

Resolução de conflitos no GitHub



<https://github.com/>



Resolução de conflitos no GitHub



Quando se trabalha em equipe, pode acontecer de dois ou mais desenvolvedores editarem o mesmo arquivo.

Posteriormente esses códigos devem ser mesclados na branch principal, o que pode gerar um conflito.

No momento da criação do Pull Request para o merge dos códigos, um alerta será gerado e o conflito deverá ser resolvido antes de prosseguir.



Resolução de conflitos no GitHub



Vamos criar um cenário que nos permita reproduzir uma situação de conflito.

Temos um repositório com a branch main e mais duas branches, programador1 e programador2.

O arquivo **principal.txt** será alterado por ambos os programadores.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ ls
README.md principal.txt

Fulano@Win11 ~/Documents/projeto1 (main)
$ git checkout -b programador1
Switched to a new branch 'programador1'

Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git checkout -b programador2
Switched to a new branch 'programador2'

Fulano@Win11 ~/Documents/projeto1 (programador2)
$ git branch
main
programador1
* programador2
```

Resolução de conflitos no GitHub



O arquivo principal.txt na branch main, possui as seguintes linhas de “código”.

A screenshot showing a file explorer window and a code editor. The file explorer shows a folder named 'projeto1' containing a file named 'principal.txt'. The code editor shows the content of 'principal.txt' with the following code:

```
Arquivo inicial do projeto1
metodoX(){
    principal
}
```

The word 'metodoX()' is underlined in red in the original image.

Resolução de conflitos no GitHub



Programador1 edita o arquivo.

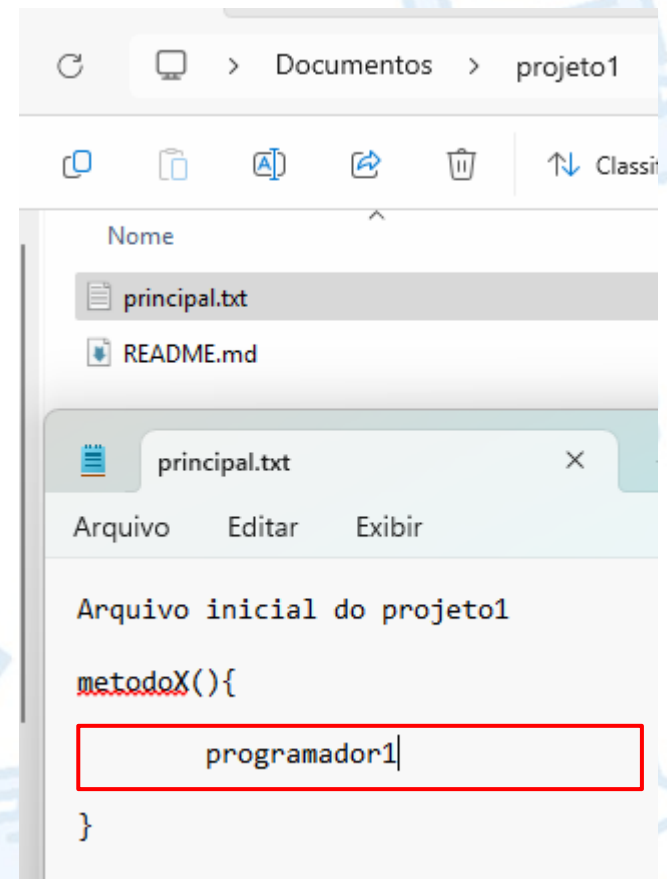
```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git checkout programador1
Switched to branch 'programador1'

Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git branch
main
* programador1
programador2
```

Programador1 acessar sua banch.

Programador1 edita o arquivo principal alterando uma linha de código.

Programador1 realiza o commit e o push para o GitHub.



Resolução de conflitos no GitHub



Na saída do comando de `git push` é possível visualizar um alertar para criar um pull request no GitHub.

