



PROGRAMACIÓN II

Trabajo Práctico 1: Introducción a Java

OBJETIVO GENERAL

Aplicar los conocimientos adquiridos sobre la instalación y configuración del entorno de desarrollo, manipulación de datos, operadores matemáticos y depuración de código en Java, mediante ejercicios prácticos introductorios.

MARCO TEÓRICO

Concepto	Aplicación en el proyecto
Instalación y entorno	Almacenan el conjunto de países
Variables y tipos de datos	Representan los datos de cada país (nombre, población, superficie, etc.)
Entrada y salida	Separan las operaciones: carga, búsqueda, estadísticas, ordenamientos
Operadores aritméticos	Aplican filtros y validaciones según criterios
Caracteres especiales	Permite ordenar países por población, nombre, superficie, etc.
Expresiones e instrucciones	Permiten obtener indicadores clave del dataset
Tipos de datos y conversiones	Lectura del dataset desde un archivo CSV
Debugging y errores comunes	Identificación y corrección de errores de compilación.
Pruebas de escritorio	Análisis paso a paso de ejecución de código.

CASO PRÁCTICO

El trabajo consiste en resolver una serie de ejercicios introductorios en Java que permitan:

- Configurar correctamente el entorno de desarrollo (Java JDK y NetBeans).
- Crear programas básicos que imprimen mensajes en consola.
- Declarar variables de distintos tipos y manipular sus valores.
- Leer datos ingresados por el usuario usando `Scanner`.
- Realizar operaciones aritméticas básicas.
- Aplicar caracteres de escape para dar formato a la salida.
- Analizar diferencias entre expresiones e instrucciones.
- Detectar y corregir errores simples en el código.
- Comprender el comportamiento del lenguaje mediante pruebas de escritorio.

CONCLUSIONES ESPERADAS

- Reforzar los conceptos fundamentales del lenguaje Java.
- Familiarizarse con la estructura básica de un programa en Java.
- Aprender a depurar errores comunes.
- Comprender la importancia de las conversiones de tipo y expresiones.
- Adquirir habilidades prácticas para manipular entradas/salidas y variables.
- Aplicar el uso de herramientas como NetBeans y prácticas de depuración.

1. Verificar que tienes instalado Java JDK y NetBeans

- Confirma que tienes Java JDK instalado ejecutando el siguiente comando en la terminal: `java -version`
- Abre NetBeans, crea un nuevo proyecto y configura el modo oscuro.
- Toma una captura de pantalla del entorno configurado y agrégala a tu entrega.

a. Java JDK instalado:

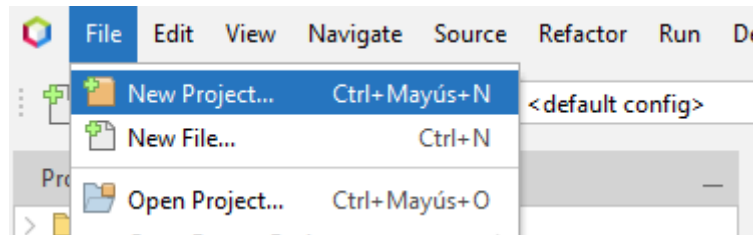
```

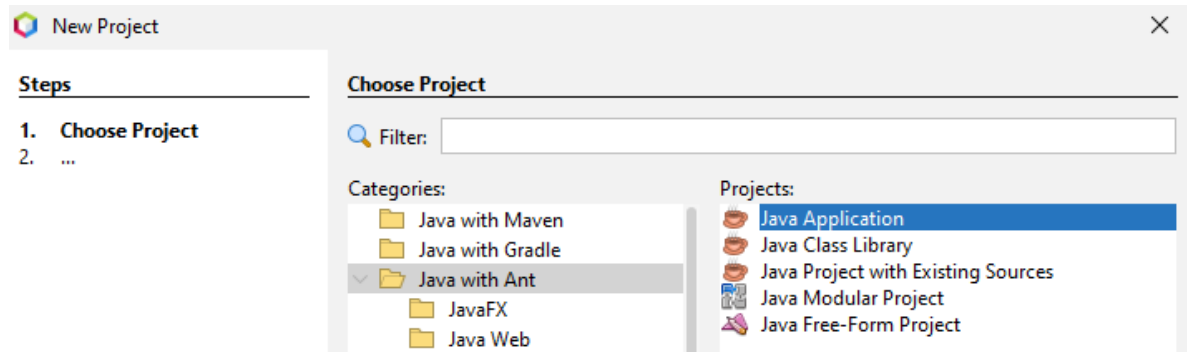
C:\> Símbolo del sistema
Microsoft Windows [Versión 10.0.19045.6216]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\veron>java -version
openjdk version "11.0.16.1" 2022-08-12 LTS
OpenJDK Runtime Environment Microsoft-40648 (build 11.0.16.1+1-LTS)
OpenJDK 64-Bit Server VM Microsoft-40648 (build 11.0.16.1+1-LTS, mixed mode)

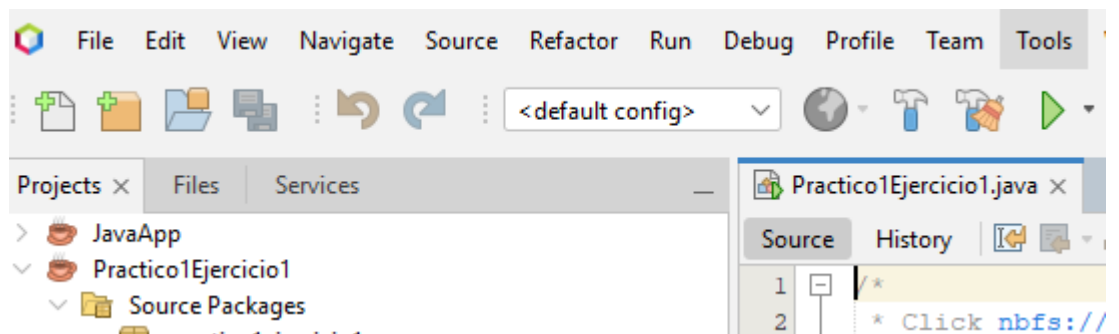
C:\Users\veron>
  
