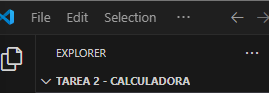
**Tarea 2**

**Crear en Visual Studio Code (u otro editor) un proyecto en Java (u otro lenguaje), para simular una calculadora.**

1. **El proyecto debe contener una clase principal Calculadora**

Antes de nada, creamos la carpeta de la Tarea 2

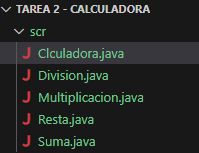


Dentro de esa carpeta, crea otra carpeta llamada scr (en donde pondré mis códigos)

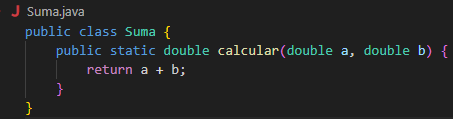


1. **La clase Calculadora debe implementar al menos estas operaciones: Suma, Resta, multiplicación y división.**

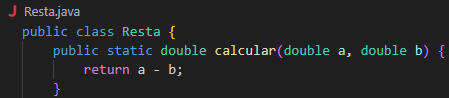
Una vez dentro creamos la calculadora, la suma, la resta la división y la multiplicación.



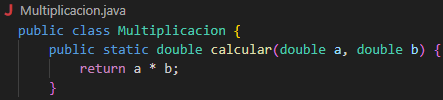
El código de la Suma:



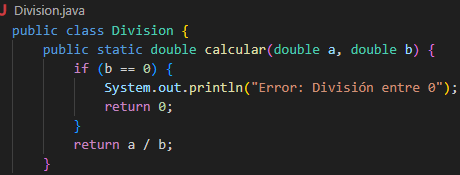
El código de la Resta:



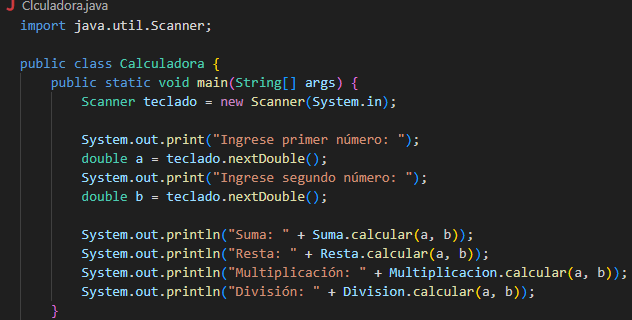
El código de la Multiplicación:



El código de la División:



El código de la Calculadora:



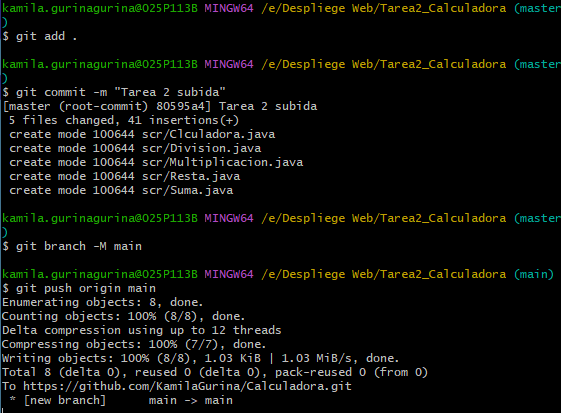
1. **Cada operación deberá realizarse en una clase diferente. Corresponderá a una funcionalidad, y por tanto un rama en GIT. Tendrá su propio archivo en el proyecto y en el repositorio.**

Iniciamos el repositorio



Enlazamos con repositorio remoto

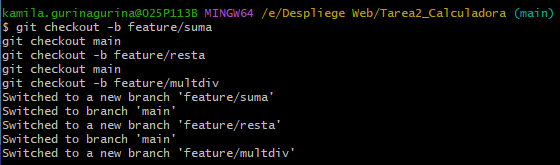


Se creó el proyecto base de la calculadora en Java con la clase principal Calculadora y las clases de las operaciones básicas (Suma, Resta, Multiplicacion, Division). Se inicializó el repositorio Git local, se hicieron los commits de los archivos y se subió la rama principal (main) al repositorio remoto en GitHub.

1. **En la máquina local habrá 3 carpetas, que corresponderán a la simulación de 3 Programadores (A, B y C). Reparte el trabajo entre los 3. Demuestra mediante los commits que han trabajado los 3 en el proyecto. Una idea es repartir el trabajo por funcionalidades.**

Crear ramas por funcionalidad

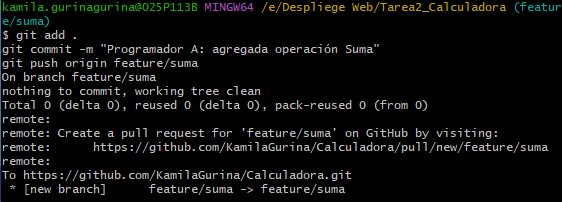
* feature/suma → Programador A
* feature/resta → Programador B
* feature/multdiv → Programador C

****

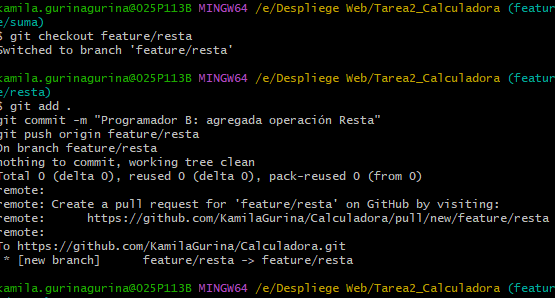
1. **Cada persona diferente deberá modificar su funcionalidad y la clase Calculadora si fuese necesario. Subir sus cambios al mismo repositorio remoto de GitHub**. Al final, deberán fusionar las ramas a la rama principal (master o main), simulando la integración del equipo.