```
Fulano@Win11 ~/Documents/projeto1 (programador1)
```

```
$ git add .
```

```
$ git commit -m "programador1 edicao"
```

```
Fulano@Win11 ~/Documents/projeto1 (programador1)
```

```
$ git push --set-upstream origin programador1
```

```
Enter passphrase for key '/c/Users/fulano/.ssh/id_ed25519':
```

```
Enumerating objects: 5, done.
```

```
Counting objects: 100% (5/5), done.
```

```
Delta compression using up to 8 threads
```

```
Compressing objects: 100% (3/3), done.
```

```
Writing objects: 100% (3/3), 332 bytes | 332.00 KiB/s, done.
```

```
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
```

```
remote:
```

```
remote: Create a pull request for 'programador1' on GitHub by visiting:
```

```
remote: https://github.com/fulano/projeto1/pull/new/programador1....
```




Resolução de conflitos no GitHub




Acessando a página web do repositório no GitHub também é possível visualizar a mensagem de criação do pull request do programador1.

 projeto1 Private


 Unwatch 1

 **programador1** had recent pushes 25 seconds ago

[Compare & pull request](#)

 main ▾

 2 Branches  0 Tags

 Go to file

t



Add file ▾

 Code ▾

Resolução de conflitos no GitHub



Para realizar o merge primeiro é necessário resolver o conflito.

 **This branch has conflicts that must be resolved**
Use the [web editor](#) or the command line to resolve conflicts before continuing.
 principal.txt

Merge pull request

▼

You can also merge this with the command line. [View command line instructions.](#)

principal.txt

1 conflict Prev ^ Next v ⚙ Mark as resolved

```
1  Arquivo inicial do projeto1
2
3  metodoX(){
4
5  <<<<<< programador1
6      programador1
7  =====
8      principal()
9  >>>>>> main
10
11  }
12
```


Resolução de conflitos no GitHub



Para realizar o merge primeiro é necessário resolver o conflito.

Os arquivos podem ser alterados no editor web.

The screenshot displays the GitHub web editor interface for a file named `principal.txt`. The interface shows a conflict state with a red "1 conflict" indicator and navigation buttons like "Prev", "Next", and "Mark as resolved". The code editor shows the following content:

```
1  Arquivo inicial do projeto1
2
3  metodoX(){
4
5  <<<<<< programador1
6      programador1
7  =====
8      principal()
9  >>>>>> main
10
11  }
12
```

The conflict resolution interface is shown in two states. In the top state, the conflict is highlighted with a yellow background. In the bottom state, the "Mark as resolved" button is highlighted with a red border, indicating the next step in the resolution process.

Resolução de conflitos no GitHub



Programador2 edita o arquivo.

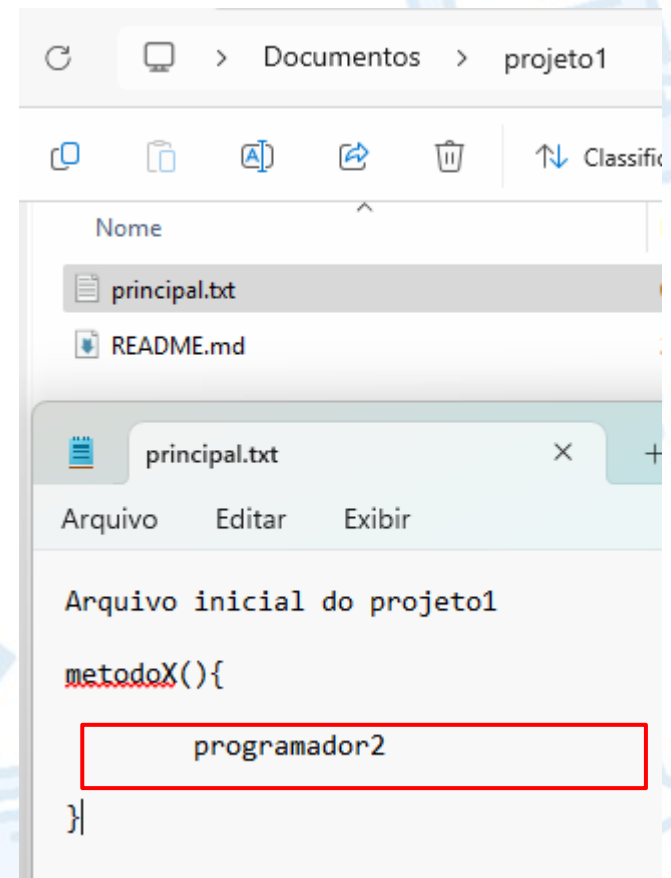
```
Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git checkout programador2
Switched to branch 'programador2'

Fulano@Win11 ~/Documents/projeto1 (programador2)
$ git branch
  main
  programador1
* programador2
```

Programador2 acessar sua branch.

Programador2 edita uma linha do arquivo principal.

Programador2 realiza o commit e o push para o GitHub.



Resolução de conflitos no GitHub



Na saída do comando de `git push` é possível visualizar um alertar para criar um pull request no GitHub.