```

- Abre NetBeans,
 - crea un nuevo proyecto

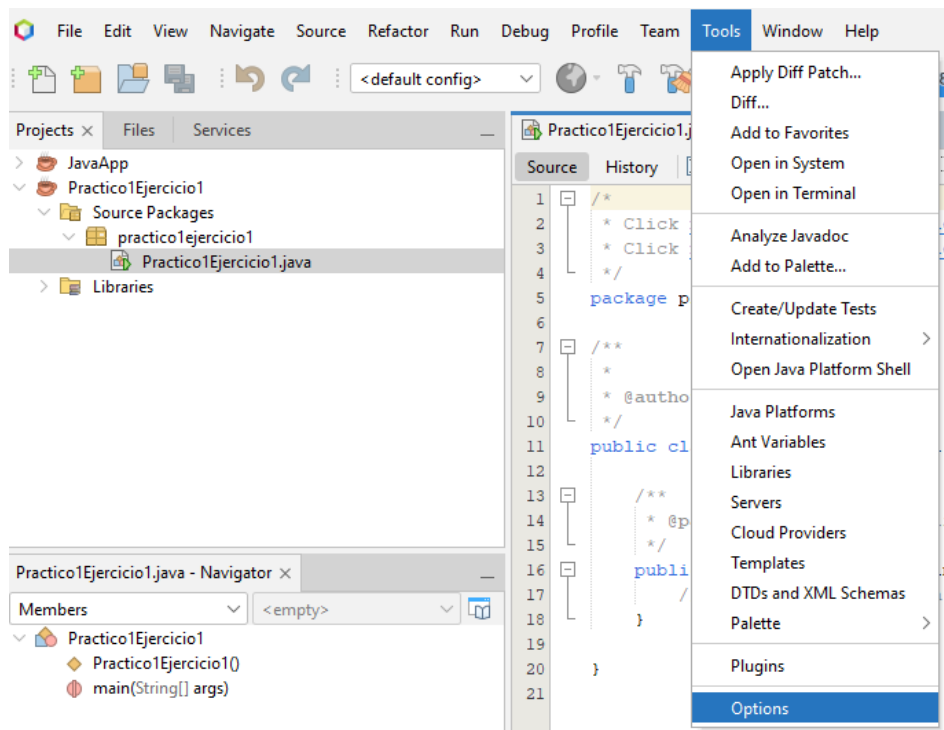




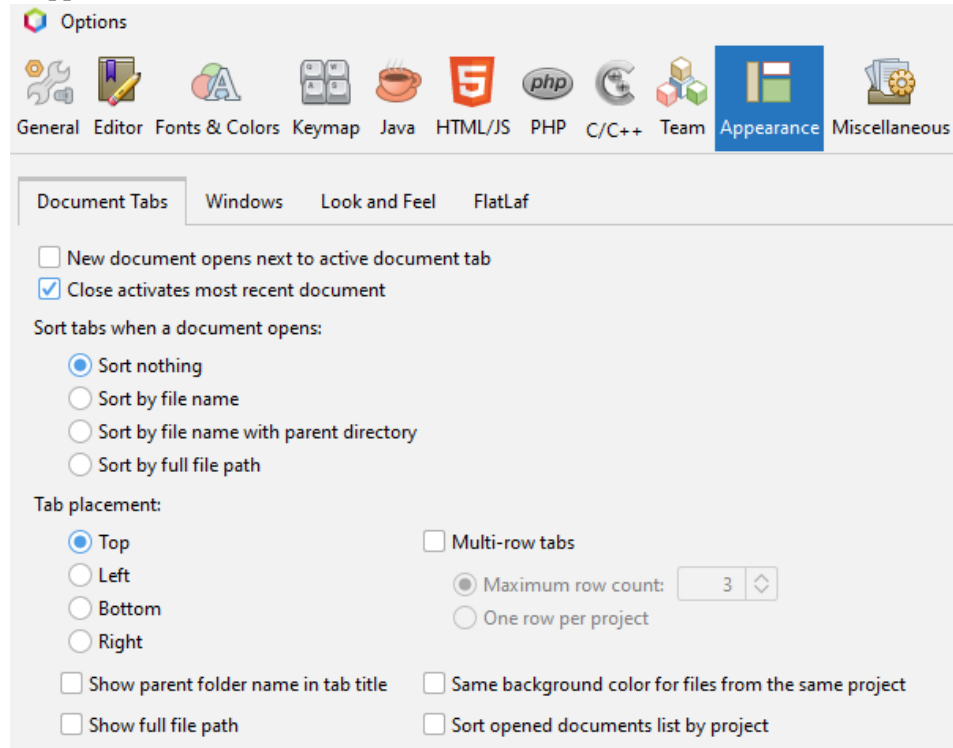
2. configura el modo oscuro>Seleccionamos **Tools**>