*Rama suma* 🡪 Se creó y subió la rama feature/suma para la operación Suma, simulando el trabajo del Programador A, quedando lista para integrarse con la rama principal.

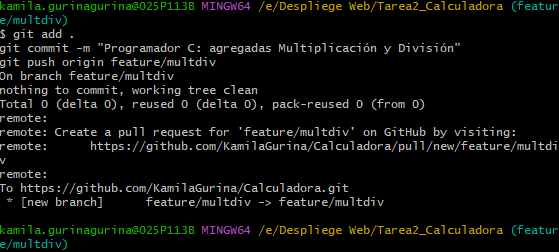




*Rama resta*

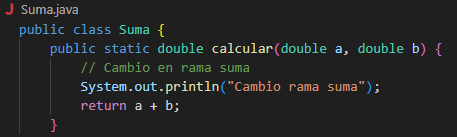


*Rama multiplicación* y división

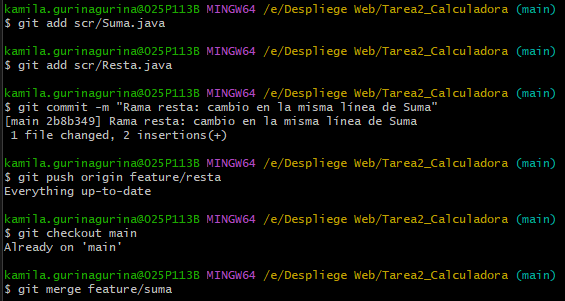


1. **Deberá simularse al menos un conflicto, y se demostrará que se ha hecho una resolución a mano del mismo.**

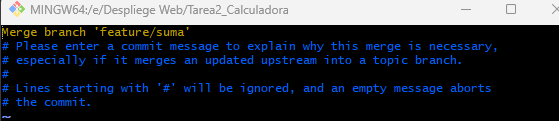
Para generar un conflicto modificamos la línea en el código Suma.java



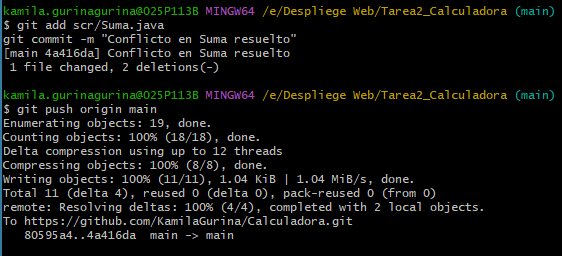
Volvemos a Git Bash y commiteamos los cambios



El conflicto se ha generado correctamente

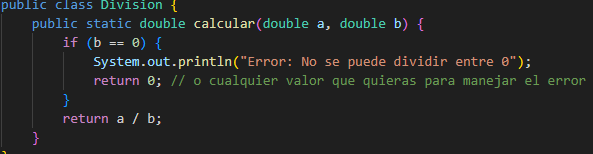


Una vez resuelto el conflicto los subimos (se me olvidó sacar foto del conflicto a resolver)

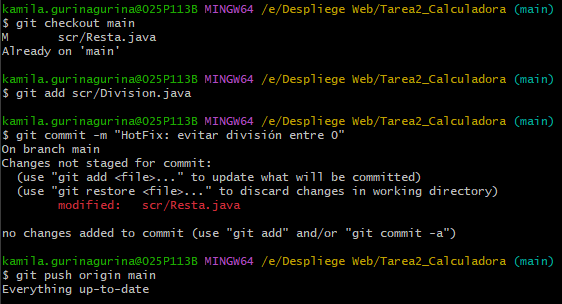


1. **Deberá simularse un HotFix en la rama master, por ejemplo una comprobación para que no se pueda dividir entre 0.**

Modificamos el método calcular asi:

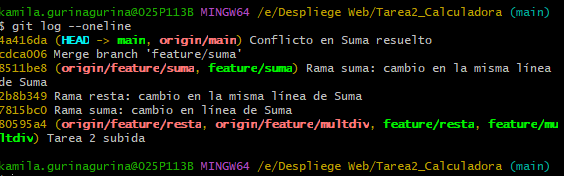


Y en el GitBash



1. **El control de commits debe reflejar los cambios realizados.**

Se muestra el historial de commits. Se observa la subida inicial del proyecto, los cambios realizados en cada rama, el merge de la rama suma y la resolución del conflicto manual en Suma.java



Conclusión:

Durante el desarrollo, el mayor reto fue comprender cómo generar y resolver conflictos en Git manualmente. También aprendí la importancia de los commits claros y del trabajo en ramas separadas.”