```
Fulano@Win11 ~/Documents/projeto1 (programador2)
```

```
$ git add .
```

```
$ git commit -m "programador2 edicao"
```

```
Fulano@Win11 ~/Documents/projeto1 (programador2)
```

```
$ git push -u origin programador2
```

```
Enter passphrase for key '/c/Users/fulano/.ssh/id_ed25519':
```

```
Enumerating objects: 5, done.
```

```
Counting objects: 100% (5/5), done.
```

```
Delta compression using up to 8 threads
```

```
Compressing objects: 100% (3/3), done.
```

```
Writing objects: 100% (3/3), 332 bytes | 332.00 KiB/s, done.
```

```
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
```

```
remote:
```

```
remote: Create a pull request for 'programador2' on GitHub by visiting:
```

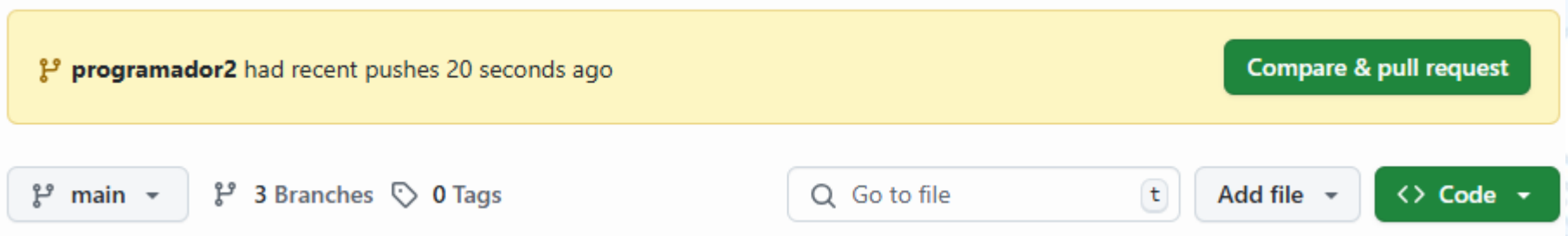
```
remote: https://github.com/fulano/projeto1/pull/new/programador2...
```



Resolução de conflitos no GitHub



Acessando a página web do repositório no GitHub agora é possível visualizar a mensagem de criação do pull request do programador2.



Resolução de conflitos no GitHub




Para realizar o merge primeiro é necessário resolver o conflito.



This branch has conflicts that must be resolved

Use the [web editor](#) or the command line to resolve conflicts before continuing.

 principal.txt

Resolve conflicts

Merge pull request



You can also merge this with the command line. [View command line instructions.](#)

principal.txt

1 conflict

Prev ^

Next v



Mark as resolved

```
1  Arquivo inicial do projeto1
2
3  metodoX(){
4
5  <<<<<< programador2
6      programador2
7  =====
8      programador1
9  >>>>>> main
10
11  }
12
```

Resolução de conflitos no GitHub



Para realizar o merge primeiro é necessário resolver o conflito.
Os arquivos podem ser alterados no editor web.

The screenshot displays the GitHub web editor interface for a file named `principal.txt`. The editor shows a conflict between two versions of the file. The top version (from `programador2`) is highlighted in yellow, and the bottom version (from `programador1`) is highlighted in light blue. The conflict is resolved by taking the changes from `programador2`. The `Mark as resolved` button is highlighted with a red box.

```
principal.txt
1  Arquivo inicial do projeto1
2
3  metodoX(){
4
5  <<<<<< programador2
6      programador2
7  =====
8      programador1
9  >>>>>> main
10
11 }
12
```

principal.txt 1 conflict Prev ^ Next v ⚙️ Mark as resolved

```
principal.txt
1  Arquivo inicial do projeto1
2
3  metodoX(){
4
5      programador2
6
7  }
8
```

principal.txt 1 conflict Prev ^ Next v ⚙️ Mark as resolved

Atualizando branches locais



Agora a branch local do programador1 estará desatualizada em relação às branches main e programador2.

Trabalhar com códigos muito defasados, principalmente em relação a branch main pode aumentar as chances de conflitos na hora de realizar o merge futuramente.

O ideal é tentar manter a sua branch mais atualizada em relação à branch main.

Para isso é possível fazer um pull na branch main local, assim esta ficará atualizada.

Também é possível fazer um merge da branch main remota com uma branch específica local (programador1 por exemplo).

Atualizando branches locais



