UT05. DISEÑO Y REALIZACIÓN DE PRUEBAS

Entornos de Desarrollo

1DAW – C.I.F.P. Carlos III - Cartagena

3. Procedimientos y casos de prueba

Procedimientos y casos de prueba

- Según el IEEE

  *Un **caso de prueba** es un*

  - *conjunto de entradas,*
  - *condiciones de ejecución y*
  - *resultados esperados,*

  desarrollados para un objetivo particular, como, por ejemplo, ejercitar un camino concreto de un programa o verificar el cumplimiento de un determinado requisito, incluyendo toda la documentación asociada.

- Complejidad de las aplicaciones informáticas → imposibilidad de probar todas la combinaciones → asegurar que los casos de prueba obtienen un nivel aceptable de probabilidad de detección de errores

4. Herramientas de depuración

Herramientas de depuración

- Herramientas de depuración de un IDE
  - Inclusión de **puntos de ruptura**
  - **Ejecución paso a paso** de cada instrucción
  - **Ejecución por procedimiento**
  - **Inspección de variables**
  - etc.

Herramientas de depuración

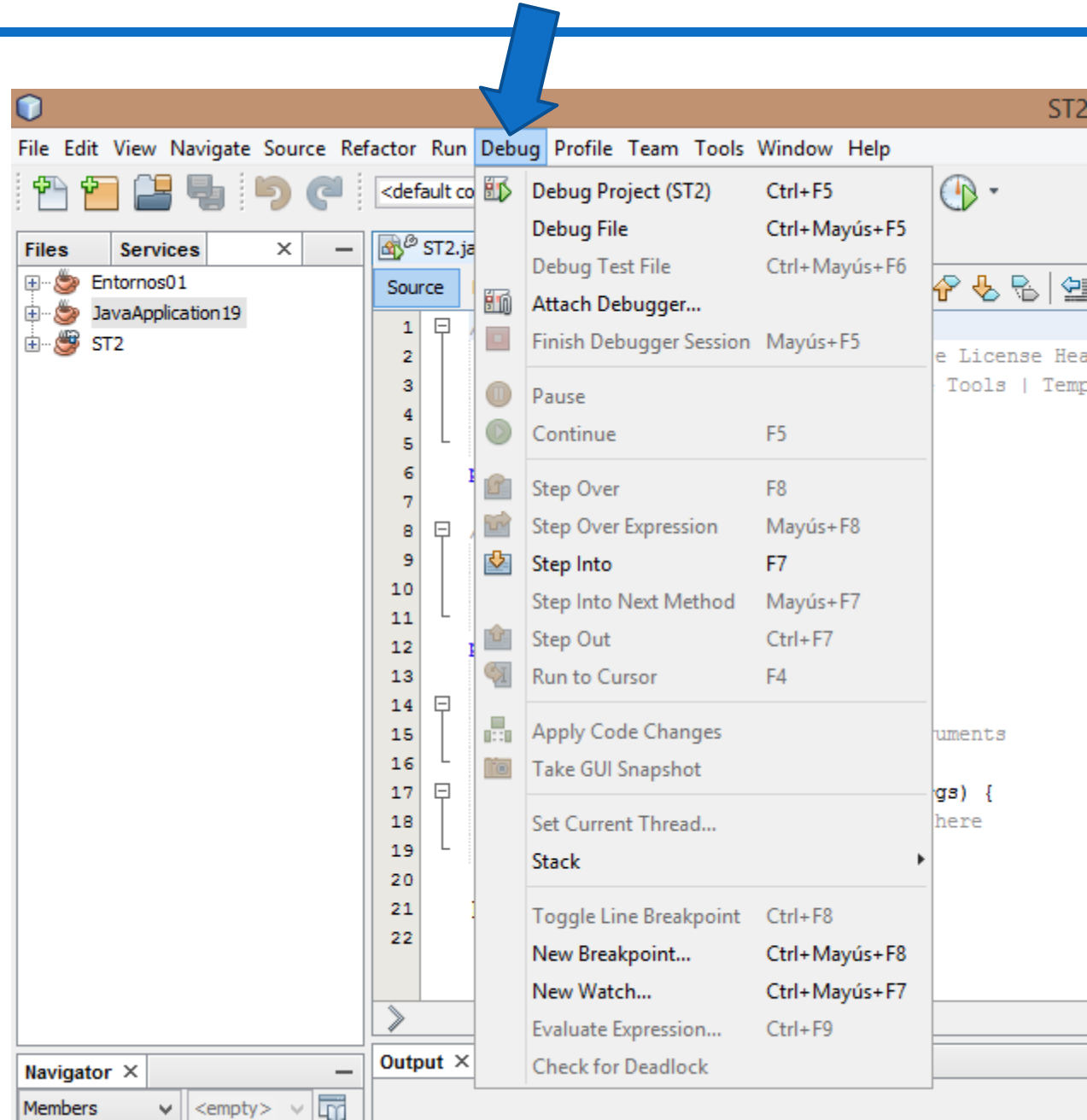
- Tipos de errores en el proceso de desarrollo
  - **Errores de compilación:** errores de sintaxis.
    - P.e. no poner punto y coma al final de una sentencia
    - El entorno avisará del error
    - No se puede continuar hasta corregir el error
  - **Bugs o errores de ejecución o lógicos**
    - No evitan que el programa se compile con éxito
    - Genera resultados inesperados, termina de forma inesperada o no termina nunca
- El proceso de depuración ➔ comienza con la ejecución de un caso de prueba.
- Se **evalúan los resultados** de la ejecución
- Se comprueba si hay **falta de correspondencia** entre los resultados esperados y los obtenidos.

