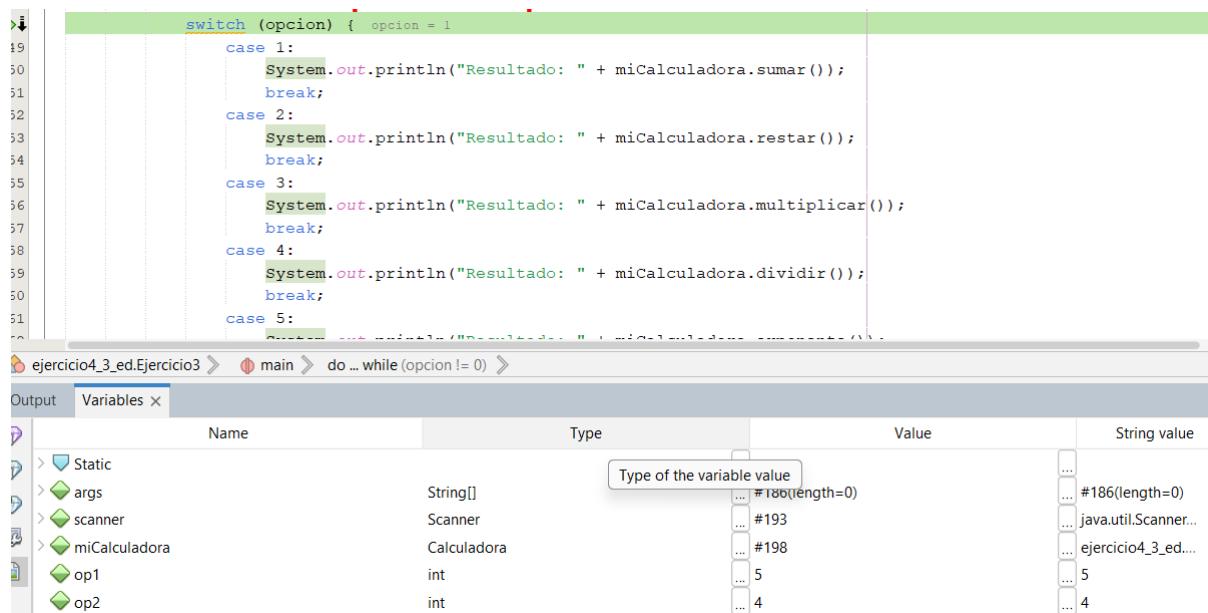


```
Scanner scanner = new Scanner(System.in);
Calculadora miCalculadora = new Calculadora();
```

Tambien detectamos otro error de logica en nuestro codigo en el case 2 debido a que suma no resta, asi es ya correctamente.



```
case 2:
    System.out.println("Resultado: " + miCalculadora.sumar());
    break;
case 3:
    System.out.println("Resultado: " + miCalculadora.multiplicar());
```



```
switch (opcion) { opcion = 1
    case 1:
        System.out.println("Resultado: " + miCalculadora.sumar());
        break;
    case 2:
        System.out.println("Resultado: " + miCalculadora.restar());
        break;
    case 3:
        System.out.println("Resultado: " + miCalculadora.multiplicar());
        break;
    case 4:
        System.out.println("Resultado: " + miCalculadora.dividir());
        break;
    case 5:
```

Name	Type	Value	String value
Static	String[]	... #186(length=0)	... #186(length=0)
args	Scanner	... #193	... java.util.Scanner...
scanner	Calculadora	... #198	... ejercicio4_3_ed....
miCalculadora	int	5	5
op1	int	4	4
op2			

Una ves nuestro codigo solucionado ya nos deja acceder y ver las variables y nuestro programa funciona correctamente aplicando pruebas de caja blanca y depuracion.

FIN