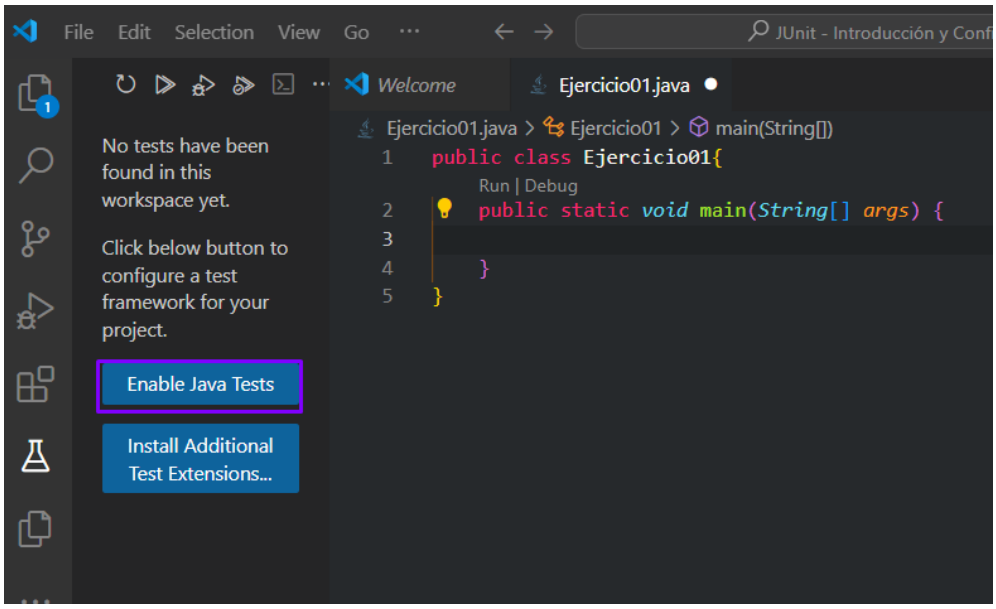
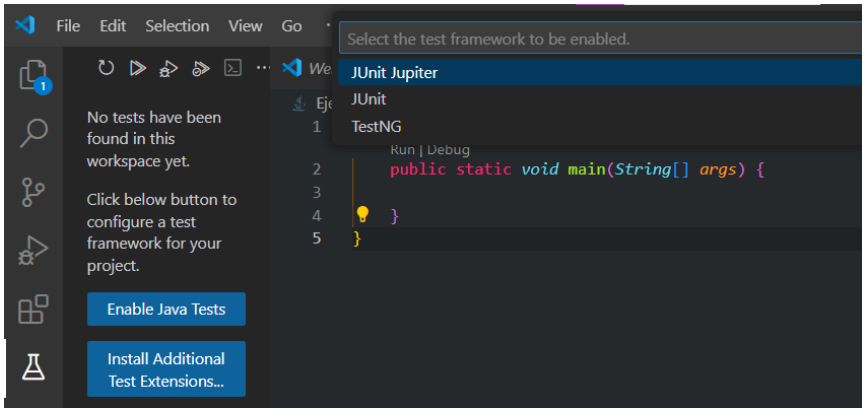
Configuración y primeros pasos en JUnit

1. Creamos una clase en Java y esperamos a que aparezca el botón en la parte izquierda de Visual llamado “Testing”, hacemos clic en él y luego hacemos clic en el botón “Enable Java Tests”.