Herramientas de depuración

- Resultados
  - Se encuentra el error ➔ se corrige y se elimina
  - No se encuentra ➔ estudiar las posibles causas, diseñar casos de prueba para verificar la causa, repetir las pruebas para identificar los errores y corregirlos

Depurador

- **Depurador**
  - Permite supervisar la ejecución de los programas, para localizar y eliminar los **bugs.**
  - Condición imprescindible: debe compilarse con éxito
    - Puntos de ruptura
    - Tipos de ejecución
    - Examinadores de variables



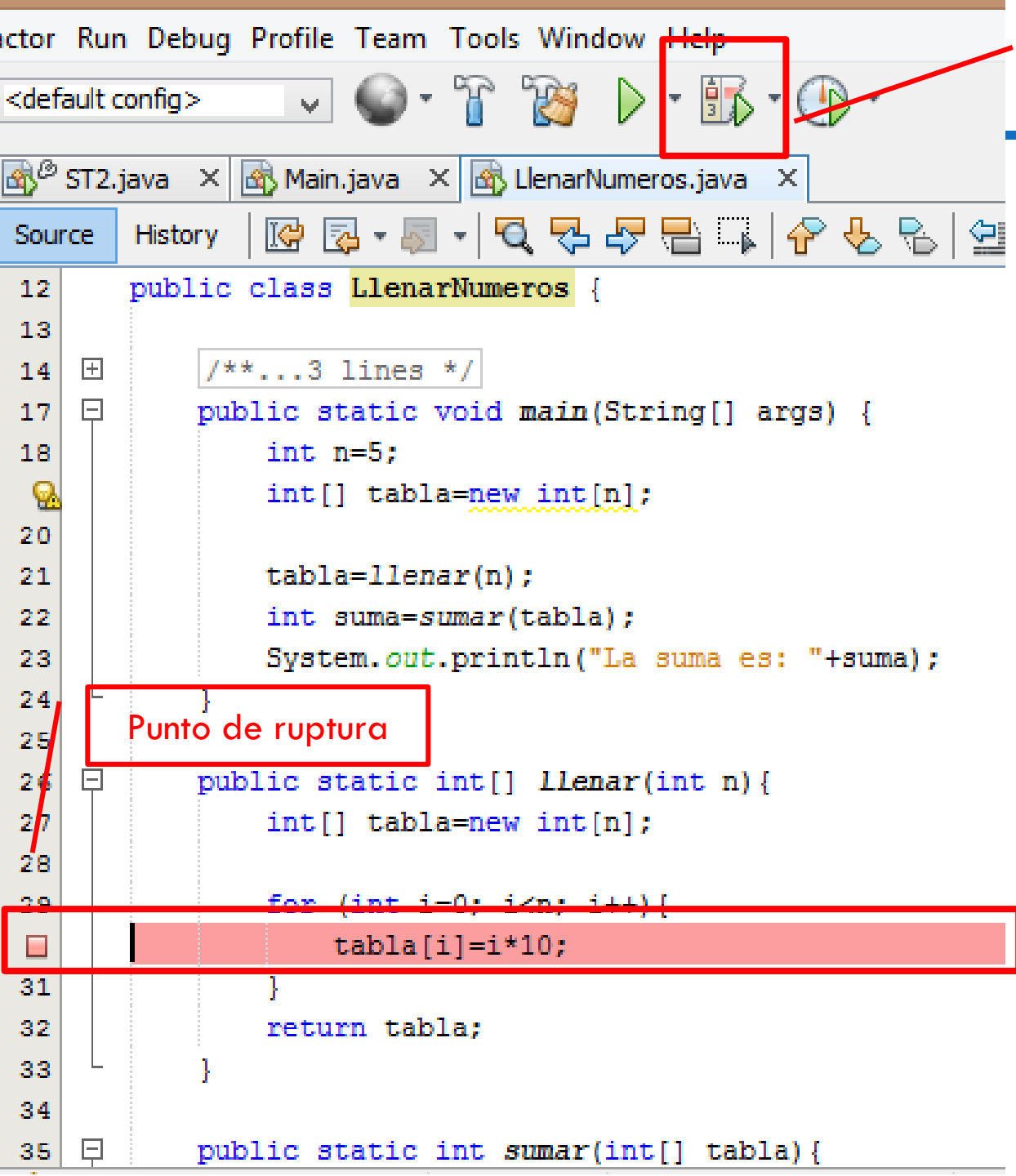
```java
public class LlenarNumeros {

    /**...3 lines */
    public static void main(String[] args) {
        int n=5;
        int[] tabla=new int[n];

        tabla=llenar(n);
        int suma=sumar(tabla);
        System.out.println("La suma es: "+suma);
    }

    public static int[] llenar(int n){
        int[] tabla=new int[n];

        for (int i=0; i<n; i++){
            tabla[i]=i*10;
        }
        return tabla;
    }

    public static int sumar(int[] tabla){
        int suma=0;
        int n=tabla.length;

        for(int i=0; i<n; i++)
            suma=suma+tabla[i];
        return suma;
    }
}
```

