## mentorama.

| PASSO | ATIVIDADE  | COMANDO  |
|-------|--|--|
| 1     | Crie uma pasta ExerciciosGit e inicialize um repositório Git no seu computador;                          | git init   |
| 2     | Adicione um arquivo nessa pasta com o nome README.md;  | touch README.md  |
| 3     | Adicione qualquer conteúdo nesse arquivo;  | Adicionar um titulo por exemplo e salvar   |
| 4     | Faça um commit dessas alterações;  | git add README.md<br>git commit -m "Primeiro commit"                                   |
| 5     | Verifique o status e veja se existem mudanças para serem "comitadas";                                    | git status   |
| 6     | Crie uma branch chamada "develop";   | git branch develop   |
| 7     | Altere o arquivo README.md e adicione outro arquivo na pasta;  | Alterar o Arquivo README.md e salvar touch meuprograma.py                              |
| 8     | Verifique o status das mudanças;   | git status   |
| 9     | Faça commit dessas mudanças na branch "develop";   | git checkout develop<br>git add .<br>git commit -m "Segundo commit"                    |
| 10    | Certifique-se que seu status esteja sem mudanças para serem "comitadas";                                 | git status   |
| 11    | Faça novamente outra alteração no arquivo README.md;   | Alterar o Arquivo README.md e salvar   |
| 12    | Faça commit dessa alteração no branch "develop";   | git commit -am "Terceiro commit"   |
| 13    | Certifique-se que seu status esteja sem mudanças para serem "comitadas";                                 | git status   |
| 14    | Agora faça um merge da branch "develop" e "master";  | git checkout master<br>git merge develop master  |
| 15    | Certifique-se que as alterações da<br>branch "develop" estão na branch<br>"master" através dos logs;     | git log  |
| 16    | Vá novamente para a branch "develop" e crie uma branch chamada "feature", esta tarefa estará finalizada; | git checkout develop<br>git branch feature   |
| 17    | Altere uma linha no arquivo<br>README.md e faça commit nessa<br>nova branch;                             | git checkout feature<br>Alterar o README.md e salvar<br>git commit -am "Quarto commit" |
| 18    | Volte para a branch "develop" e crie<br>mais uma branch, mas com o nome<br>"feature2";                   | git checkout develop<br>git branch feature2  |
| 19    | Altere o arquivo README.md na<br>mesma linha da alteração feita na<br>branch "feature";                  | Alterar o README.md e salvar   |

## mentorama.

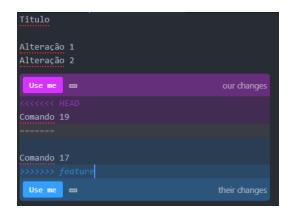
| 20              | Faça commit dessa alteração na branch "feature2";  | git checkout feature2<br>git commit -am "Quinto commit"   |
|-----------------|--|---|
| 21              | Agora tente fazer merge das branchs "feature" e "feature2" com a branch "develop";                         | git merge feature feature2 develop  |
| 22              | Acontecerá propositalmente um conflito no arquivo README.md depois que realizar o merge da segunda branch; | ok  |
| 23              | Resolva esse conflito e verifique se<br>ambas as alterações estão na branch<br>"develop";                  | git diff feature2 feature Eu escolhi manter o texto da ultima alteração no README.md git add . git commit -m "Sexto commit" git merge feature feature2 git checkout develop git merge feature2 develop git status git log (verifique a posição do HEAD e qual branch você está etc)                           |
| <mark>24</mark> | Por último, faça merge das branch "main" e "develop";  | git checkout master<br>git merge master develop   |
| 25              | Crie um repositório no github e faça push;   | Siga as instruções da aula sobre Acesso Remoto com Github e ao criar o diretório no Github, utilize os comandos sugeridos na tela para fazer upload dos seus arquivos no Githubb  Obs: Ao fazer a ligação do Git com Github, aparecerá uma tela para informar o email e senha cadastrado na plataforma do Git |

Algumas dicas: Você poderá escolher algumas instruções um pouco diferentes das apresentadas por aqui. O objetivo é que você trabalhe com os comandos básicos.

Experimente deixar a linha de comando do git de um lado da tela e o editor de texto (código) do outro lado da tela, para que você possa fazer as alterações nos arquivos e visualizar, a medida que navega nas branches.

Sobre o conflito do Passo 22, uma possível tela que irá aparecer no Atom é a seguinte:

## mentorama.



Você também poderá utilizar o comando git diff para avaliar as diferenças entre os arquivos nas branches ©

Muito obrigado. Sucesso sempre.

Felipe M. Assunção