Recordar: Tener instalado la Extensión Pack for Java

2. Luego seleccionamos la opción “JUnit Jupiter”



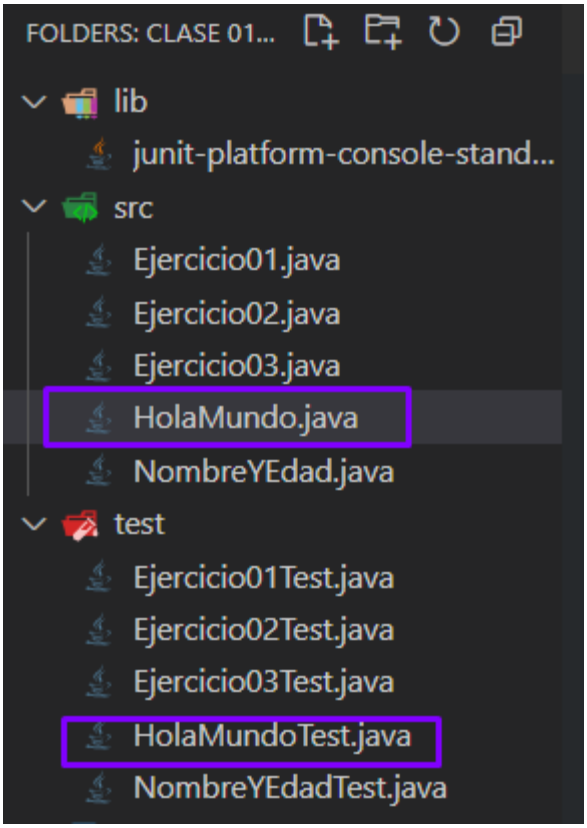
3. Esperamos que se descargue la librería en un archivo.jar

Primeros Ejercicios.

➖ Actividad: Hola mundo

- Crea un método en tu clase Main que imprima "Hola Mundo" en la consola.
- Crea una clase test para testear este método.

Estructura de ejercicios (separación de carpetas)



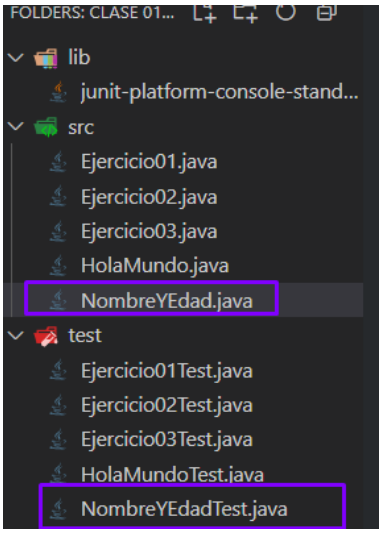
```java
package src;

public class HolaMundo {
  // Método que imprime "Hola Mundo" en la consola
    public static void main(String[] args) {
        System.out.println("Hola Mundo");
    }

}
```

```java
package test;

//Se realizan las diferentes importaciones
 import static org.junit.jupiter.api.Assertions.*
import static org.junit.jupiter.api.Assertions.assertEquals;

import java.io.ByteArrayOutputStream;
import java.io.PrintStream;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

import src.HolaMundo;

public class HolaMundoTest {

    private final PrintStream standarOut = System.out; // Guarda la referencia al PrintStream original (System.out) para restaurarlo más tarde

    // Crea un ByteArrayOutputStream para capturar la salida de la consola
    private final ByteArrayOutputStream outputStreamCaptor = new ByteArrayOutputStream();

    @BeforeEach // Redirige la salida estándar al ByteArrayOutputStream antes de cada prueba.
    public void setUp(){
```

```
        System.setOut(new PrintStream(outputStreamCaptor));
    }

    @AfterEach // Restaura la salida estándar original después de cada
prueba.
    public void tearDown(){
       System.setOut(standarOut);
    }

    @Test  // Indica que este método es una prueba unitaria

    void testMain() {
        HolaMundo.main(new String[1]); // Ejecuta el método main del
programa
        String salida = outputStreamCaptor.toString();// Captura la salida
del sistema.
        // Compara el primer argumento (esperado) con el segundo argumento
(salida) y lanza una excepción si no son iguales
        assertEquals("Hola Mundo\r\n", salida);
    }
}
```