Copia el siguiente código en Netbeans.

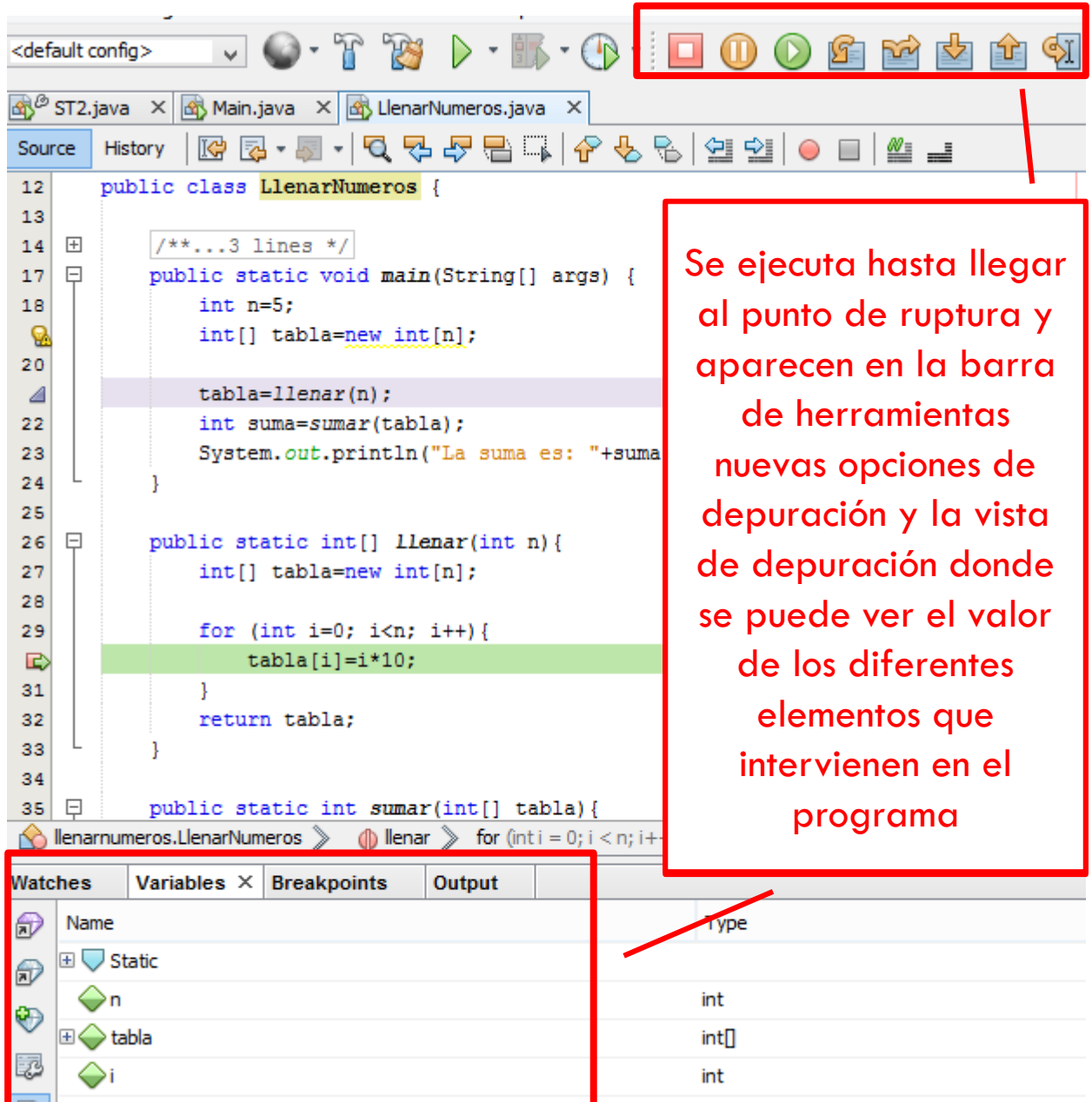
¿Qué hace?

