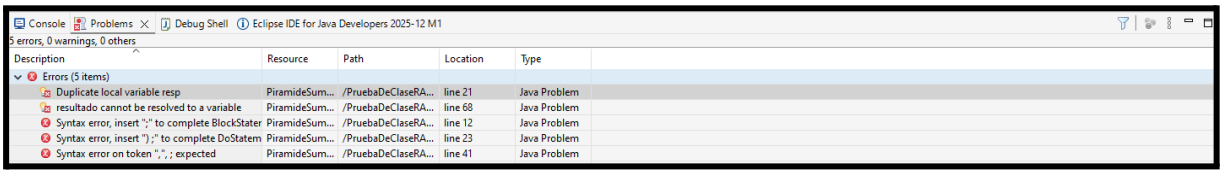
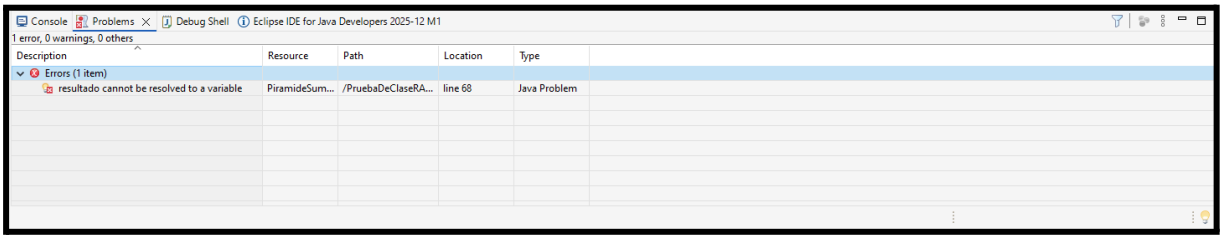
- ¿Qué es un error sintáctico?
  Un error sintáctico en el código es un error al escribir algo en el código como el punto y coma (;) al final de una instrucción o poner una coma (,) en vez de un punto y coma (;).

- ¿Cuándo los visualizamos?
  Eclipse, al intentar ejecutar el código, te muestra una lista de errores que hay en él.

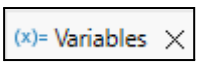
- ¿Podemos depurar con errores de sintaxis?
  No, ya que el código directamente se detiene al detectar uno para que la persona que esté programando lo arregle.





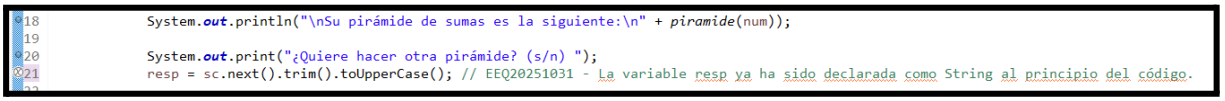
- Proporciona una mini guía de cómo realizar la depuración de un programa *(si te es más fácil, elige uno de los fallos lógicos y úsalo de ejemplo)*
  Primero hay que hacer doble clic en el lateral para marcar donde queremos que pare el depurador y luego pulsar el botón de Debug de Eclipse

  Entonces el programa empezará a ejecutarse hasta donde hayamos marcado primero.
  Para continuar pulsaremos f8, para pasar a la siguiente función f6 y para ver qué está pasando dentro de una función f5, cuando queramos salir de la función usaremos f7.
  Además en el lateral podemos ver las variables