```
Fulano@Win11 ~/Documents/projeto1 (programador2)
```

```
$ git checkout main
```

```
Fulano@Win11 ~/Documents/projeto1 (main)
```

```
$ git commit -m "programador2 edicao"
```

```
Fulano@Win11 ~/Documents/projeto1 (main)
```

```
$ git pull
```

```
Enter passphrase for key '/c/Users/fulano/.ssh/id_ed25519':
```

```
remote: Enumerating objects: 2, done.
```

```
remote: Counting objects: 100% (2/2), done.
```

```
remote: Compressing objects: 100% (2/2), done.
```

```
remote: Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
```

```
Unpacking objects: 100% (2/2), 1.72 KiB | 195.00 KiB/s, done.
```

```
From github.com:diegomo2/projeto1
```

```
9ec7c50..a2bdb7e main -> origin/main
```

```
a06d2d7..577033c programador2 -> origin/programador2
```

```
Updating 9ec7c50..a2bdb7e
```

```
Fast-forward
```

```
principal.txt | 2 +-  
1 file changed, 1 insertion(+), 1 deletion(-)
```

Atualizando branches locais



Atualizando a branch programador1 local em relação à branch main.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git checkout programador1

Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git merge main
Updating 5b2b1fc..a2bdb7e
Fast-forward
 principal.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

Agora podemos atualizar a branch programador1 remota.

```
Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git commit -m "Atualizando com branch main"

Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git push
Enter passphrase for key '/c/Users/fulano/.ssh/id_ed25519':
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:diegomo2/projeto1.git
 5b2b1fc..a2bdb7e  programador1 -> programador1
```

Resolução de conflitos locais



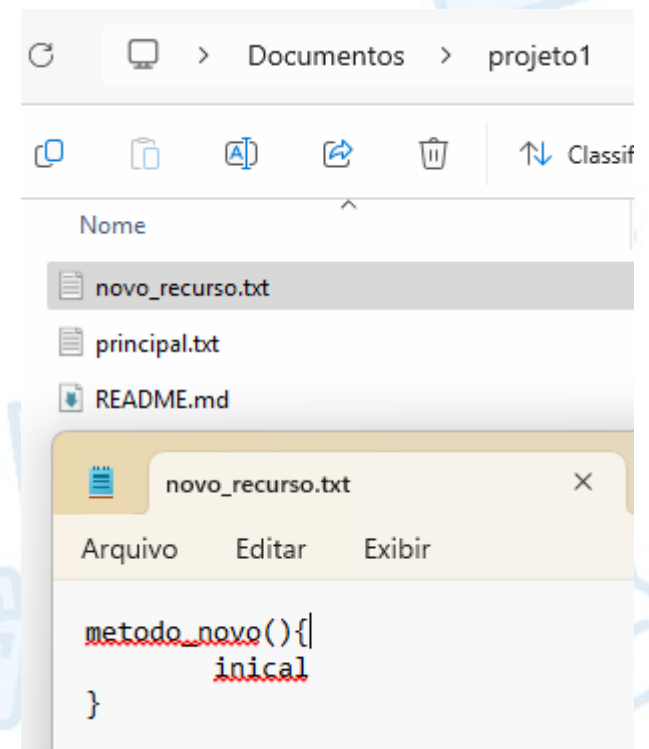
Agora vamos criar um cenário onde um conflito irá acontecer e será resolvido localmente.

Neste caso, o arquivo **novo_recurso.txt** será editado.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git branch
* main
  programador1
  programador2

Fulano@Win11 ~/Documents/projeto1 (main)
$ ls
README.md novo_recurso.txt principal.txt
```

Neste ponto o arquivo está igual em todas as branches.