Depurador. Puntos de ruptura

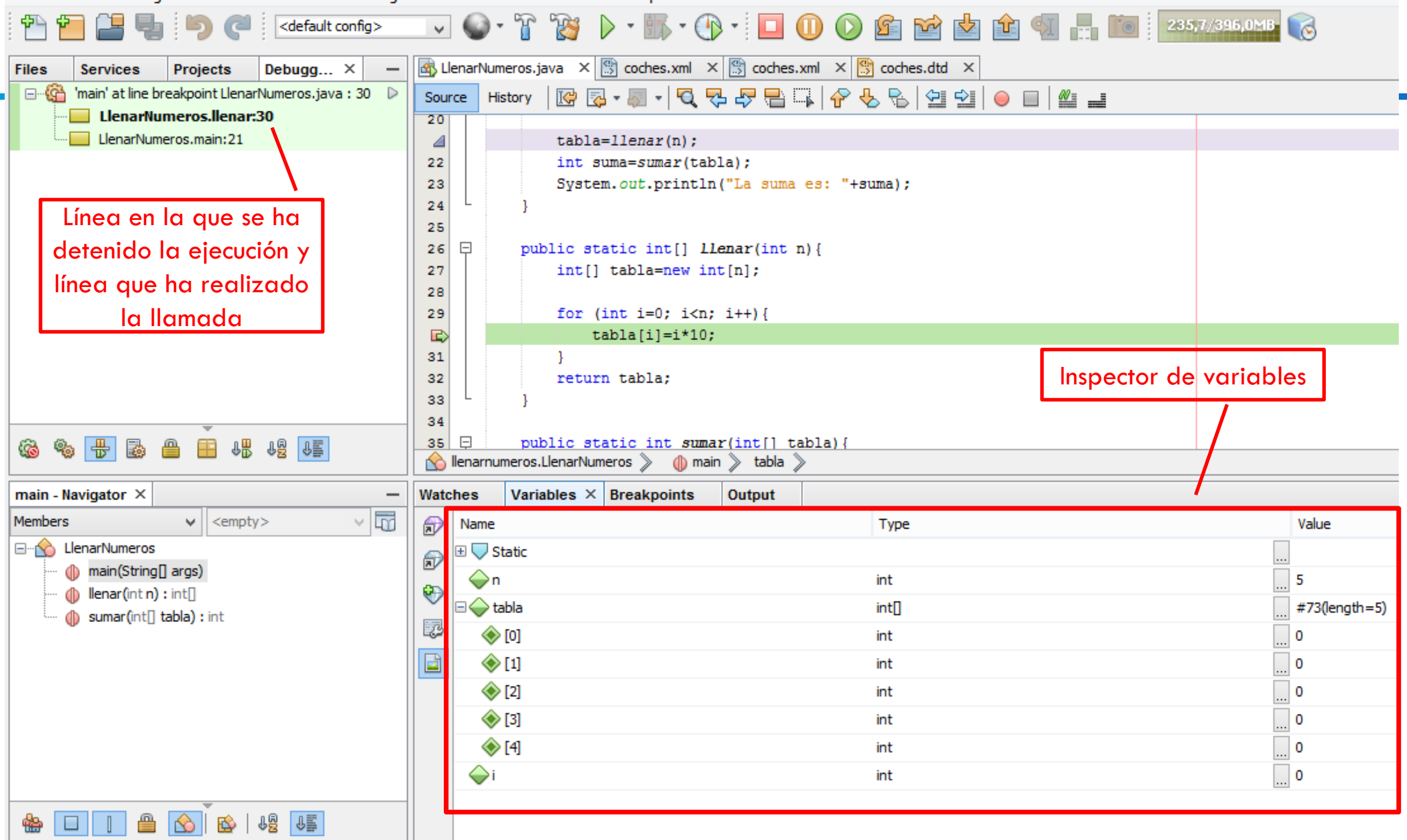
- **Puntos de ruptura** o **breakpoint y seguimiento**
  - Seleccionar la línea de código donde queremos que el programa se pare
  - Inspeccionar variables, o realizar una **ejecución paso a paso**, para verificar la corrección del código
  - Son marcadores que pueden establecerse en cualquier línea de código ejecutable.
  - Se pueden insertar varios puntos de ruptura
  - Se pueden eliminar fácilmente
    - Se establecen haciendo clic en el margen izquierdo o botón derecho > Breakpoint
    - Debug > New breakpoints



