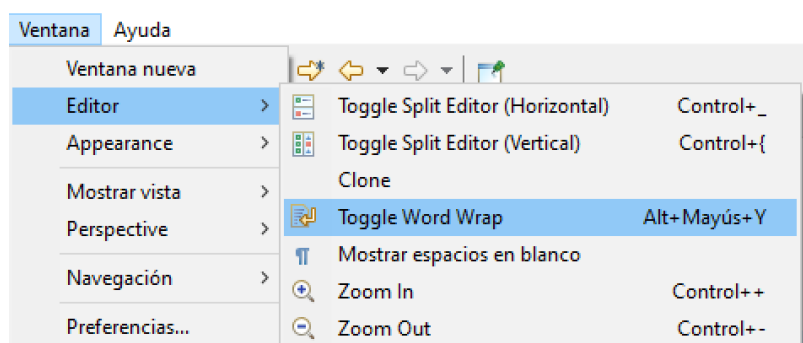


TRABAJAR EN ECLIPSE

CONTENIDOS

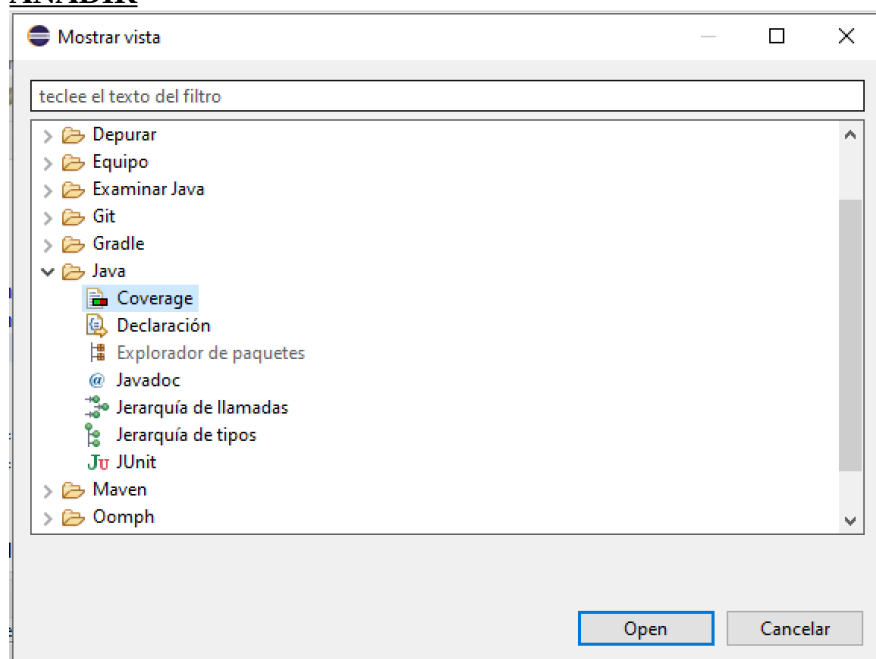
1. MENÚ VENTANA.....	1
1.1. PERSPECTIVAS.....	3
1.2. PREFERENCIAS.....	4
2. MENÚ CÓDIGO FUENTE.....	5
2.1. GENERAR MÉTODOS AUTOMÁTICAMENTE.....	5
3. MENÚ EJECUTAR.....	9