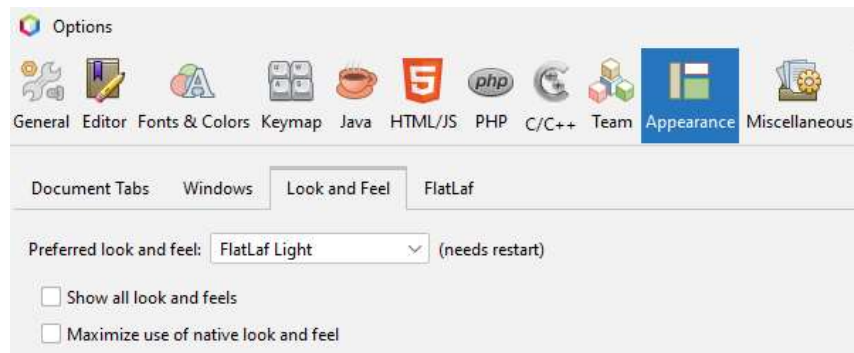
>Options



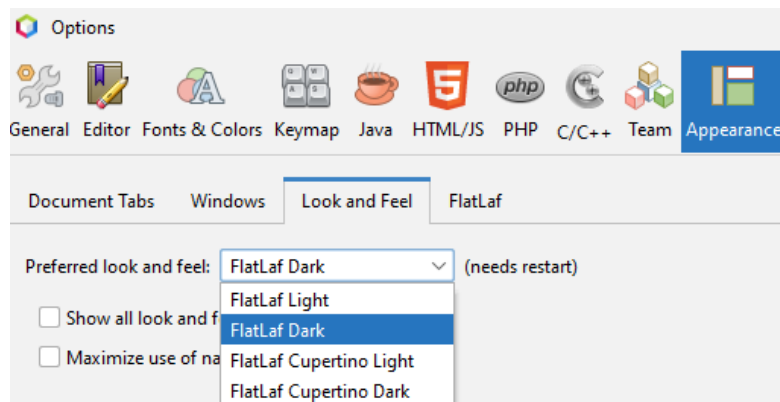
>Appearance



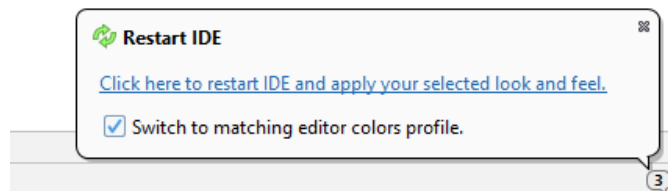
>Look and Feel



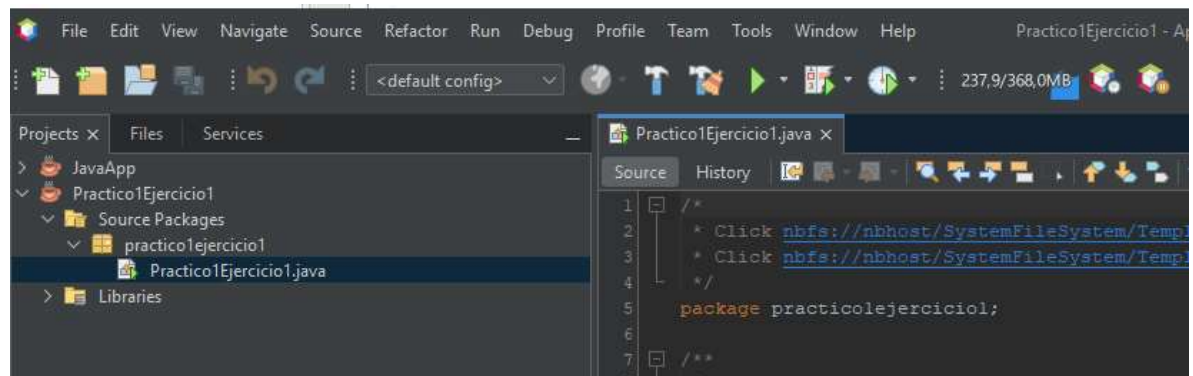
>Dark



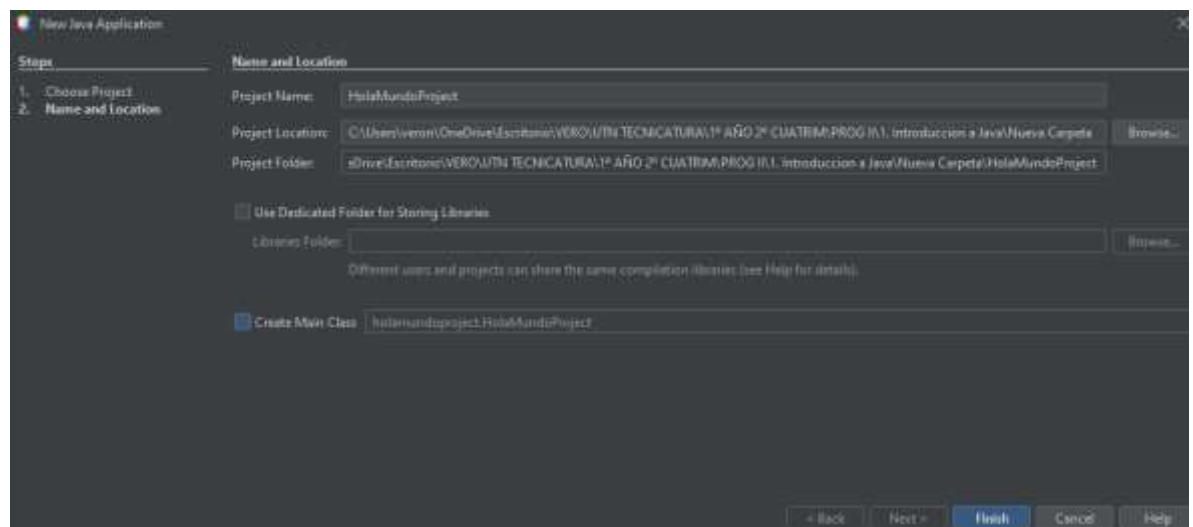
>Restart



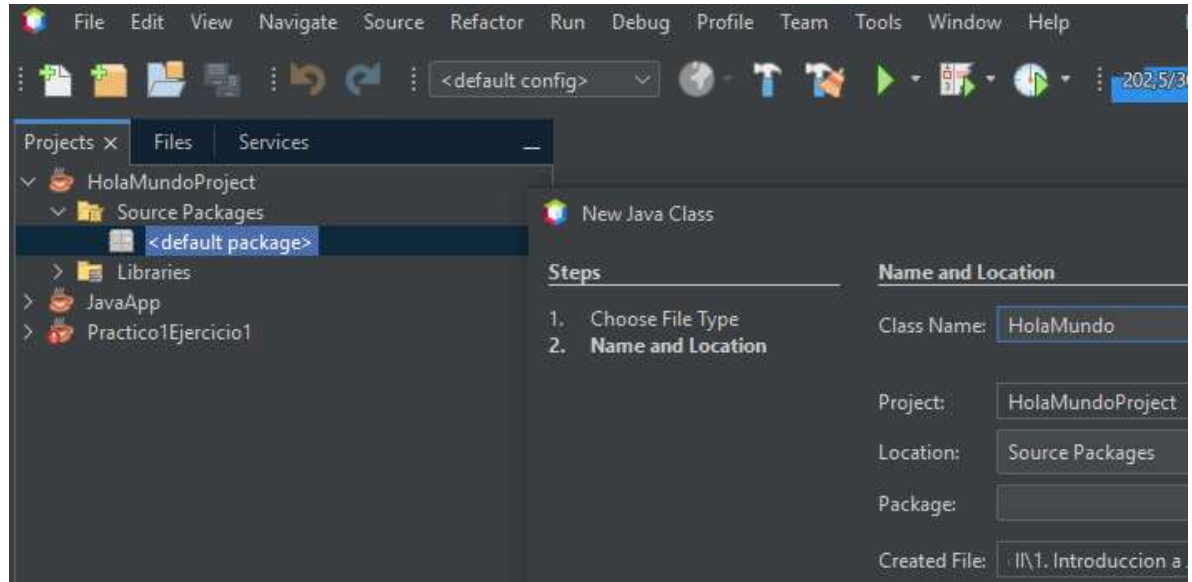
>Modo Oscuro



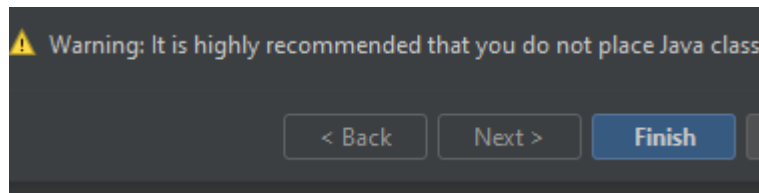
2. Escribir y ejecutar un programa básico en Java.
 - a. Creá una clase llamada **HolaMundo**.
 - b. Escribe un programa que imprima el mensaje: **¡Hola, Java!**
 - c. Ejecuta el programa en NetBeans y adjunta una captura del resultado en la consola.
- a. Crear la clase
1. Crear Proyecto