➖ Actividad: Imprimir nombre y edad

- Crea un método en tu clase Main que reciba por parámetro el nombre y la edad, y luego imprima el mensaje "Me llamo [nombre] y tengo [edad] años". No debes hacer uso de la clase Scanner. En su lugar, simplemente crea el método correspondiente y reemplaza [nombre] y [edad] con tus datos almacenados previamente en una variable, para comparar con la salida del output.
- Crea una clase para testear esta actividad.



```java
package src;
public class NombreYEdad {
    public static void main(String[] args) {
        String nombre = "Fatima";
        Integer edad = 26;
        System.out.println("Me llamo " + nombre + " y tengo " + edad + "
años");


    }
}
```

```java
package test;

import static org.junit.jupiter.api.Assertions.assertEquals;

import java.io.ByteArrayOutputStream;
import java.io.PrintStream;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

import src.NombreYEdad;

public class NombreYEdadTest {
    // Configuración de la captura de salida del sistema
    private final PrintStream standarOut = System.out; // Guarda la salida
estándar original.
    // Inicializa un ByteArrayOutputStream para capturar la salida.
    private final ByteArrayOutputStream outputStreamCaptor = new
ByteArrayOutputStream();

    @BeforeEach // Redirige la salida estándar al ByteArrayOutputStream
antes de cada prueba.
    public void setUp(){
        System.setOut(new PrintStream(outputStreamCaptor));
    }

    @AfterEach // Restaura la salida estándar original después de cada
prueba.
    public void tearDown(){
        System.setOut(standarOut);
    }
```

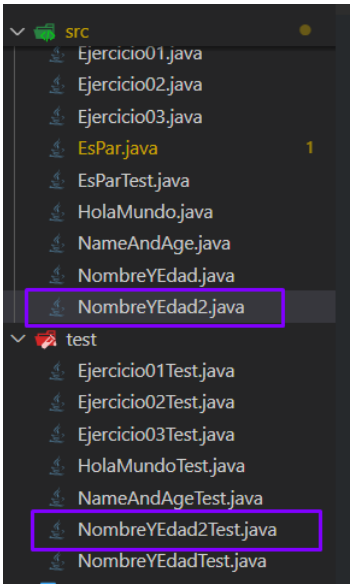
```java
    @Test
    void testMain() {
        NombreYEdad.main(new String[1]);// Ejecuta el método main del
programa
        String salida = outputStreamCaptor.toString();// Captura la salida
del sistema.
        // Compara el primer argumento (esperado) con el segundo argumento
(salida) y lanza una excepción si no son iguales
        assertEquals("Me llamo Fatima y tengo 26 años\r\n", salida);


    }
}
```

Ejercicios con pruebas de entradas de datos

➖ Actividad: Imprimir nombre y edad 2.0

Dado lo aprendido en la actividad Nombre y Edad, procede a resolver la siguiente actividad contemplando las siguientes mejoras:

- Crea dos métodos para obtener el nombre y la edad, respectivamente. Ambos métodos deben recibir por parámetro una variable de tipo "Scanner".
- Crea una clase para testear esta actividad, incluyendo cada uno de los métodos.



```java
package src;
import java.util.Scanner;

public class NombreYEdad2 {
    // Método para obtener el nombre desde el Scanner
    public String getNombre(Scanner scanner) {
        System.out.print("Ingrese su nombre: "); // Imprime un mensaje solicitando el nombre
        return scanner.nextLine(); // Lee y retorna el nombre ingresado por el usuario
    }

    // Método para obtener la edad desde el Scanner
    public int getEdad(Scanner scanner) {
        System.out.print("Ingrese su edad: "); // Imprime un mensaje solicitando la edad
        return scanner.nextInt(); // Lee y retorna la edad ingresada por el usuario
    }

    // Método que imprime el nombre y la edad en la consola
    public void printNombreYEdad(String nombre, int edad) {
        System.out.println("Me llamo " + nombre + " y tengo " + edad + " años"); // Imprime el mensaje con el nombre y la edad
    }


}
```