1. MENÚ VENTANA

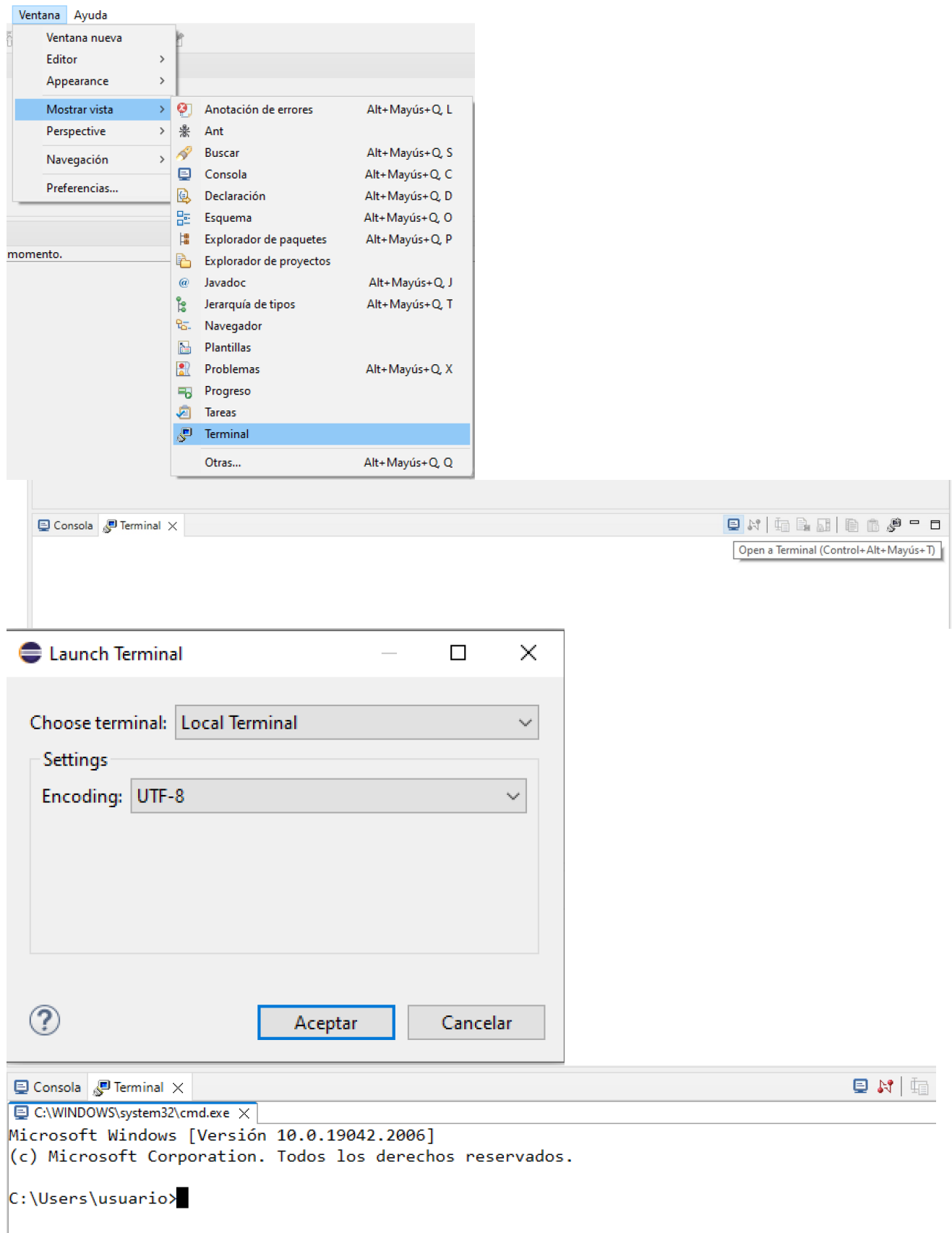


MOSTRAR VISTA

AÑADIR

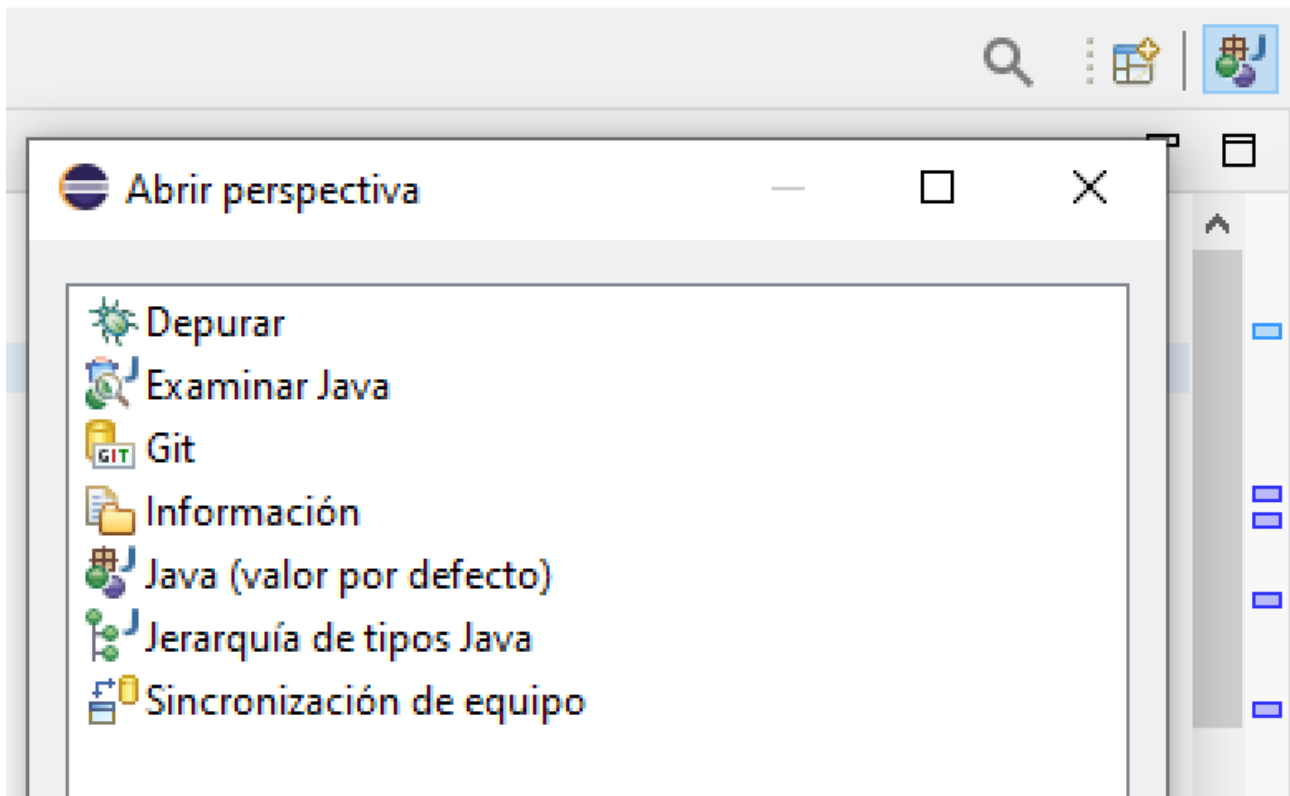


AÑADIR UN TERMINAL

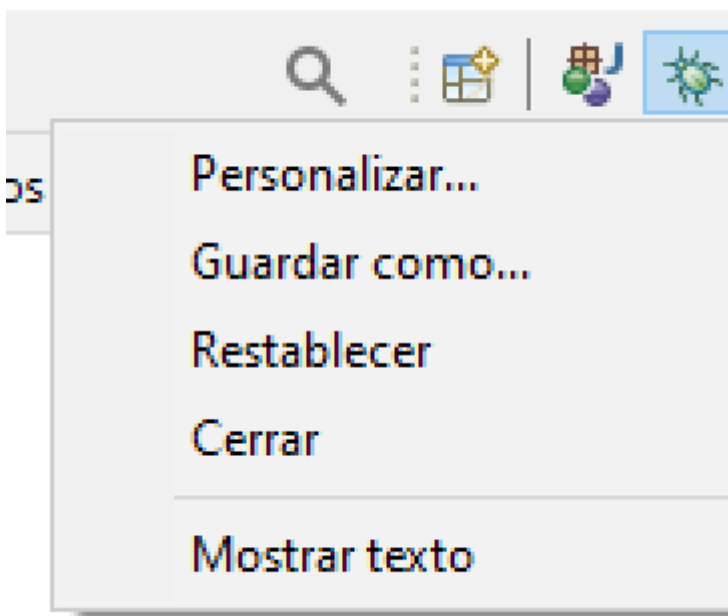


1.1. PERSPECTIVAS

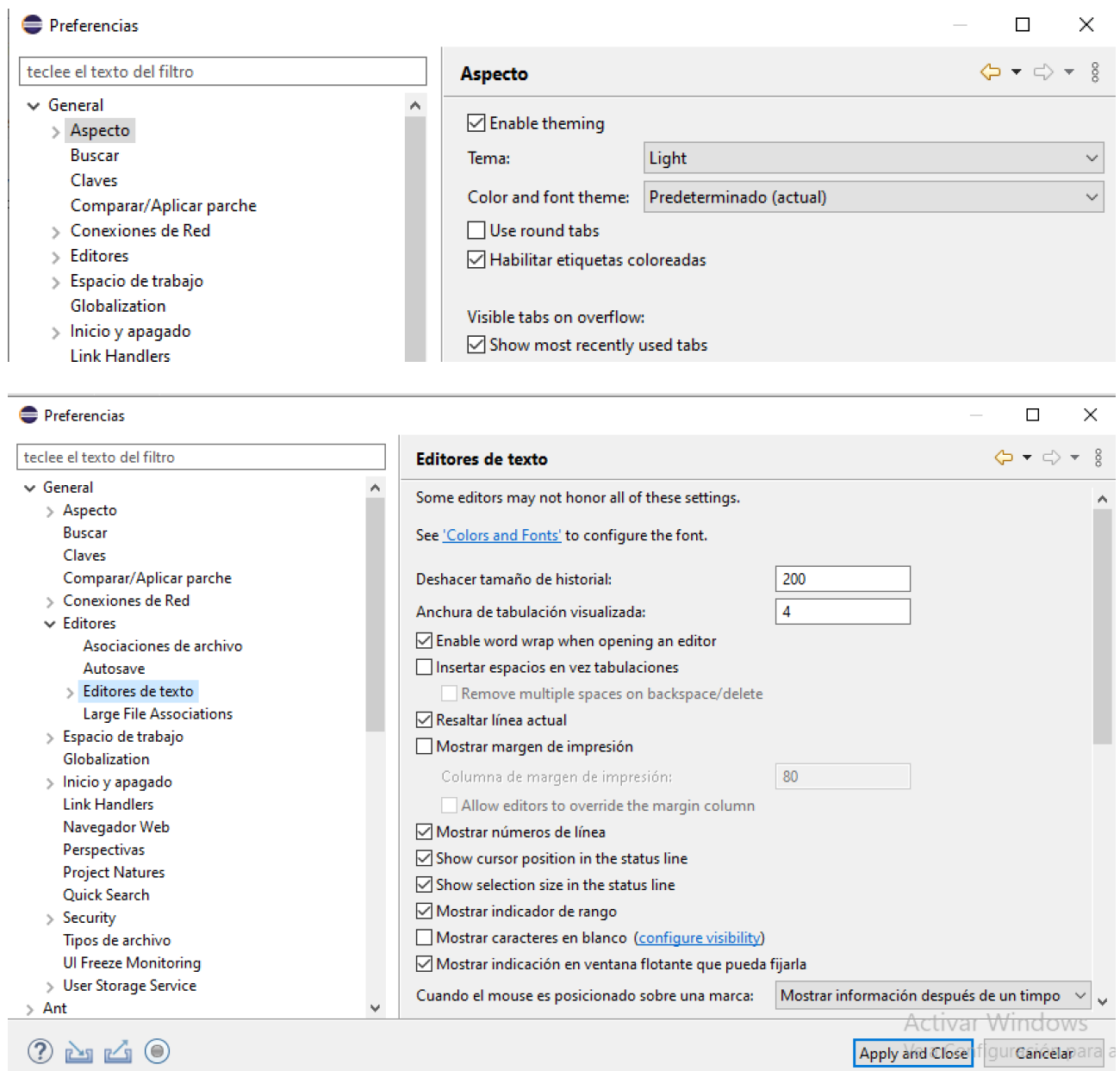
BOTÓN AÑADIR PERSPECTIVA O MENÚ VENTANA



CERRAR PERSPECTIVA: CLIC DERECHO, CERRAR



1.2. PREFERENCIAS



2. MENÚ CÓDIGO FUENTE

Conmutar comentario	Control+7
Añadir comentario de bloque	Control+Mayús+/'
Eliminar comentario de bloque	Control+Mayús+\
Generar comentario de elemento	Alt+Mayús+J
Add Text Block	Control+Mayús+'
Desplazar a la derecha	
Desplazar a la izquierda	
Sangrado correcto	Control+I
Formatear	Control+Mayús+F
Formatear elemento	
Añadir importación	Control+Mayús+M
Organizar importaciones	Control+Mayús+O
Ordenar miembros...	
Limpiar...	
Alterar temporalmente/Implementar métodos...	
Generar métodos de obtención y establecimiento...	
Generar métodos delegados...	
Generar hashCode() y equals()...	
Generar toString()...	
Generar constructor utilizando campos...	
Generar constructores de la superclase...	
Rodear con	Alt+Mayús+Z >
Extraer cadenas de texto...	
Actualizar cadenas de texto extraídas	

2.1. GENERAR MÉTODOS AUTOMÁTICAMENTE

- Crear la clase con método main

Carpeta fuente:

Paquete:

☐ Tipo delimitador:

Nombre:

Modificadores: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static
☒ none ☐ sealed ☐ non-sealed ☐ final

Superclase:

Interfaces:

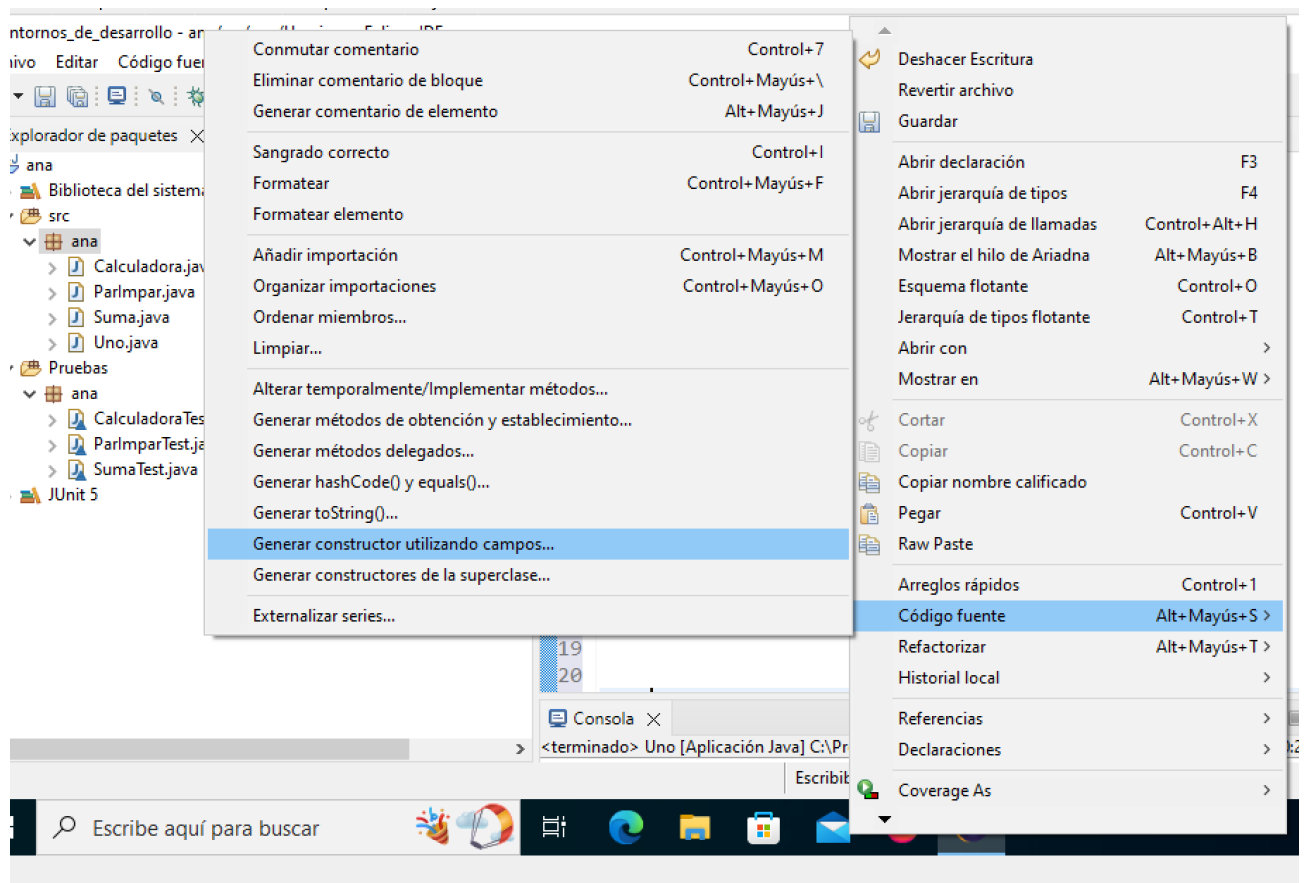
¿Qué apéndices de método desea crear?

☒ public static void main(String[] args)

- Crear los atributos de la clase

```
package ana;
public class Uno {
    //atributos
    private int num1;
    private int num2;
```

- clic derecho o menú Código fuente
- Generar el método constructor utilizando campos
- Generar métodos de obtención y establecimiento (get y set)
- Generar el método toString



Método constructor:

```
public Uno(int num1, int num2) {
    super();
    this.num1 = num1;
    this.num2 = num2;
}
```

Métodos get, set y toString

```

public int getNum1() {
    return num1;
}

public void setNum1(int num1) {
    this.num1 = num1;
}

public int getNum2() {
    return num2;
}

public void setNum2(int num2) {
    this.num2 = num2;
}

@Override
public String toString() {
    return "Uno [num1=" + num1 + ", num2=" + num2 + "]";
}

```

- En main crear una instancia de la clase, mediante la creación de un objeto

```

public static void main(String[] args) {
    // TODO Apéndice de método generado automáticamente
    Uno nuevo= new Uno(30,50);