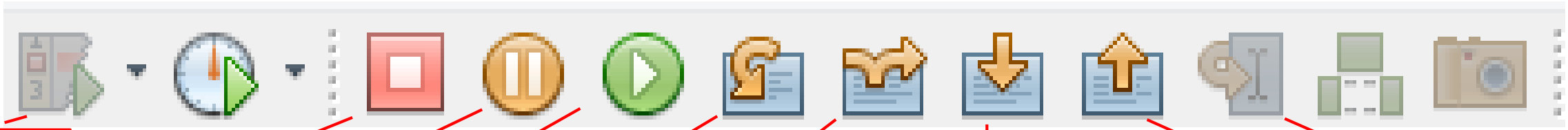
Depurar proyecto
(Debug Project)



Se ejecuta hasta llegar al punto de ruptura y aparecen en la barra de herramientas nuevas opciones de depuración y la vista de depuración donde se puede ver el valor de los diferentes elementos que intervienen en el programa


Línea en la que se ha detenido la ejecución y línea que ha realizado la llamada
Inspector de variables
'main' at line breakpoint LlenarNumeros.java : 30
LlenarNumeros.llenar:30
LlenarNumeros.main:21
LlenarNumeros.java
coches.xml
coches.xml
coches.dtd
Source
History
tabla=llenar(n);
int suma=sumar(tabla);
System.out.println("La suma es: "+suma);
public static int[] llenar(int n){
int[] tabla=new int[n];
for (int i=0; i<n; i++){
tabla[i]=i*10;
return tabla;
public static int sumar(int[] tabla){
llenarnumeros.LlenarNumeros
main
tabla
main - Navigator
Members
LlenarNumeros
main(String[] args)
llenar(int n) : int[]
sumar(int[] tabla) : int
Watches
Variables
Breakpoints
Output
Name
Type
Value
Static
n
int
5
tabla
int[]
#73(length=5)
[0]
int
0
[1]
int
0
[2]
int
0
[3]
int
0
[4]
int
0
i
int
0

Depurador. Barra de herramientas



Debug Proyect

Finish Debugger Session

Pause

Continue

Salta los métodos

Step Over: Ejecuta una línea de código. Si la instrucción es una llamada a un método, ejecuta el método sin entrar dentro del código del método..

Step Over Expression: Si una expresión tiene múltiples llamadas a los métodos, se puede utilizar **Step Over Expression** y ver el valor de cada llamada de método en la expresión en la ventana Variables locales

