

INDICE

Enunciado	2
INTRODUCCIÓN	3
USUARIO-A	4
Clonado-A.....	4
Config	4
ERROR-SIMULADO.....	5
CREACION-MAIN	6
Código	6
USUARIO-B	7
Clonado-B.....	7
Config	7
MÉTODOS.....	7
COMMITTS	7
CODIGO	8
USUARIO-C	9
Clonado-C.....	9
Config	9
COMMIT	9
CODIGO	9
Master	10
HOTFIX.....	10
DIFICULTADES.....	11

Enunciado

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

- El proyecto debe contener una **clase principal Calculadora**.
- La clase **Calculadora** debe implementar al menos estas operaciones: Suma, Resta, multiplicación y división.
- 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.
- 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.
- Cada persona diferente deberá modificar su funcionalidad y la clase Calculadora si fuese necesario.
- Hacer commits como si fueran distintos usuarios (`git config user.name` y `git config user.email` por carpeta). 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.
- Deberá simularse al menos un conflicto, y se demostrará que se ha hecho una resolución a mano del mismo.
- Deberá simularse un HotFix en la rama master, por ejemplo una comprobación para que no se pueda dividir entre 0.
- El control de commits debe reflejar los cambios realizados.
- **ENTREGAR:** Un .pdf con capturas de todo el proceso. EL pdf debe tener una introducción con el IDE o editor usado y el lenguaje. Se debe añadir al final un párrafo en el que se expliquen dificultades encontradas y/o cuestiones a reseñar.
- El nombre del PDF deberá tener el siguiente formato:
Tarea2_Apellido1_Apellido2_Nombre.

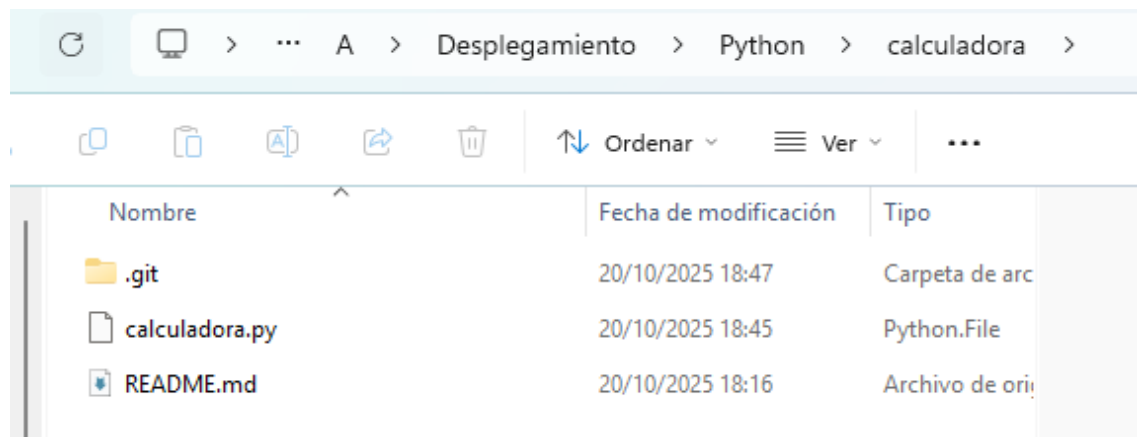
INTRODUCCIÓN

IDE: Visual Studio Code

Lenguaje: Python

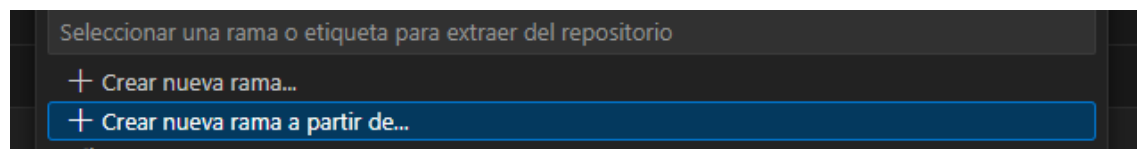
1-

Primero cree una carpeta para subir el main de la calculadora, rama develop y luego clonar el repositorio en las carpetas A,B, y C.