```

- Llamar a los métodos usando el objeto

```

public static void main(String[] args) {
    // TODO Apéndice de método generado automáticamente
    Uno nuevo= new Uno(30,50);
    System.out.println("Usando método toString: "+nuevo.toString());
    System.out.println("El primer número es: "+ nuevo.getNum1());
    System.out.println("El segundo número es: "+ nuevo.getNum2());
    int result=nuevo.num1*nuevo.getNum2();
    System.out.println ("El resultado de la suma es: "+ result);
    nuevo.setNum1(25);
    nuevo.setNum2(150);
    System.out.println("Usando método toString: "+nuevo.toString());
}

```

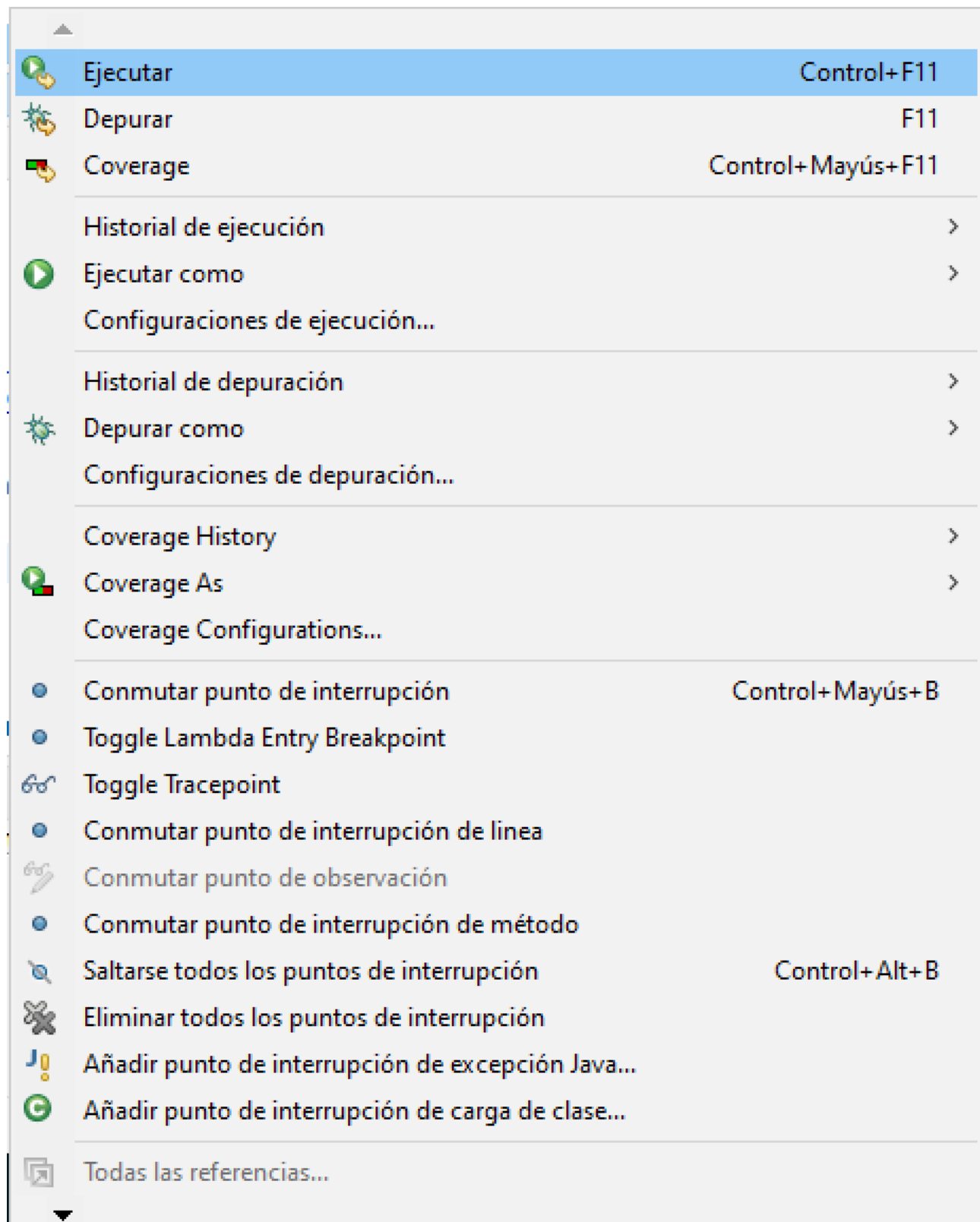
- Ejecutar la clase y ver los resultados:

```

Usando método toString: Uno [num1=30, num2=50]
El primer número es: 30
El segundo número es: 50
El resultado de la suma es: 1500
Usando método toString: Uno [num1=25, num2=150]