Resolução de conflitos locais

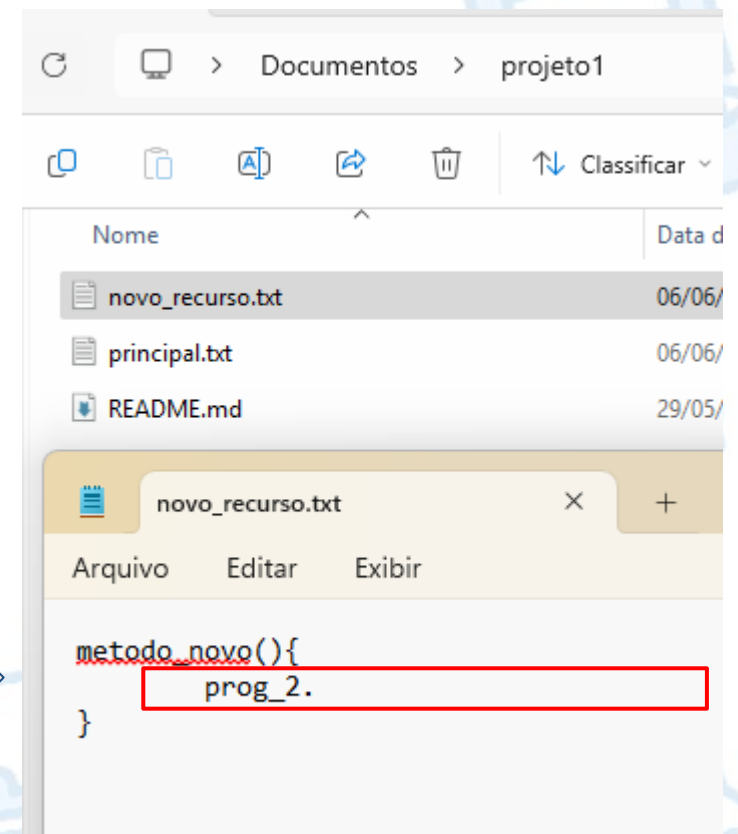


Programador2 edita o arquivo.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git checkout programador2

Fulano@Win11 ~/Documents/projeto1 (programador2)
$ git branch
main
programador1
* programador2

Fulano@Win11 ~/Documents/projeto1 (programador2)
$ ls
README.md novo_recurso.txt principal.txt
```



Programador2 acessar sua branch.

Programador2 edita uma linha do arquivo.

Programador2 faz o commit, push e pull request / merge no GitHub.

Resolução de conflitos locais

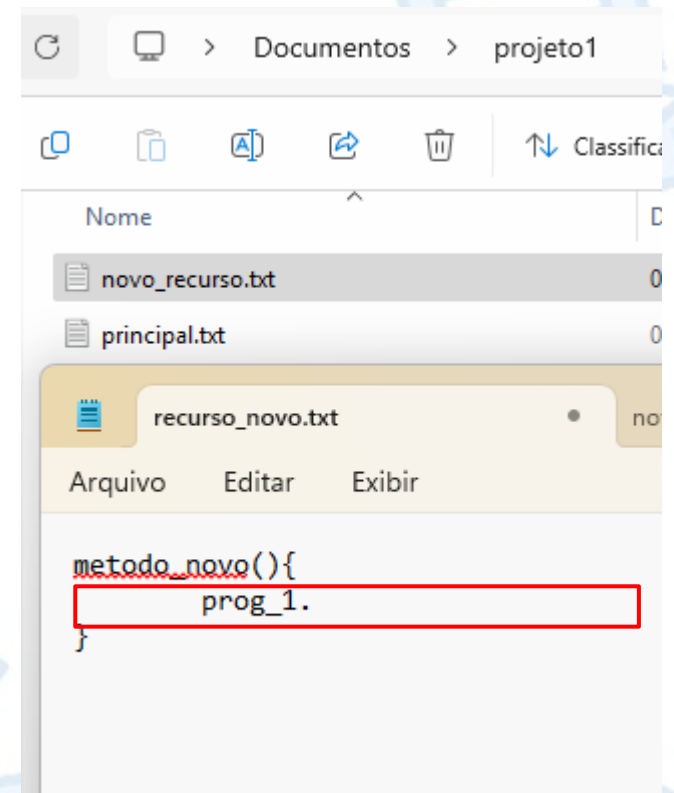


Programador1 edita o arquivo.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git checkout programador1

Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git branch
  main
* programador1
  programador2

Fulano@Win11 ~/Documents/projeto1 (programador1)
$ ls
README.md novo_recurso.txt principal.txt
```



Programador1 acessar sua branch.

Programador1 edita uma linha do arquivo.

Programador1 faz o commit.