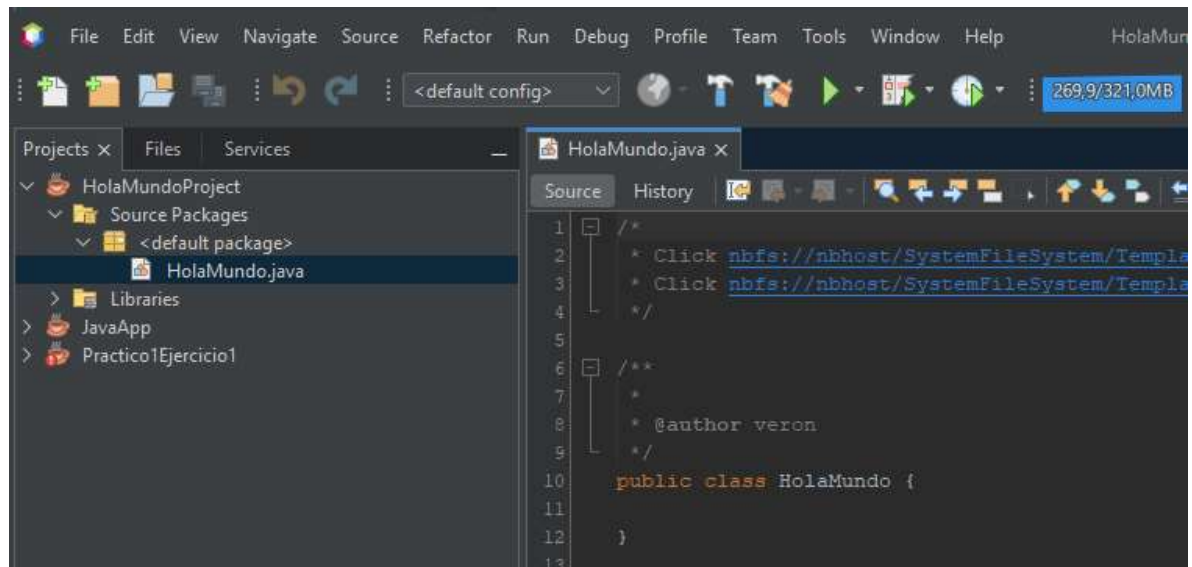
2. Crear la clase HolaMundo



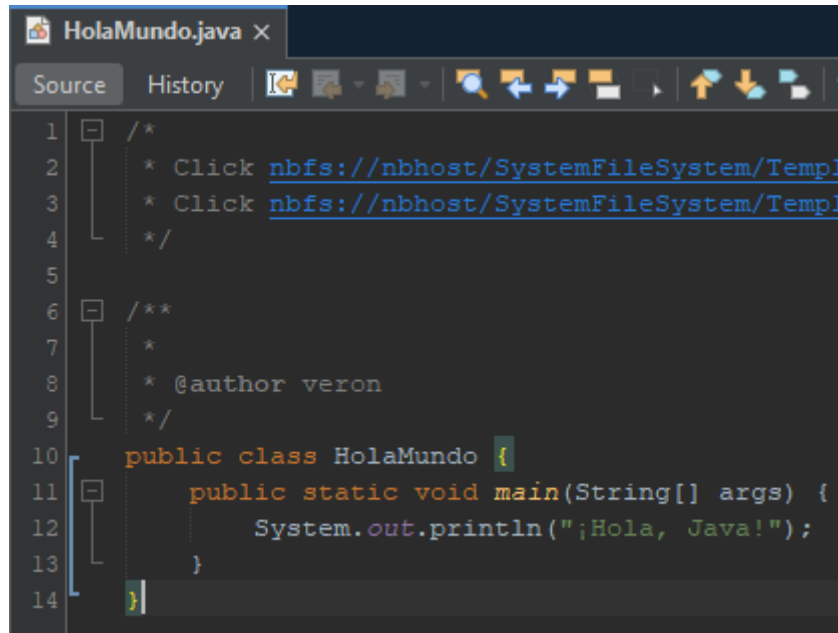
>Finish



>class HolaMundo

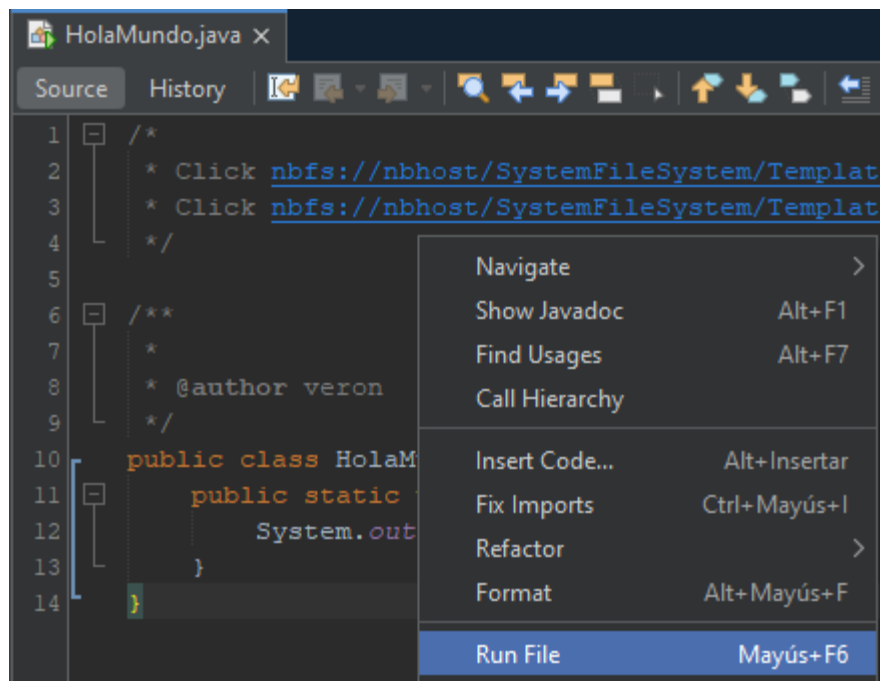


- b. Escribe un programa que imprima el mensaje: ¡Hola, Java!



```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Comment
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Comment
4   */
5
6  /**
7   *
8   * @author veron
9   */
10 public class HolaMundo {
11     public static void main(String[] args) {
12         System.out.println("¡Hola, Java!");
13     }
14 }
```

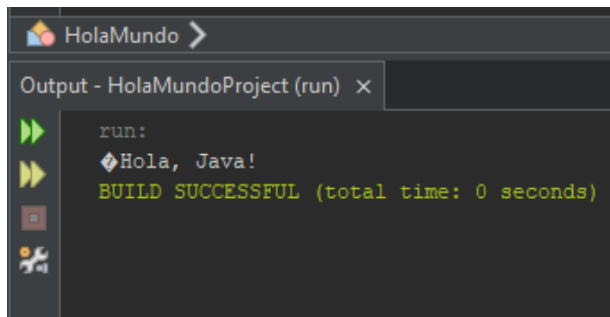
- c. Ejecuta el programa en NetBeans y adjunta una captura del resultado en la consola.



```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Comment
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Comment
4   */
5
6  /**
7   *
8   * @author veron
9   */
10 public class HolaM
11     public static
12         System.out
13     }
14 }
```

Context Menu Options	
Navigate	>
Show Javadoc	Alt+F1
Find Usages	Alt+F7
Call Hierarchy	
Insert Code...	Alt+Insertar
Fix Imports	Ctrl+Mayús+I
Refactor	>
Format	Alt+Mayús+F
Run File	Mayús+F6

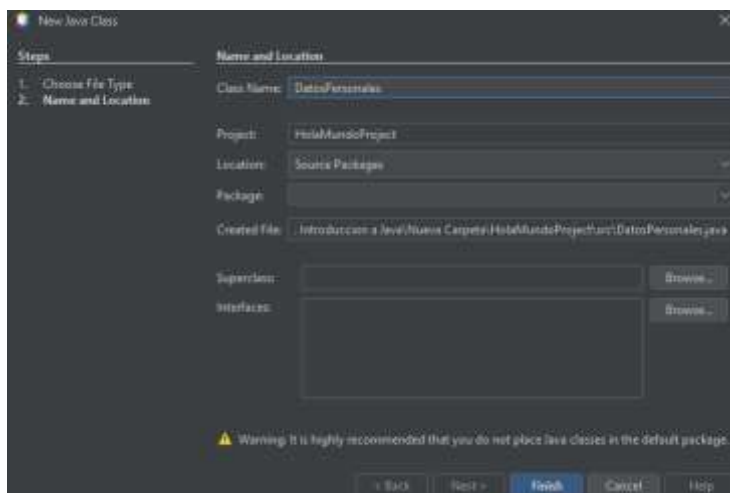
> ¡Hola, Java!



3. Crea un programa que declare las siguientes variables con valores asignados:
 - a. String nombre
 - b. int edad
 - c. double altura
 - d. boolean estudiante

Imprime los valores en pantalla usando `System.out.println()`.

>Crear la clase:



>Crea un programa que declare las siguientes variables con valores asignados:

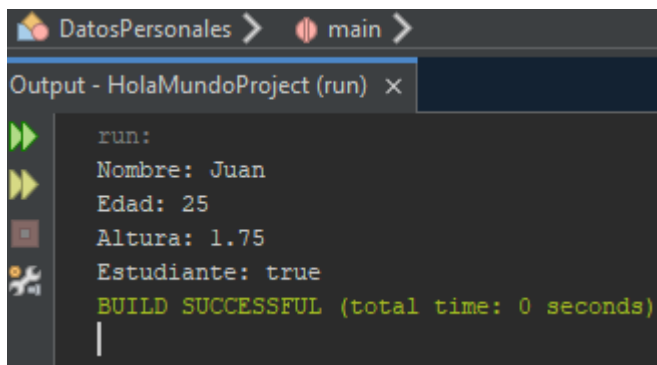
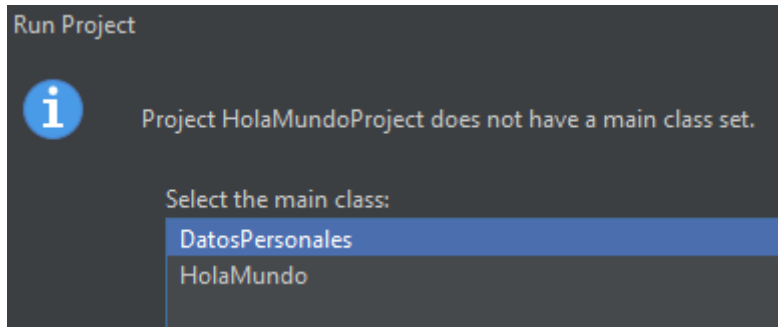
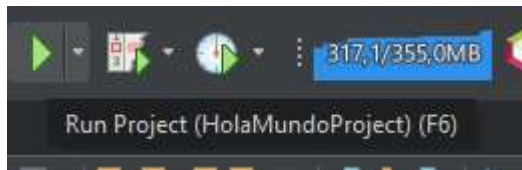
```

 * @author veron
 */
public class DatosPersonales {
    public static void main(String[] args) {
        // Declaración e inicialización de variables
        String nombre = "Juan"; // nombre
        int edad = 25;           // edad
        double altura = 1.76;    // altura en metros
        boolean estudiante = true; // se cambia true a false si no es estudiante

        // Imprimir los valores en pantalla
        System.out.println("Nombre: " + nombre);
        System.out.println("Edad: " + edad);
        System.out.println("Altura: " + altura);
        System.out.println("Estudiante: " + estudiante);
    }
}

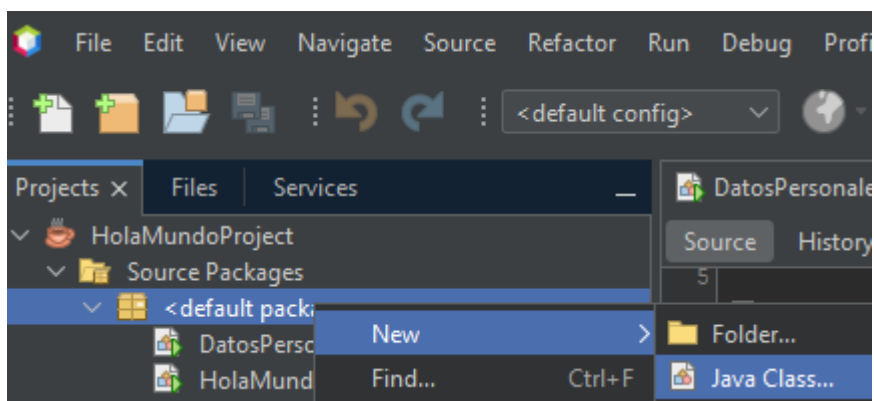
```