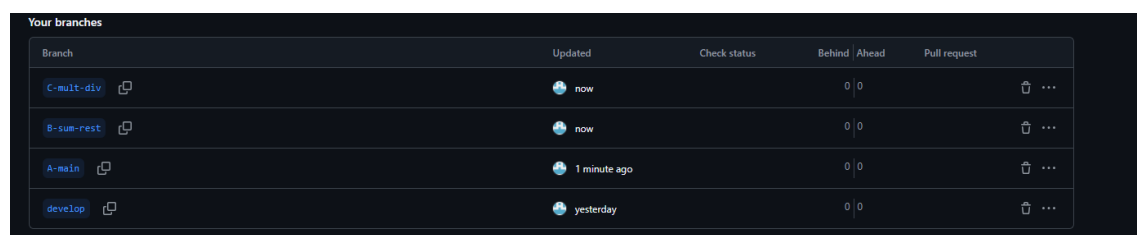


2-

Creé desde visual Studio las ramas antes de hacer "pull" en cada directorio.



Elegir develop y escribir el nombre de la rama



USUARIO-A

Los pasos de B y C se ejecutaron antes que A

Clonado-A

```
PS C:\Users\marcos.fernandezgarc\A\Despliegamiento\Python\A> git clone https://github.com/marcosfergar/calculadora.git
Cloning into 'calculadora'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 9 (delta 0), reused 9 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (9/9), done.
PS C:\Users\marcos.fernandezgarc\A\Despliegamiento\Python\A>
```

Config

Configure el email y el usuario

```
PS C:\Users\marcos.fernandezgarc\A\Despliegamiento\Python\A\calculadora> git config user.name "userA"
PS C:\Users\marcos.fernandezgarc\A\Despliegamiento\Python\A\calculadora> git config user.email "mfg99995@gmail.com"
```

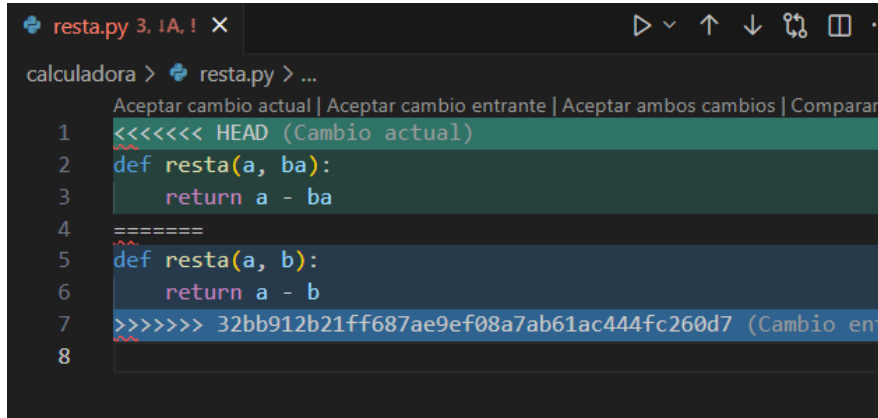
Comprobación:

```
[user]
name = userA
email = mfg99995@gmail.com
```

ERROR-SIMULADO

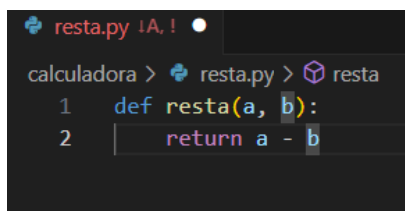
-1

Lo primero es traer los cambios de develop que contienen las funciones creadas por los usuarios B y C, pero simular un error subiendo el método resta desde A hacia develop en el que esta el método resta creado por B



```
resta.py 3, !A, ! X
calculadora > resta.py > ...
  Aceptar cambio actual | Aceptar cambio entrante | Aceptar ambos cambios | Comparar
1 <<<<<< HEAD (Cambio actual)
2 def resta(a, ba):
3     return a - ba
4 =====
5 def resta(a, b):
6     return a - b
7 >>>>>> 32bb912b21ff687ae9ef08a7ab61ac444fc260d7 (Cambio ent
8
```

En la imagen se puede observar el error al hacer el merge desde develop, para solucionarlo debemos indicarle a git con que versión quedarnos.