Resolução de conflitos locais

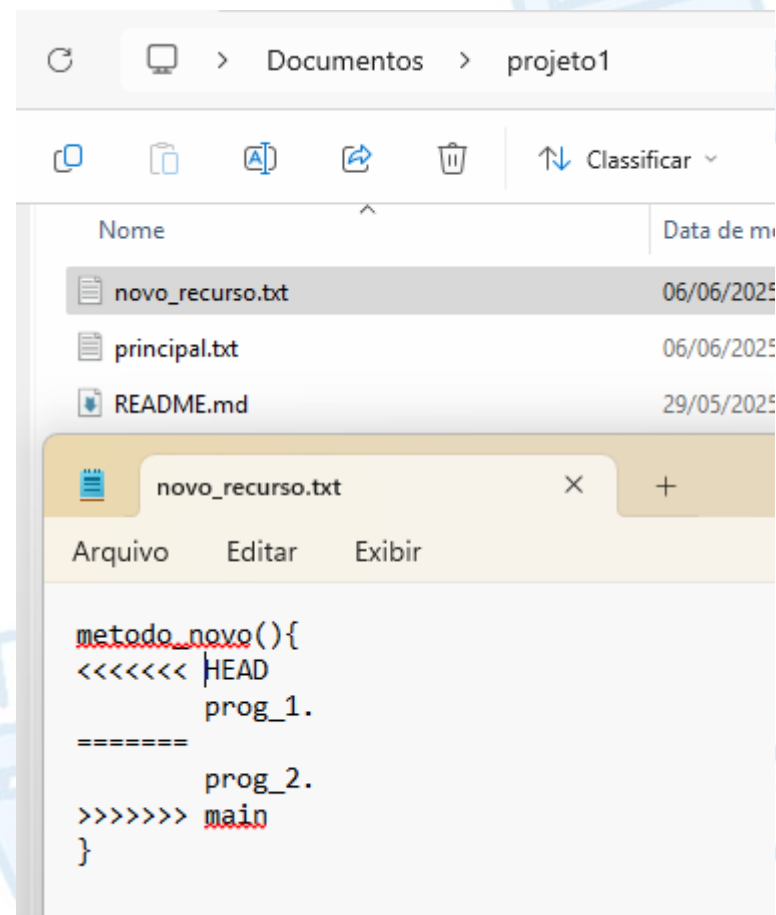


Neste ponto se o programador1 tentar atualizar sua branch com a branch main, receberá um aviso de que houve um conflito.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git merge main
Auto-merging novo_recurso.txt
CONFLICT (content): Merge conflict in novo_recurso.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Ao abrir o arquivo localmente, é possível ver as linhas em conflito.

Para resolver o conflito, basta editar o arquivo e fazer um commit.



Resolução de conflitos locais

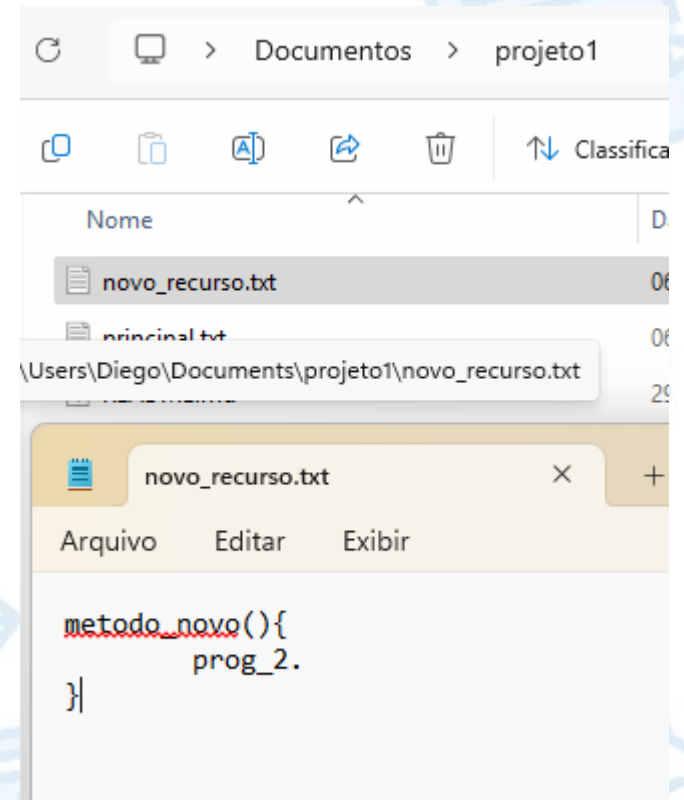


O arquivo deve ser editado para resolver o conflito.
Com o arquivo já editado é possível prosseguir com o fluxo.

```
Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git add .

Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git commit -m "conflito resolvido"

Fulano@Win11 ~/Documents/projeto1 (programador1)
$ git merge main
Already up to date.
```



Conflito resolvido.

Obrigado