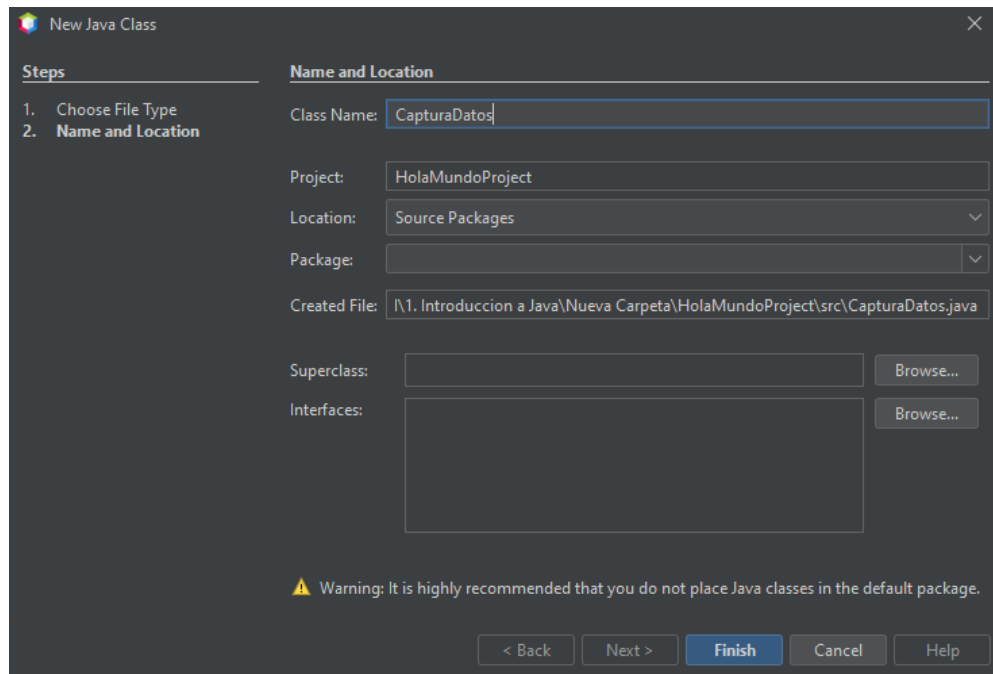

>Imprime los valores en pantalla usando System.out.println().



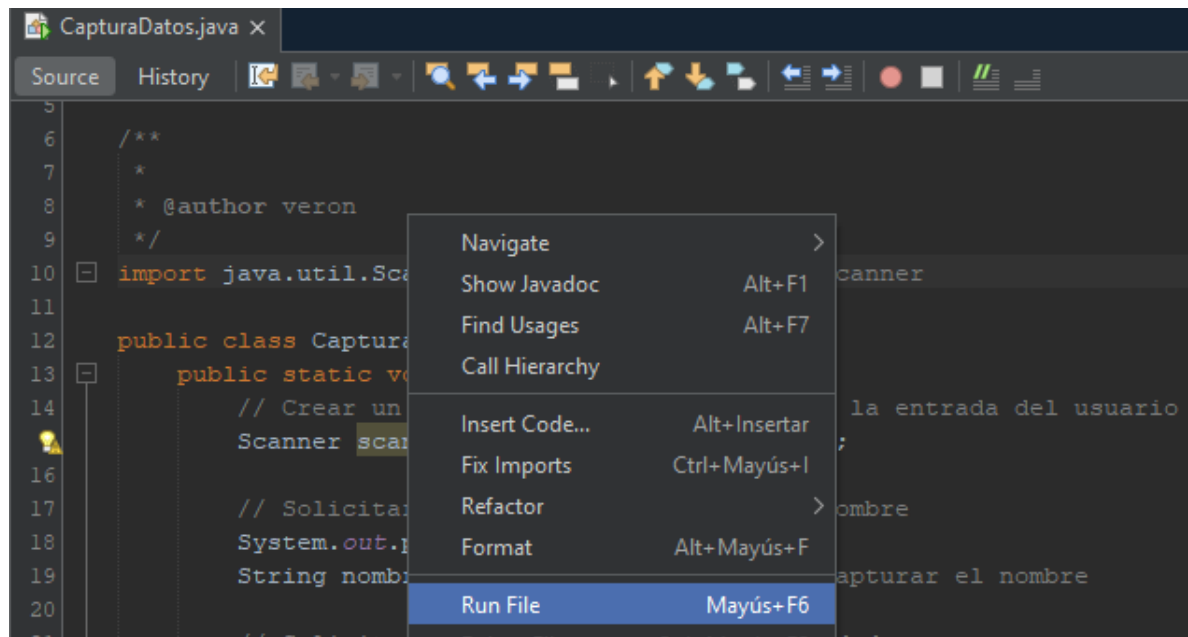
4. Escribe un programa que solicite al usuario ingresar su nombre y edad, y luego los muestre en pantalla. Usa [Scanner](#) para capturar los datos.

>Creamos la class

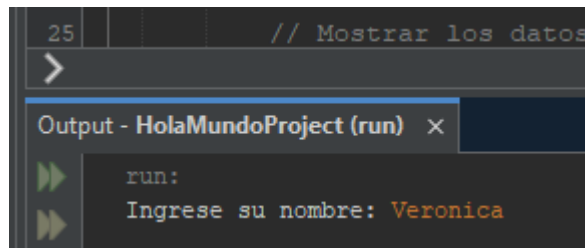




>Ejecutar el programa



>Ingresa su Nombre

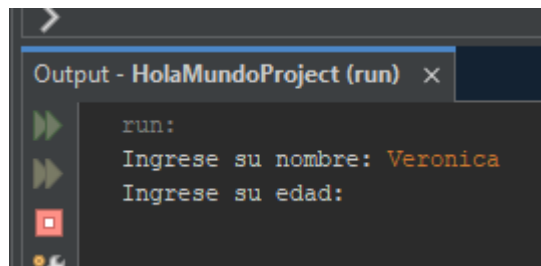


```
25 // Mostrar los datos
```

Output - HolaMundoProject (run) x

```
run:
Ingreso su nombre: Veronica
```

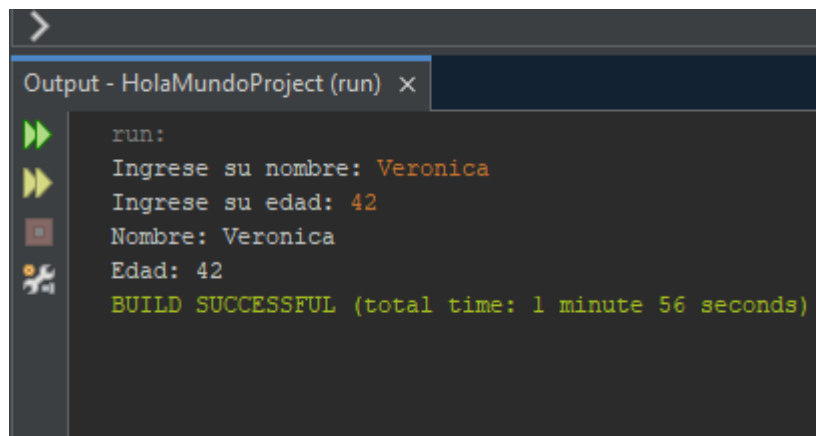
>Ingrese su Edad



Output - HolaMundoProject (run) x

```
run:
Ingreso su nombre: Veronica
Ingreso su edad:
```