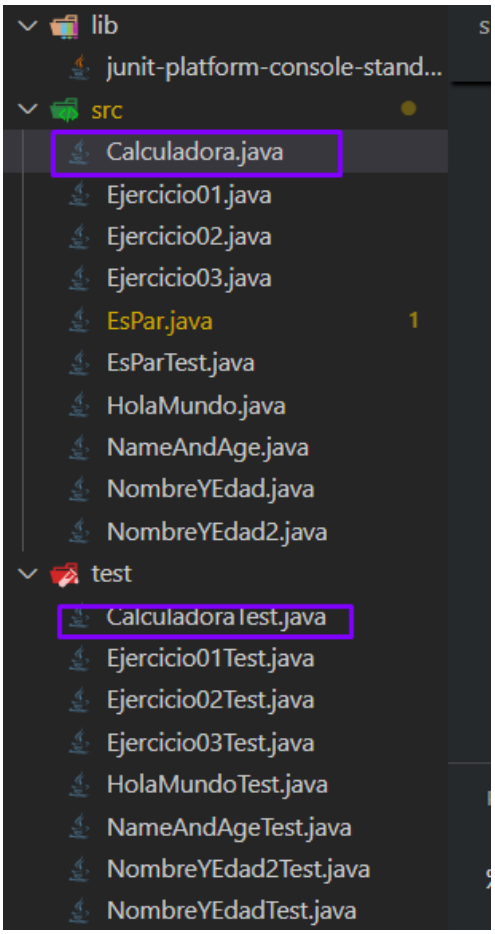
```java
package test;

import static org.junit.jupiter.api.Assertions.assertEquals;

import java.io.ByteArrayOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

import src.NombreYEdad2;
```

```java
public class NombreYEdad2Test {
    // Crear un ByteArrayOutputStream para capturar la salida de la consola
    // outContent: Captura cualquier salida de texto enviada a la consola
(System.out).
    private final ByteArrayOutputStream outContent = new
ByteArrayOutputStream();
    // Guardar la referencia al PrintStream original (System.out) para
restaurarlo
    // originalOut: Guarda el valor original de System.out para restaurarlo
después de la prueba.
    private final PrintStream originalOut = System.out;
    // Creamos una instancia de la clase que se está probando
(NombreYEdadTest2).
    private NombreYEdad2 main;

    @BeforeEach
    public void setUp() {
        // Redirigir la salida de la consola al ByteArrayOutputStream
outContent
        System.setOut(new PrintStream(outContent));
        // Inicializa una nueva instancia de NombreYEdadTest2.
        main = new NombreYEdad2();
    }

    @AfterEach
    public void tearDown() {
        // Restaurar la salida de la consola al PrintStream original
        System.setOut(originalOut);
    }

    // Verificamos que el método getNombre() devuelve el nombre correcto
ingresado por el usuario.

    @Test
    public void testGetNombre() {
        // Se simula una entrada de consola para ingresar nombre
        String input = "Fatima";
        Scanner scanner = new Scanner(input);
        // Se invoca al método getNombre(scanner) de la clase
NombreYEdadTest2.
        String nombre = main.getNombre(scanner);
        // Se compara el resultado devuelto con "Fatima" usando
assertEquals.
        assertEquals("Fatima", nombre);
    }
```

```
    // Verificamos que el método getEdad() devuelve la edad correcta
ingresada por el usuario.
    @Test
    public void testGetEdad() {
        String input = "26";
        Scanner scanner = new Scanner(input);
        int edad = main.getEdad(scanner);
        assertEquals(26, edad);
    }

    // Verificamos que el método printNombreYEdad() imprime correctamente el
nombre y la edad en la consol
    @Test
    public void testPrintNombreYEdad() {
        // Se asignan valores de prueba: nombre = "Fatima" y edad = 26.
        String nombre = "Fatima";
        int edad = 26;
        // Se llama al método printNombreYEdad(nombre, edad).
        main.printNombreYEdad(nombre, edad);
        // La salida de la consola capturada en outContent se compara con la
cadena esperada:
        assertEquals("Me llamo Fatima y tengo 26 años" +
System.lineSeparator(), outContent.toString());
    }

}
```