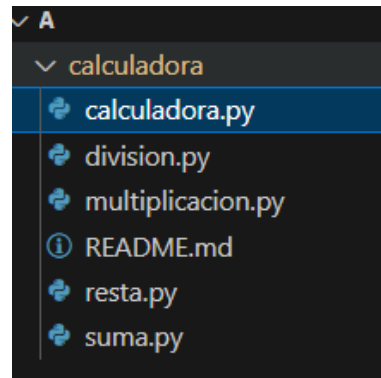
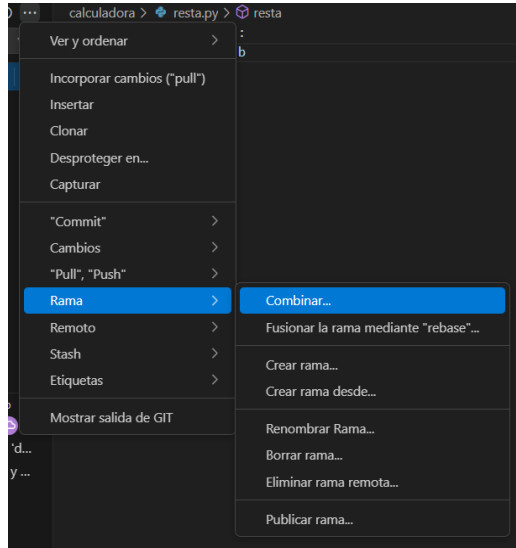
```
resta.py !A, ! ●
calculadora > resta.py > resta
1 def resta(a, b):
2     return a - b
```

En este caso me quedare con la versión de B, para indicarle la versión que queremos bastaría con borrar del archivo lo que queremos (como se ve en la imagen).

CREACION-MAIN

Ahora se traen los cambios desde develop hacia A

(utilizo otro método usando la interfaz de visual, en el apartado de git seleccionar los tres puntos Rama>Combinar>develop)



Código

```
import division,multiplicacion,resta,suma

def calculadora():
    print("=== Calculadora básica ===")
    while True:
        print("\nOperaciones disponibles:")
        print("1. Sumar")
        print("2. Restar")
        print("3. Multiplicar")
        print("4. Dividir")
        print("5. Salir")

        opcion = input("Elige una opción (1-5): ")

        if opcion == "5":
            print("¡Hasta luego!")
            break

        a = float(input("Ingresa el primer número: "))
        b = float(input("Ingresa el segundo número: "))

        if opcion == "1":
            print("Resultado:", suma.suma(a, b))
        elif opcion == "2":
            print("Resultado:", resta.rest(a, b))
        elif opcion == "3":
            print("Resultado:", multiplicacion.multiplicar(a, b))
        elif opcion == "4":
            print("Resultado:", division.division(a, b))
        else:
            print("Opción no válida.")

calculadora()
```

USUARIO-B

Clonado-B

```
PS C:\Users\marcos.fernandezgarc\A\Desplegamiento\Python\B> git clone https://github.com/marcosfergar/calculadora.git
Cloning into 'calculadora'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 9 (delta 0), reused 9 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (9/9), done.
PS C:\Users\marcos.fernandezgarc\A\Desplegamiento\Python\B> |
```

Config

```
PS C:\Users\marcos.fernandezgarc\A\Desplegamiento\Python\B\calculadora> git config user.name "userB"
PS C:\Users\marcos.fernandezgarc\A\Desplegamiento\Python\B\calculadora> git config user.email "mfg99995@gmail.com"
```

