Comandos do Git:

- **git init** –> inicia um novo repositório local na pasta atual;
- **git add** –> adiciona arquivos selecionados para a área de stage como uma nova versão do código;
    - **git add .** -> adiciona todas os arquivos/pastas modificadas ou criadas
- **git status** –> verifica o status atual do repositório, incluindo arquivos modificados na área do stage;
- **git commit** -> grava as alterações feitas na área de stage como uma nova versão do código;
    - Modelo: git commit -m (string)
        - ex.: git commit -m “criando arquivo”
- **git log** –> vê históricos de commits;
    - Funcionalidade extra: git log –all
        - Mostra todas as commits
- **git checkout** -> usado para alternar entre branches, criar novos branches, ou mudar para uma versão anterior do código em uma branch existente;
    - Modelo para navegação: git checkout (id da commit)
        - Ex.: git checkout b8e93f979c3143fe57976c0fc53156b3ae43bf70
    - Modelo para criar branches: primeiro você terá que escolher em que arquivo irá realizar a ramificação e indo até ela por meio do git checkout (id), e em seguida git checkout -b (string), onde “(string)” será o nome da ramificação, e a partir disso você já será redirecionada para ela
        - Ex.: git checkout master
            - git checkout -b “Arquivamento”
- **git branch** -> alista as branches do projeto;
- **git merge** -> permite unir o conteúdo de duas ou mais branches em uma única branch. É usado para integrar alterações feitas em uma branch secundária (geralmente uma branch de desenvolvimento) na branch principal (geralmente a branch maior);

    - Modelo de integração: primeiramente você terá que ir para a commit base que receberá o incremento e após isso, git merge (string), onde em "string" será a ramificação que será unida a commit base
        - Ex.: git merge somar
- git remote add origin -> usado para conectar o projeto local com o repositório do github. Você aponta a URL do projeto do GitHub
- git push -> comando para enviar para o github
- git clone -> pega o repositório do github e baixa na sua máquina. Você aponta a URL do projeto no GitHub que quer fazer
- (HEAD) -> nos diz em que commit estamos no projeto, ela aparece após digitar "git commit";
- (MASTER) -> commit mais avançada do projeto, vulgo principal, ela aparece após digitar "git commit";

Como enviar/sincronizar arquivos com o GitHub:

1. Entre no terminal do vscode (ctrl + ');
2. Digite: git remote add origin (link do repositório).git
    - Opcional: git branch -M main
        - Para caso queira renomear o arquivo principal de master para main
3. Digite: git push -u origin master
4. Após isso os arquivos serão enviados
5. E caso tenha realizado alguma alteração nos arquivos basta apenas digitar: git push

Como pegar arquivos com o GitHub:

1. Entre no terminal, pode ser o do Windows
2. Digite o caminho onde você irá colocar os arquivos: cd (caminho do arquivo)
3. Digite: git clone (link do repositório).git
4. E para entrar diretamente no seu editor de código fonte, digite: code .

Como trabalhar de forma colaborativa no Git e GitHub:

1. Ir até o repositório;
2. Procurar a função “Fork”;
3. Cria um fork;
4. A partir disso os passos posteriores são o mesmo do: “Como pegar arquivos com o GitHub”;
5. Após esses passos o projeto será aberto no seu editor de código;
6. Agora você terá que criar a sua vertente do código, ou seja, a sua branch
7. Com as alterações realizadas basta fazer a adição e commitar o projeto;
8. E para mandar para o github você digita: git push origin (nome da sua branch);
9. A partir disso haverá duas vertentes: a sua e a principal;
10. Todavia, no github terá uma opção que é: contribute -> open pull request
    a. Onde sua alteração poderá ser inserida no projeto principal