>Salida



Output - HolaMundoProject (run) x

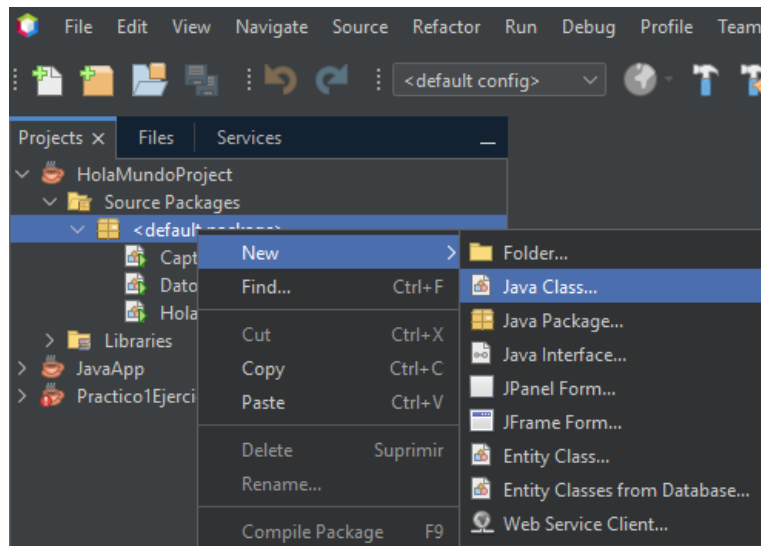
```
run:
Ingreso su nombre: Veronica
Ingreso su edad: 42
Nombre: Veronica
Edad: 42
BUILD SUCCESSFUL (total time: 1 minute 56 seconds)
```

5. Escribe un programa que solicite dos números enteros y realice las siguientes operaciones:

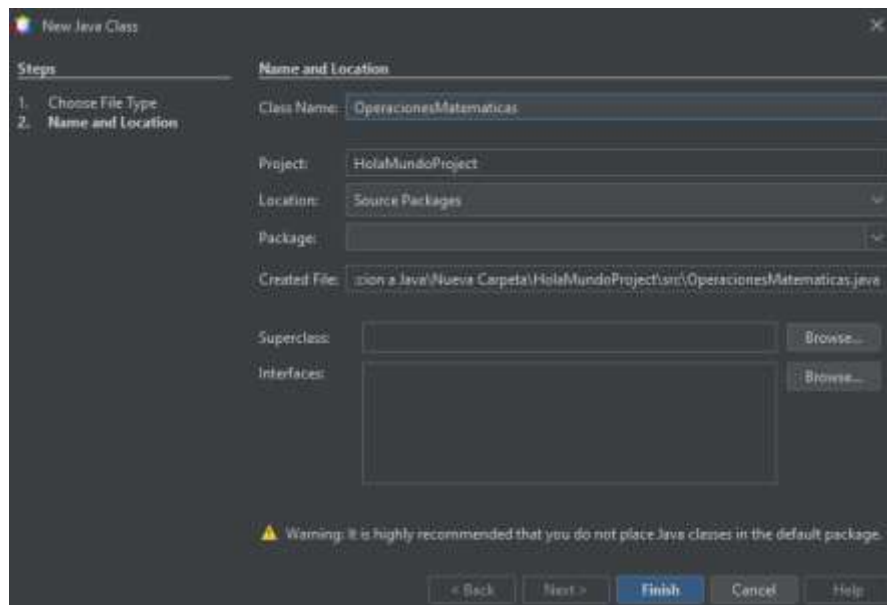
- a. Suma
- b. Resta
- c. Multiplicación
- d. División

Muestra los resultados en la consola.

>Crear la class



>class OperacionesMatematicas



>programa

```

    * @author veron
    */
import java.util.Scanner; // Importar la clase Scanner

public class OperacionesMatematicas {
    public static void main(String[] args) {
        // Crear un objeto Scanner para capturar la entrada del usuario
        Scanner scanner = new Scanner(System.in);

        // Solicitar al usuario que ingrese el primer número
        System.out.print("Ingrese el primer número entero: ");
        int numero1 = scanner.nextInt(); // Capturar el primer número

        // Solicitar al usuario que ingrese el segundo número
        System.out.print("Ingrese el segundo número entero: ");
        int numero2 = scanner.nextInt(); // Capturar el segundo número

        // Realizar las operaciones
        int suma = numero1 + numero2;
        int resta = numero1 - numero2;
        int multiplicacion = numero1 * numero2;

        // Para la división, asegurarse de que el segundo número no sea cero
        double division;
        if (numero2 != 0) {
            division = (double) numero1 / numero2; // double para obtener un resultado más preciso
        } else {
            division = 0; // Evitar división por cero
            System.out.println("Error: División por cero no es permitida.");
        }
    }
}

```

```

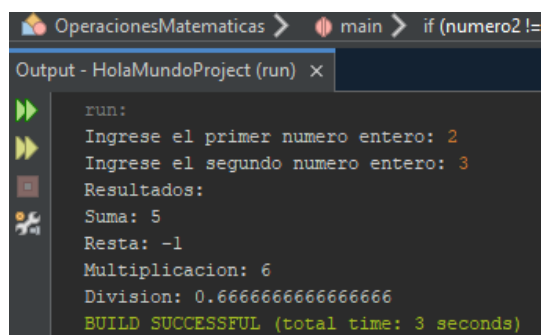
// Mostrar los resultados en la consola
System.out.println("Resultados:");
System.out.println("Suma: " + suma);
System.out.println("Resta: " + resta);
System.out.println("Multiplicación: " + multiplicacion);

// Solo mostrar la división si el segundo número no es cero
if (numero2 != 0) {
    System.out.println("División: " + division);
}

// Cerrar el objeto Scanner
scanner.close();
}

```

>Muestra los resultados en la consola



```

run:
Ingrese el primer numero entero: 2
Ingrese el segundo numero entero: 3
Resultados:
Suma: 5
Resta: -1
Multiplicacion: 6
Division: 0.6666666666666666
BUILD SUCCESSFUL (total time: 3 seconds)

```

6. Escribe un programa que muestre el siguiente mensaje en consola:

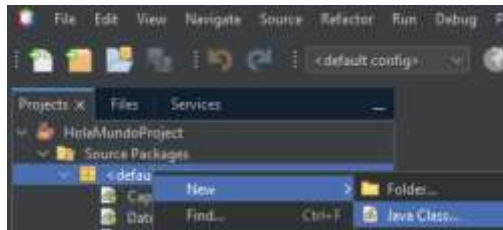
Nombre: Juan Pérez

Edad: 30 años

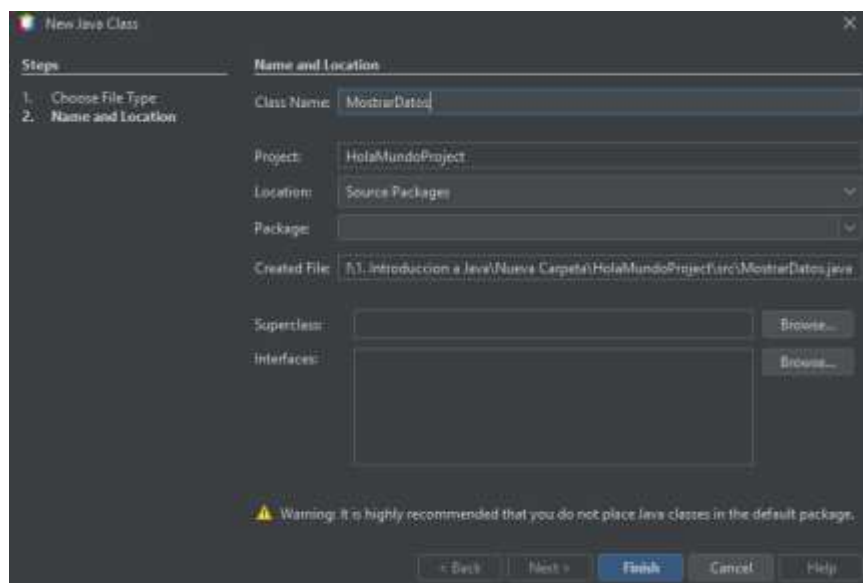
Dirección: "Calle Falsa 123"

Usa caracteres de escape (\n, \") en `System.out.println()`.