Comprobación:

```
[user]
name = userB
email = mfg99995@gmail.com
```

MÉTODOS

```
suma.py X
suma.py > ...
1 def suma(a, b):
2     return a + b
3
```

```
suma.py resta.py X
resta.py > resta
1 def resta(a, b):
2     return a - b
```

COMMITTS

suma userB B-sum-rest

resta userB B-sum-rest

Con el comando "git checkout develop" me situo en la rama y con el comando git merge B-sum-rest subo los cambios a develop

CODIGO

```
def resta(a, b):  
    return a - b
```

```
def suma(a, b):  
    return a + b
```


USUARIO-C

Clonado-C

```
PS C:\Users\marcos.fernandezgarc\A\Despliegamiento\Python\C> git clone https://github.com/marcosfergar/calculadora.git
Cloning into 'calculadora'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 9 (delta 0), reused 9 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (9/9), done.
PS C:\Users\marcos.fernandezgarc\A\Despliegamiento\Python\C> |
```

Config

```
PS C:\Users\marcos.fernandezgarc\A\Despliegamiento\Python\C\calculadora> git config user.name "userC"
PS C:\Users\marcos.fernandezgarc\A\Despliegamiento\Python\C\calculadora> git config user.email "mfg99995@gmail.com"
PS C:\Users\marcos.fernandezgarc\A\Despliegamiento\Python\C\calculadora> |
```

Comprobación:

```
[user]
name = userC
email = mfg99995@gmail.com
```

COMMIT



Con el comando "git checkout develop" me situo en la rama y con el comando git merge C-mult-div subo los cambios a develop

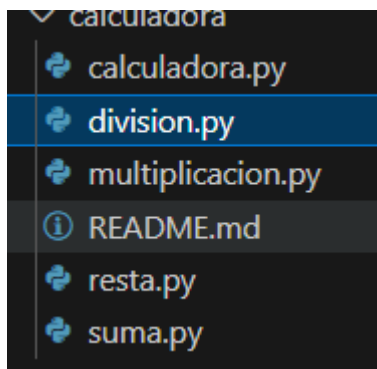
CODIGO

```
def multiplicar(a,b):
    return a*b
```

```
def division(a,b):
    return a/b
```

Master

Subí la aplicación completa al main



HOTFIX

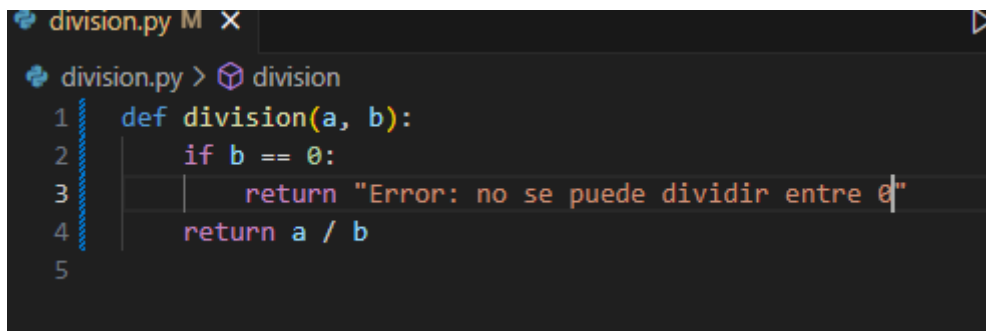
1-

Primero cree una rama de hotfix y entre en ella

```
>>git checkout -b hotfix  
Switched to a new branch 'hotfix'
```

2-

Modifique la función de división con el error al dividir entre 0

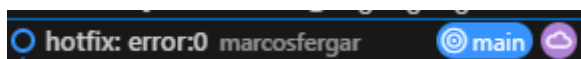


3-

Hice el commit desde la rama hotfix



Luego un merge desde main



DIFICULTADES

Las dificultades fueron al principio, nunca había trabajado con la estructura de develop con tres subramas y me toco empezar de cero varias veces.