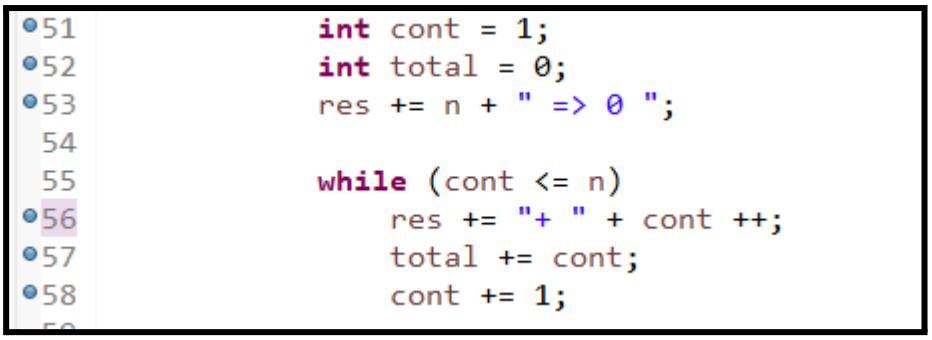


  lo que nos ayudara a entender mejor que es lo que esta realizando nuestro código.

- Agrega capturas con los puntos de ruptura y las vistas que consideres más importantes a la hora de depurar un programa y por qué.



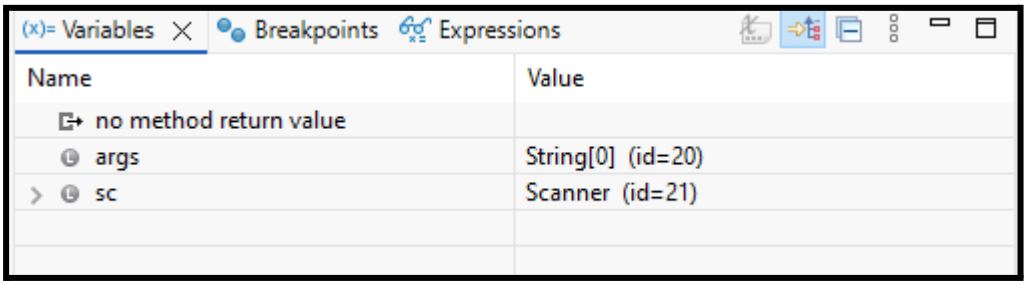
Los puntos de ruptura son necesarios por ejemplo cuando se imprime algo en pantalla para ver que el texto se muestre correctamente y en los puntos donde se les de valor a las variables.



Y las vistas mas importantes a la hora de depurar son:

La consola:

Las variables:



El editor:

```java
package tema2_prueba;

import java.util.Scanner;

public class PiramideSumas {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String resp;

        do {
            int num = leerEntero(sc, "Introduzca un número: "); // EEQ20251031 - Falta punto y

            while (num < 0 && num > 20) {
                num = leerEntero(sc, "**Valor fuera de rango** Introduzca un número entre 0 y 2
            }

            System.out.println("\nSu pirámide de sumas es la siguiente:\n" + piramide(num));

            System.out.print("¿Quiere hacer otra pirámide? (s/n) ");
            resp = sc.next().trim().toUpperCase(); // EEQ20251031 - La variable resp ya ha sido

        } while (resp.equals("S")); // EEQ20251031 - Falta paréntesis antes del punto y coma.

        borrarConsola();
        System.out.println("¡¡¡PROGRAMA FINALIZADO!!!");

        sc.close();
    }

    private static int leerEntero(Scanner sc, String mensaje) {
        System.out.print(mensaje);
        while (!sc.hasNextInt()) {
            System.out.print("**Valor no válido** Introduzca un número entero: ");
            sc.next();
        }
```

- ¿Dónde colocaste los breakpoints y qué valores viste que confirmaron el fallo?

  Los puse en los puntos en los que se declaraban las variables y el valor que vi que confirmaba el fallo fue el contador ya que seguía contando infinitamente e impedía que el bucle terminase.

- ¿En qué situación debemos usar `Step Into (F5)`, `Step Over (F6)`, `Step Return (F7)` y `Resume (F8)`?

  Step Into (f5) cuando queramos ver que esta ocurriendo en una función, Step Over (f6) cuando queremos ver que ocurre con la siguiente instrucción, Step Return (f7) tras usar Step Into (f5) para salir de la función y Resume (f8) para continuar depurando el código.