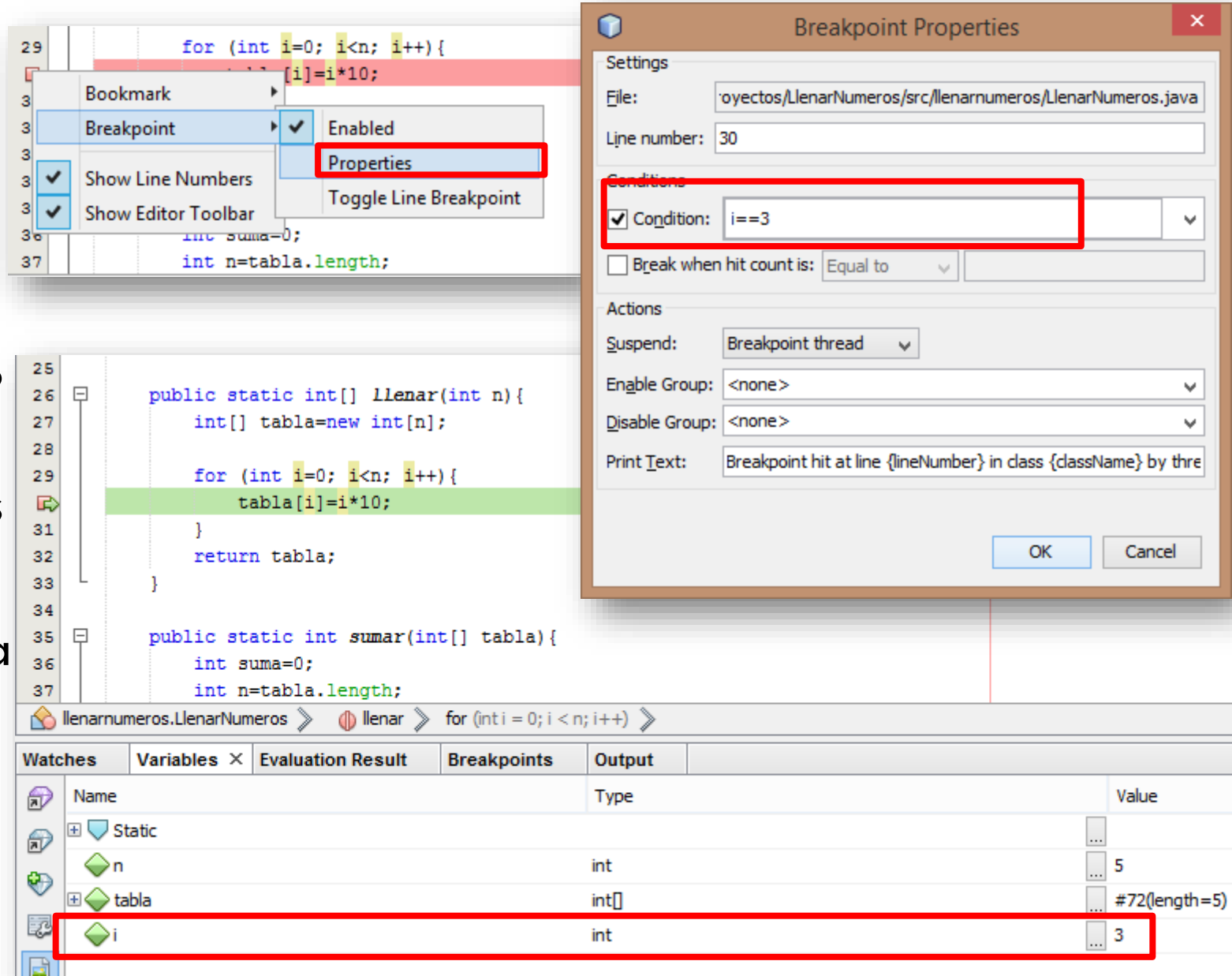
Step Into: Si la instrucción es una llamada a un método, salta al método y continúa la ejecución por la primera línea del método.

Step Out: Ejecuta una línea del fuente. Si la línea de código actual se encuentra dentro de un método, se ejecutarán todas las instrucciones que queden del método y se vuelve a la instrucción desde la que se llamó al método

Run to Cursor: Se ejecuta el programa hasta la instrucción donde se encuentra el cursor.