```

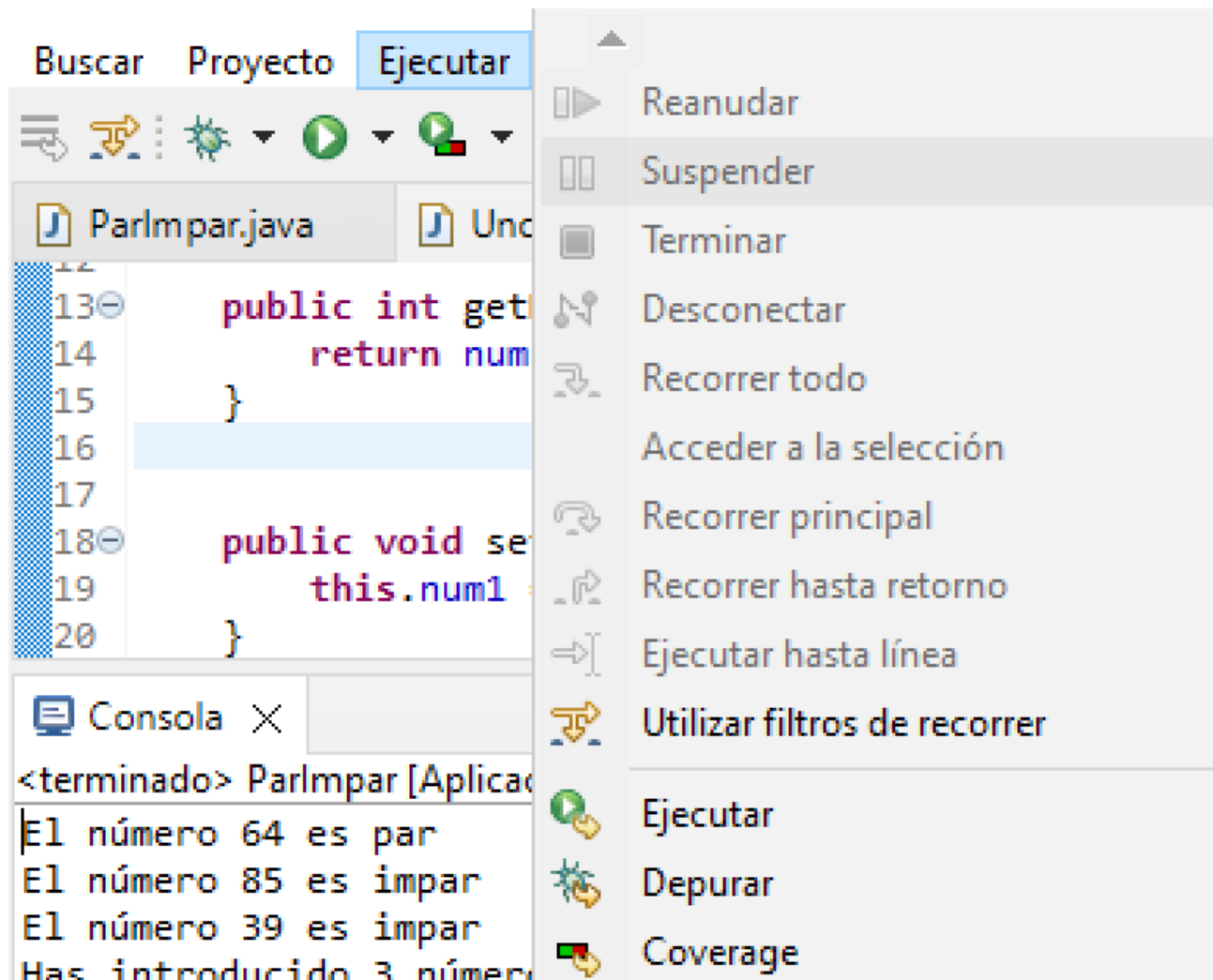

3. MENÚ EJECUTAR



PARAR Y QUITAR EJECUCIONES TERMINADAS

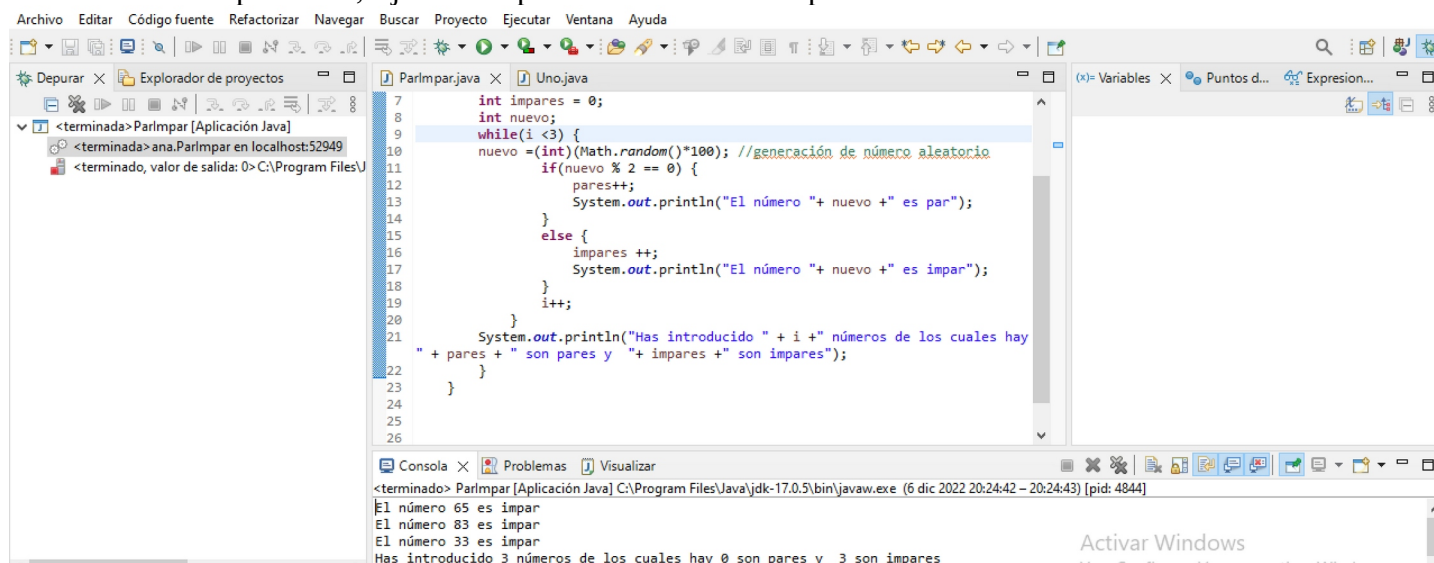


EJECUTAR UNA CLASE



DEPURAR UNA CLASE

Añadir la vista depuración , Ejecutar depuración o icono de depuración



EJECUTAR COBERTURA DE CLASES

Seleccionar la clase o el proyecto y menu ejecutar, ejecutar coverage

The screenshot shows an IDE with the following components:

- Explorador de paquetes:** A tree view on the left showing the project structure. The 'ana' package is expanded, showing 'Calculadora.java', 'ParImpar.java', 'Suma.java', and 'Uno.java'. The 'Pruebas' package is also expanded, showing 'CalculadoraTest.java', 'ParImparTest.java', and 'SumaTest.java'.
- ParImpar.java:** The main editor showing the source code of the 'ParImpar' class. The code is as follows:

```
1 package ana;
2 public class ParImpar {
3     public static void main(String[] args) {
4         // TODO Apéndice de método generado automáticamente
5         int i = 0;
6         int pares = 0;
7         int impares = 0;
8         int nuevo;
9         while(i < 3) {
10             nuevo = (int)(Math.random()*100); //generación de número aleatorio
11             if(nuevo % 2 == 0) {
12                 pares++;
13                 System.out.println("El número "+ nuevo +" es par");
14             }
15             else {
16                 impares ++;
17                 System.out.println("El número "+ nuevo +" es impar");
18             }
19         }
20     }
21 }
```
- Coverage:** A table showing the coverage of the code. The table has the following columns: 'Element', 'Coverage', 'Covered Instructio...', 'Missed Instructions', and 'Total Instructions'. The data is as follows:

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
> Uno.java	0,0 %	0	107	107
> Suma.java	0,0 %	0	23	23
> ParImpar.java	77,9 %	53	15	68
> Calculadora.java	0,0 %	0	11	11
> Pruebas	0,0 %	0	37	37

QUITAR LA EJECUCIÓN DE LA COBERTURA