>Crear la class



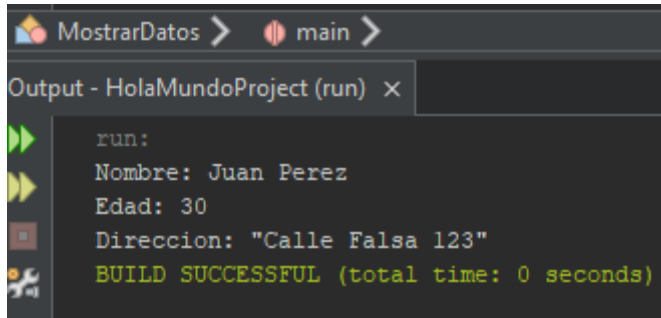
>class MostrarDatos



>Usa caracteres de escape (\n, \") en `System.out.println()`.

```
* @author veron
*/
public class MostrarDatos {
    public static void main(String[] args) {
        // Usar caracteres de escape para mostrar el mensaje en consola
        System.out.println("Nombre: Juan Perez\n" +
                           "Edad: 30\n" +
                           "Direccion: \"Calle Falsa 123\"");
    }
}
```

>Salida



```
run:
Nombre: Juan Perez
Edad: 30
Direccion: "Calle Falsa 123"
BUILD SUCCESSFUL (total time: 0 seconds)
```

7. Analiza el siguiente código y responde: ¿Cuáles son expresiones y cuáles son instrucciones? Explica la diferencia en un breve párrafo.

```
int x = 10; // Línea 1
x = x + 5; // Línea 2
System.out.println(x); // Línea 3
```

Podemos distinguir entre expresiones e instrucciones:

int x = 10 => **Instrucción** de declaración que define una variable x de tipo entero y la inicializa con el valor 10.

x = x + 5 => **Instrucción** de asignación.

La parte **x + 5** es una **expresión** que se evalúa para producir un valor que luego se asigna a la variable x.

System.out.println(x) => **Instrucción** que invoca un método (println) para imprimir el valor de x.

Diferencia entre expresiones e instrucciones:

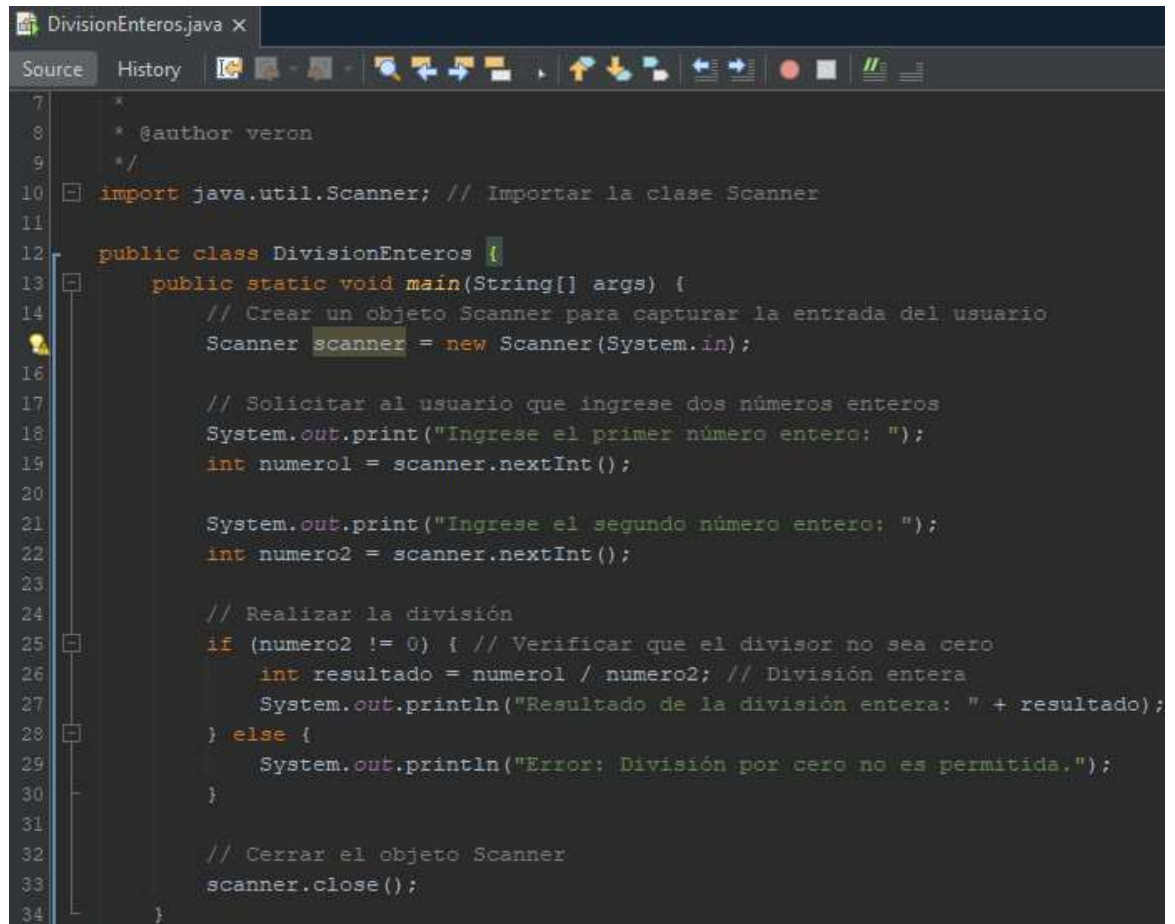
Una **expresión** es una combinación de variables, operadores y valores que se evalúan para producir un nuevo valor. Por ejemplo, en la línea 2, **x + 5** es una **expresión** que evalúa y devuelve 15.

Una **instrucción** es una línea de código que realiza una acción completa, como declarar una variable, asignarle un valor, o llamar a un método, como en las líneas 1 y 3.

En resumen, las **expresiones** son componentes de las **instrucciones**, y las **instrucciones** son las que realizan las acciones en un programa.

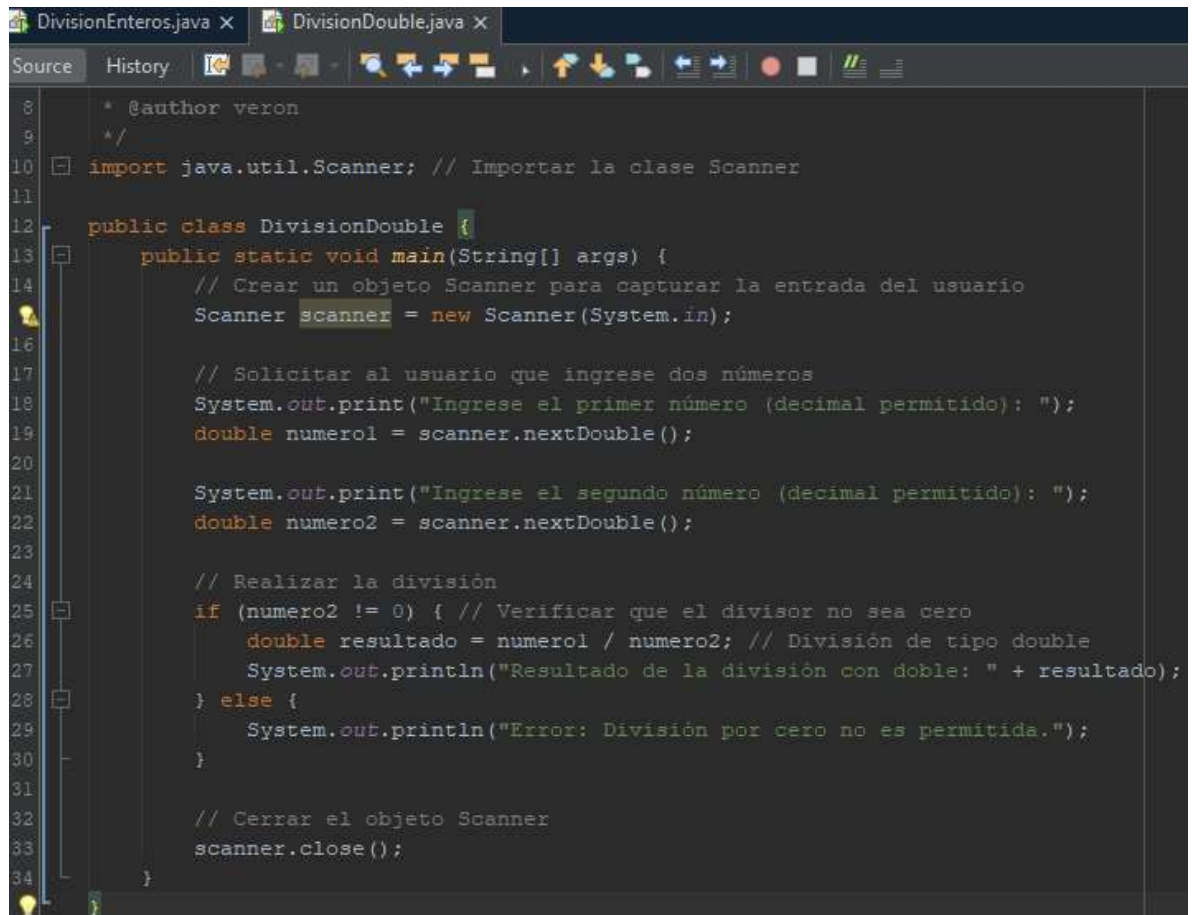
8. Manejar conversiones de tipo y división en Java.
- Escribe un programa que divida dos números enteros ingresados por el usuario.
 - Modifica el código para usar `double` en lugar de `int` y compara los resultados.