Actividad Calculadora

- Considerando la lógica para simular una calculadora con las operaciones básicas, crea un método llamado "**menu()**" que no reciba ningún parámetro y se encargue de generar la lógica de mostrar el menú de opciones y producir el bucle para seguir mostrando el menú hasta seleccionar la opción de "salir". Solicita dos números al usuario y realiza la operación matemática seleccionada, teniendo en cuenta las validaciones necesarias como la división por cero.
- Crea una clase para testear esta actividad, incluyendo cada uno de los métodos.




```java
package src;

import java.util.InputMismatchException;
import java.util.Scanner;

public class Calculadora {
    // Se crea un objeto Scanner que permite capturar datos desde la
consola.
    private Scanner scanner = new Scanner(System.in);

    public void menu() {
        int opcion;
        do {
            System.out.println("\n--- CALCULADORA ---");
            System.out.println("1. Sumar");
            System.out.println("2. Restar");
            System.out.println("3. Multiplicar");
            System.out.println("4. Dividir");
```

```
            System.out.println("5. Salir");
            System.out.print("Seleccione una opción: ");

            try {
                opcion = scanner.nextInt();  //captura la opción ingresada.
                if (opcion >= 1 && opcion <= 4) {
                    realizarOperacion(opcion);
                // Si la opción es 5, finaliza el programa.
                } else if (opcion == 5) {
                    System.out.println("¡Gracias por usar la calculadora!");
                } else {
                    System.out.println("Opción no válida. Intente de
nuevo.");
                }
            // Si la entrada no es un número entero, se captura la excepción
            // Captura las entradas del usuario, manejando errores con
InputMismatchException.
            } catch (InputMismatchException e) {
                System.out.println("Error: Debe ingresar un número
válido.");
                scanner.next(); // Limpiar el buffer del scanner
                opcion = 0; // Reiniciar opción para continuar el bucle
            }
        } while (opcion != 5);
    }

    private void realizarOperacion(int opcion) {
        // Solicita dos números usando el método solicitarNumero().
        double num1 = solicitarNumero("Ingrese el primer número: ");
        double num2 = solicitarNumero("Ingrese el segundo número: ");

        switch (opcion) {
            case 1 -> System.out.println("Resultado: " + sumar(num1, num2));
            case 2 -> System.out.println("Resultado: " + restar(num1,
num2));
            case 3 -> System.out.println("Resultado: " + multiplicar(num1,
num2));
            case 4 -> {
                // La división verifica si el segundo número es cero antes
de realizar la operación.
                if (num2 != 0) {
                    System.out.println("Resultado: " + dividir(num1, num2));
                } else {
                    System.out.println("Error: No se puede dividir entre
cero.");
```

```
                }
            }
        }
    }

    private double solicitarNumero(String mensaje) {
        // Se usa solicitarNumero() para asegurarse de que el usuario
ingrese números válidos.
        double numero = 0;
        boolean valido = false;
        do {
            try {
                System.out.print(mensaje);
                numero = scanner.nextDouble();
                valido = true;
            } catch (InputMismatchException e) {
                System.out.println("Error: Debe ingresar un número
válido.");
                scanner.next(); // Limpiar el buffer
            }
        } while (!valido);
        return numero;
    }

    public double sumar(double a, double b) {
        return a + b;
    }

    public double restar(double a, double b) {
        return a - b;
    }

    public double multiplicar(double a, double b) {
        return a * b;
    }

    public double dividir(double a, double b) {
        if (b == 0) throw new ArithmeticException("División por cero no
permitida");
        return a / b;
    }

    public static void main(String[] args) {
        Calculadora calc = new Calculadora();
        calc.menu();
```

```
    }
}
```

```
package test;

import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertThrows;

import org.junit.jupiter.api.Test;

import src.Calculadora;

public class CalculadoraTest {

    @Test
    // Comprobar que el método sumar() de la clase Calculadora funciona
correctamente.
    void testSumar() {
        // Se crea una instancia de la clase Calculadora llamada calc.
        Calculadora calc = new Calculadora();
        // Comprueba que el resultado real de calc.sumar() coincida con el
resultado esperado:
        assertEquals(5.0, calc.sumar(2, 3));
        // La prueba debe pasar si el método sumar() devuelve los resultados
correctos
        assertEquals(0.0, calc.sumar(2, -2));
    }

    @Test
    void testRestar() {
        Calculadora calc = new Calculadora();
        assertEquals(1.0, calc.restar(3, 2));
        assertEquals(-4.0, calc.restar(2, 6));
    }

    @Test
    void testMultiplicar() {
        Calculadora calc = new Calculadora();
        assertEquals(6.0, calc.multiplicar(2, 3));
        assertEquals(0.0, calc.multiplicar(2, 0));
    }

    @Test
```

```
    void testDividir() {
        Calculadora calc = new Calculadora();
        assertEquals(2.0, calc.dividir(6, 3));
        // Comprueba que llamar a calc.dividir(1, 0) lanza una excepción del
tipo ArithmeticException.
        assertThrows(ArithmeticException.class, () -> calc.dividir(1, 0));
    }
}
```