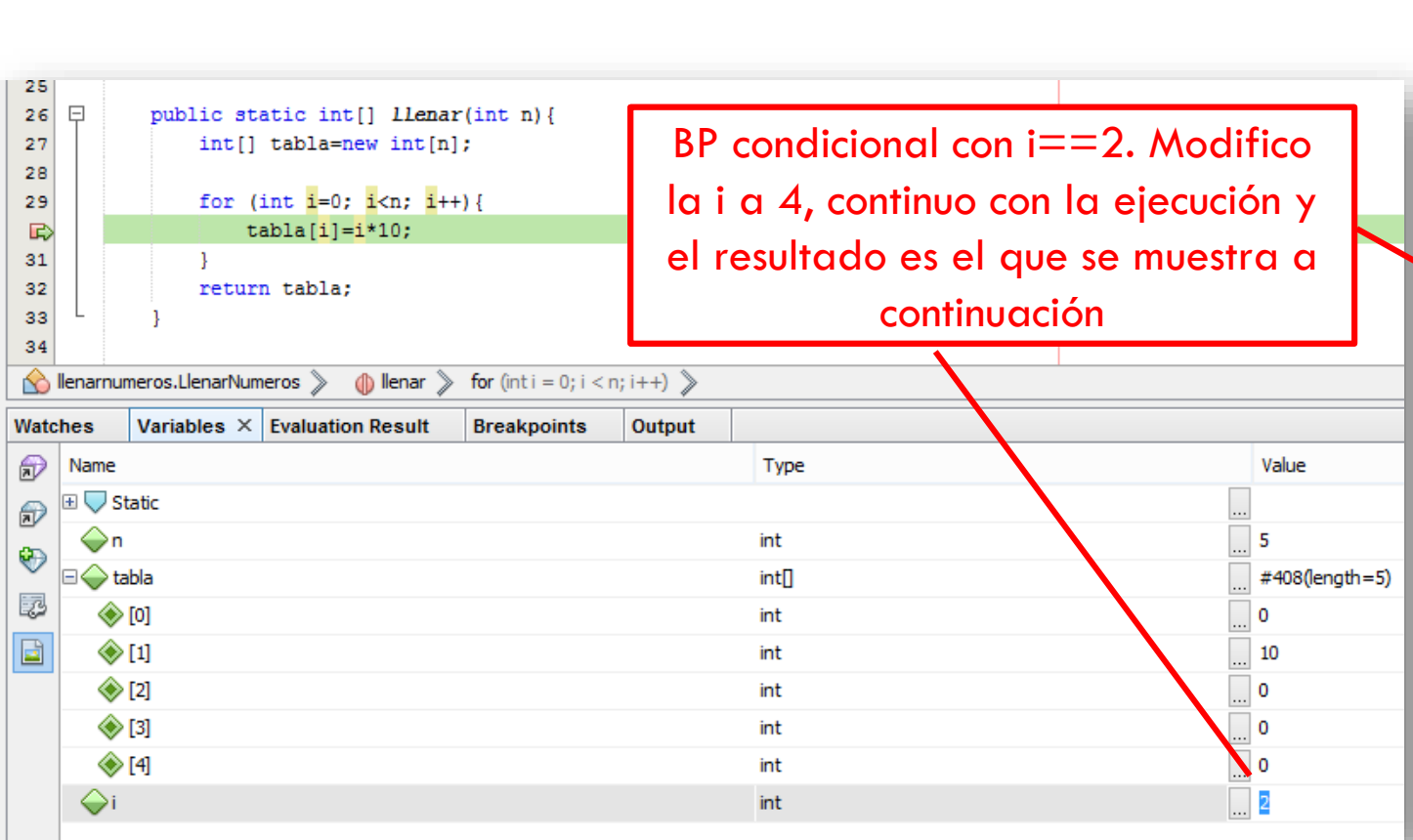
Depurador. Puntos de ruptura condicionales

- **BP Condicionales:** el programa para sólo si se cumple la condición especificada.
  - Nuevo BP
  - Menú contextual en el BP >> Breakpoint>>Propierties
  - Condition ➔ i==3
  - Debug Project ➔ ejecuta hasta que se cumple la condición

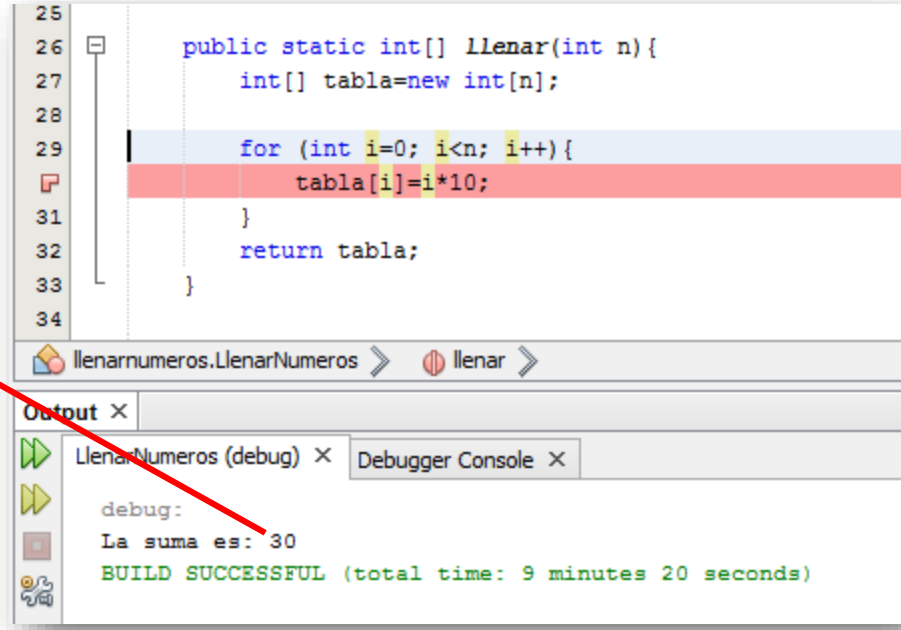


Depurador. Examen y modificación de variables

- Uno de los elementos más importantes del proceso de depuración de un programa.
- NetBeans➔Ventana de Inspección > modificar el contenido de las variables y continuar con la ejecución.