- a. programa que divida dos números enteros ingresados por el usuario.



```
7      *
8      * @author veron
9      */
10     import java.util.Scanner; // Importar la clase Scanner
11
12     public class DivisionEnteros {
13     public static void main(String[] args) {
14         // Crear un objeto Scanner para capturar la entrada del usuario
15         Scanner scanner = new Scanner(System.in);
16
17         // Solicitar al usuario que ingrese dos números enteros
18         System.out.print("Ingrese el primer número entero: ");
19         int numero1 = scanner.nextInt();
20
21         System.out.print("Ingrese el segundo número entero: ");
22         int numero2 = scanner.nextInt();
23
24         // Realizar la división
25         if (numero2 != 0) { // Verificar que el divisor no sea cero
26             int resultado = numero1 / numero2; // División entera
27             System.out.println("Resultado de la división entera: " + resultado);
28         } else {
29             System.out.println("Error: División por cero no es permitida.");
30         }
31
32         // Cerrar el objeto Scanner
33         scanner.close();
34     }
```

- b. Modifica el código para usar `double` en lugar de `int` y compara los resultados.



```
8      * @author veron
9      */
10     import java.util.Scanner; // Importar la clase Scanner
11
12     public class DivisionDouble {
13     public static void main(String[] args) {
14         // Crear un objeto Scanner para capturar la entrada del usuario
15         Scanner scanner = new Scanner(System.in);
16
17         // Solicitar al usuario que ingrese dos números
18         System.out.print("Ingrese el primer número (decimal permitido): ");
19         double numero1 = scanner.nextDouble();
20
21         System.out.print("Ingrese el segundo número (decimal permitido): ");
22         double numero2 = scanner.nextDouble();
23
24         // Realizar la división
25         if (numero2 != 0) { // Verificar que el divisor no sea cero
26             double resultado = numero1 / numero2; // División de tipo double
27             System.out.println("Resultado de la división con doble: " + resultado);
28         } else {
29             System.out.println("Error: División por cero no es permitida.");
30         }
31
32         // Cerrar el objeto Scanner
33         scanner.close();
34     }
```

Comparación de Resultados:

- **División Entera:** si se ingresa 5 y 2, el resultado será 2, ya que se realiza una división entera y se descarta la parte decimal.
- **División con double:** si se ingresa 5.0 y 2.0, el resultado será 2.5, ya que double permite el manejo de números con decimales y no descarta la parte decimal.

9. Corrige el siguiente código para que funcione correctamente. Explica qué error tenía y cómo lo solucionaste.

```
import java.util.Scanner;

public class ErrorEjemplo {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Ingresa tu nombre: ");

        String nombre = scanner.nextInt(); // ERROR

        System.out.println("Hola, " + nombre);

    }

}
```

>Corrección: se utiliza nextLine() en lugar de nextInt()

The screenshot shows an IDE window titled 'ErrorEjemplo.java'. The code is as follows:

```
7  *
8  * @author veron
9  */
10 import java.util.Scanner;
11
12 public class ErrorEjemplo {
13     public static void main(String[] args) {
14         Scanner scanner = new Scanner(System.in);
15         System.out.print("Ingresa tu nombre: ");
16         String nombre = scanner.nextLine(); // Corrección: se utiliza nextLine() en lugar de nextInt()
17         System.out.println("Hola, " + nombre);
18
19         // Cerrar el scanner
20         scanner.close();
21     }
22 }
```

Below the code editor, the 'Output' window shows the successful execution of the program:

```
run:
Ingresa tu nombre: Veronica
Hola, Veronica
BUILD SUCCESSFUL (total time: 4 seconds)
```

>Explica qué error tenía:

El error en el código original se encuentra en esta línea:

`String nombre = scanner.nextInt(); // ERROR`

Uso incorrecto del método:

El método `nextInt()` de la clase `Scanner` se utiliza para leer un número entero desde la entrada estándar. Sin embargo, se está intentando asignar este valor a una

variable de tipo String, lo cual provoca un error de tipo de datos porque nextInt() devuelve un int y no un String.

Lectura de entrada incorrecta:

Para capturar un nombre (que es una cadena de texto), se debe utilizar el método nextLine(), que lee toda la línea de entrada del usuario hasta el salto de línea, devolviendo un String.

>y cómo lo solucionaste:

Se cambió scanner.nextInt() a scanner.nextLine().

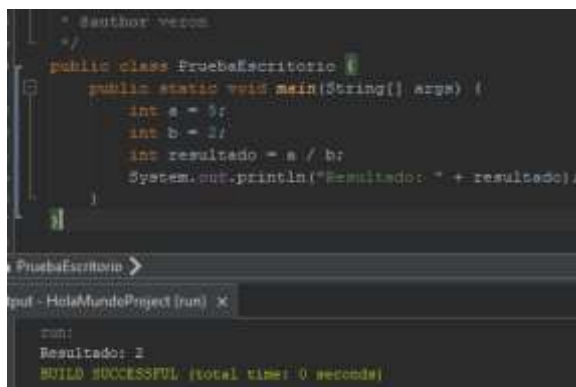
Esto permite que el programa lea el nombre del usuario correctamente como una cadena de texto, y luego se puede utilizar en el saludo.

Además, por buenas prácticas, se agregó una línea para cerrar el objeto Scanner al final del programa con scanner.close(), lo que es importante para liberar los recursos del sistema.

10. Completa la tabla de prueba de escritorio para el siguiente código. ¿Cuál es el valor de resultado y por qué?

```
public class PruebaEscritorio {
    public static void main(String[] args) {
        int a = 5;
        int b = 2;
        int resultado = a / b;
        System.out.println("Resultado: " + resultado);
    }
}
```

>Codigo:



```

/*
 * @author veron
 */
public class PruebaEscritorio {
    public static void main(String[] args) {
        int a = 5;
        int b = 2;
        int resultado = a / b;
        System.out.println("Resultado: " + resultado);
    }
}

PruebaEscritorio >
run - HolaMundoProject [run] X
run:
Resultado: 2
BUILD SUCCESSFUL (total time: 0 seconds)

```

> Tabla de prueba de escritorio:

Paso	Variable	Valor	Descripción
1	a	5	Se asigna el valor 5 a la variable a.
2	b	2	Se asigna el valor 2 a la variable b.
3	resultado	2	Se calcula $5 / 2$ (división entera), obteniendo 2.
4	Imprimir	“Resultado: 2”	Se imprime el resultado en la consola.

> **Valor de resultado:**

El valor de resultado es 2. ¿Porque?

- En Java, cuando se realiza una división entre dos enteros (a y b son de tipo int), se lleva a cabo una **división entera**, es decir que cualquier parte decimal de la división se descarta.
- Al calcular $5 / 2$, el resultado es 2.5, pero al hacer la división de enteros, Java solo retiene la parte entera, que es 2.

Así que resultado tendrá el valor 2 y la impresión en consola será:

Resultado: 2