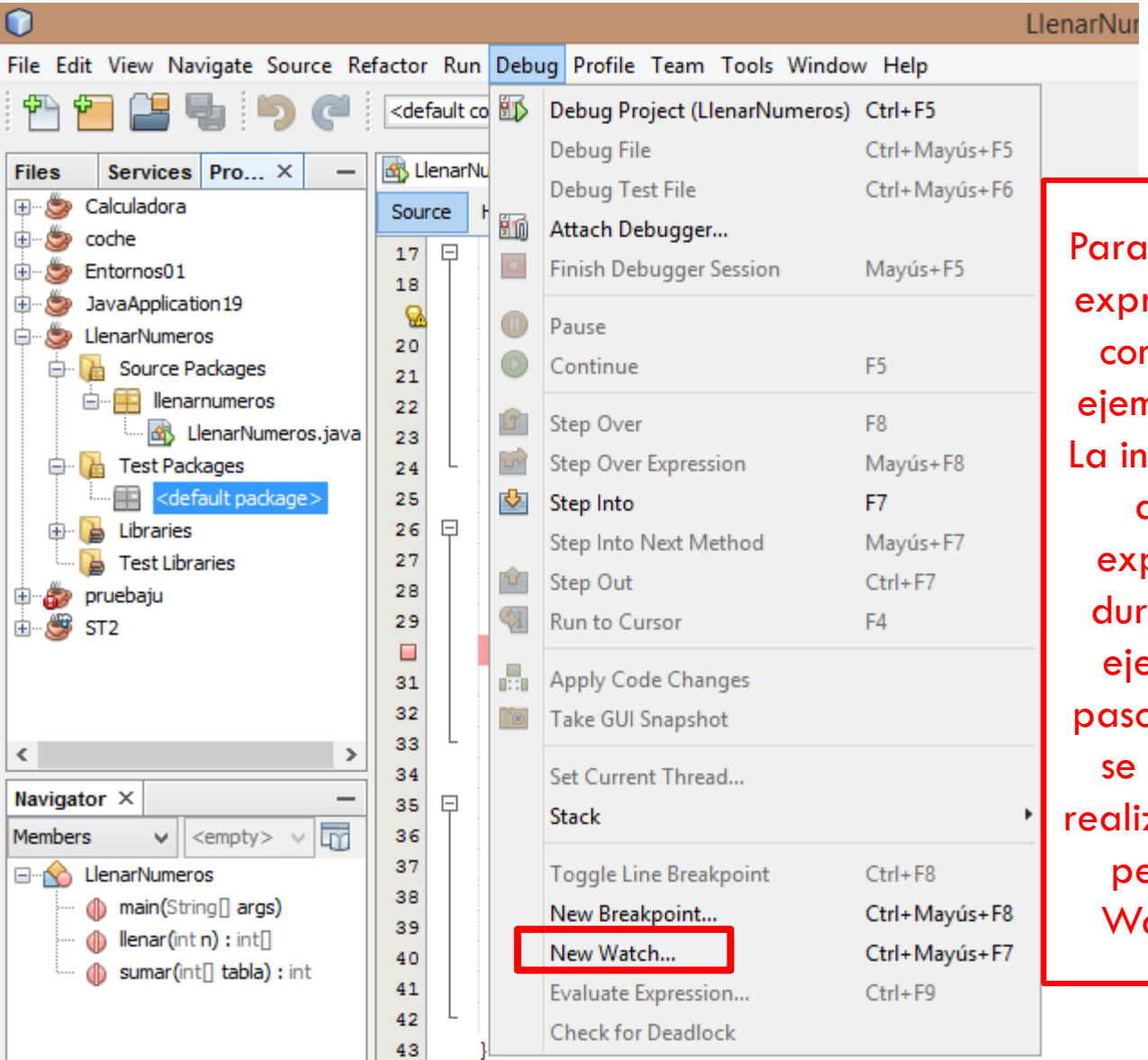
BP condicional con i==2. Modifico la i a 4, continuo con la ejecución y el resultado es el que se muestra a continuación



Depurador. Ejemplo

- **Ejemplo:** Escribir un programa en Java en Netbeans que resuelva la potencia de un número. Depurar y ejecutar observando la evolución de las variables en la **Ventana de Inspección**.

Depurador. Examen y modificación de variables



Para evaluar expresiones, como por ejemplo i*n. La inspección de la expresión durante la ejecución paso a paso se puede realizar en la pestaña Watches.