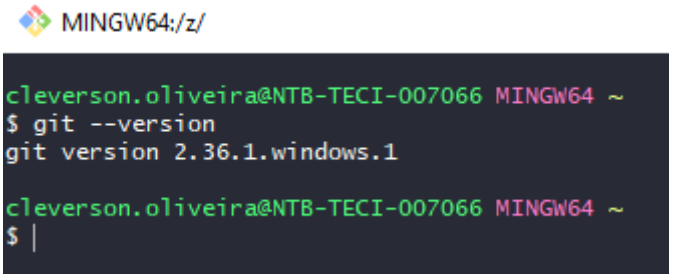
1. Qual o comando para obter a versão instalada do Git?

get –version



2. Qual o efeito da execução de cada um dos comandos abaixo?

a. git help

b. git help checkout

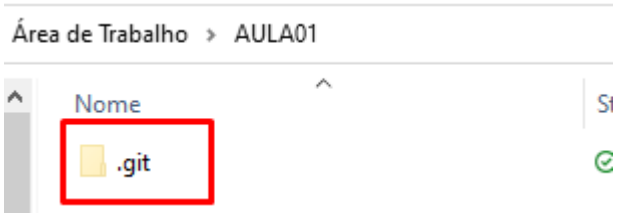
Atualiza os arquivos na árvore de trabalho para corresponder à versão no índice ou na árvore especificada. Se nenhum pathspec foi fornecido, o **git checkout** também será atualizado `HEAD` para definir o branch especificado como o branch atual.

c. git help merge

Incorpora as alterações dos commits nomeados (desde o momento em que seus históricos divergiram da ramificação atual) na ramificação atual. Este comando é usado pelo **git pull** para incorporar alterações de outro repositório e pode ser usado manualmente para mesclar alterações de uma ramificação em outra.

d. git init

O comando git init cria um novo repositório do Git. Ele pode ser usado para converter um projeto existente e não versionado em um repositório do Git ou inicializar um novo repositório vazio.



e. git add –all

Comando é para adicionar os arquivos ou alterações ao repositório.

f. git add -u

Adiciona conteúdo ao índice ou área de teste.

g. git config -l

É uma ferramenta chamada git config que permite ver e atribuir variáveis de configuração que controlam todos os aspectos de como o Git aparece e opera.

h. git mv a.txt b.txt

git-mv - Move ou renomeia um arquivo, um diretório ou um link simbólico

i. git reset –hard

Ao redefinir arquivos no Git, você basicamente tem duas opções: você pode redefinir os arquivos com força ou redefinir com software.

j. git log -27

Depois de ter criado vários commits, ou se você tiver clonado um repositório com um histórico de commits existente, você provavelmente vai querer olhar para trás para ver o que aconteceu. A ferramenta mais básica e poderosa para fazer isso é o git logcomando.

```
cleverson.oliveira@NTB-TECI-007066 MINGW64 /c/Users/cleverson.oliveira.IPOG.000/
OneDrive - IPOG - INSTITUTO DE POS-GRADUAÇÃO E GRADUAÇÃO LTDA/Área de Trabalho/A
ULA01 (master)
$ git log -27
commit 0c01122a412eef054cfd33c405de0f08c7d77a4e (HEAD -> master)
Author: Cleverson Oliveira <cleoepifanio2@gmail.com>
Date:   Fri Jun 3 14:25:45 2022 -0300

    PESQUISA REST API

commit 5e4688577509d03cbe350f10545fdc709599a61b
Author: Cleverson Oliveira <cleoepifanio2@gmail.com>
Date:   Fri Jun 3 13:58:11 2022 -0300

    PESQUISA: REST API
```

3. O fluxo “clássico” de interação com o Git é algo como “alterar um ou mais arquivos”, “acrescentar essas mudanças para serem contemplados no próximo commit” e, finalmente, executar um “commit”. Quais os comandos necessários para realizar os dois últimos “passos” desse fluxo?

Primeiro: git add [file]

Segundo: git commit -m “mensagem de commit”

4. Qual o comando deve ser executado para identificar o que foi alterado desde o último “commit”?

git diff

5. Em um dado repositório, arquivos simplesmente copiados para lá, ou seja, _untracked_, podem ser exibidos/identificados com que comando?

git ls-files . --exclude-standard --others

6. Qual o comando para efetuar um _commit_?

git commit -m “Mensagem”

7. Qual o comando que devemos empregar para descartar mudanças ocorridas no arquivo teste.txt, por exemplo?

git reset teste.txt

8. O que deve ser feito para que um determinado diretório do seu repositório seja ignorado pelo Git? Faça uma busca por **.gitignore**.

Crie ou edite o arquivo .gitignore e adicione o nome do diretório em uma linha do arquivo.

9. O que acontece se o seu repositório local for acidentalmente removido?

Todos os arquivos do meu repositório local serão removidos, porém as mudanças não reflitirão no repositório remoto, a não que seja realizado um commit. É possível reverter a exclusão por meio dos comandos git reset e git checkout, que podem ser combinados para obter o resultado desejado.

10. Como clonar um repositório remoto?

Através do comando ‘git clone’

11. Em alguns cenários **git log** pode produzir extensos resultados. Se houver interesse em visualizar o histórico de um repositório, onde cada mudança é fornecida exatamente em uma única linha, qual o comando que deve ser empregado?

git log -1

12. Em qual arquivo o Git armazena informações de configuração empregadas por usuário?

No arquivo .gitconfig

13. Qual o comando para criar um repositório local?

git init [nome do repositório]

14. Qual o nome do diretório criado pelo Git quando se executa o comando **git init**?

.git

15. Qual o comando para adicionar todos os arquivos modificados? (Aqueles para os quais **git status** identificam como **modified**?)

git add -u

16. O Git faz uso do valor de hash conhecido por SHA1. O que isto significa? Qual o propósito? O que é SHA1?

O git usa SHA com o propósito de segurança e de identificação de alterações nos arquivos. SHA1 é um algoritmo que recebe alguns dados como entrada e gera uma string única de 40 caracteres através desses dados. Ou seja, nenhuma outra entrada de dados deveria produzir o mesmo hash. Porém, caso a mesma entrada de dados seja usada, ela sempre irá produzir o mesmo hash.

Isso é extremamente importante, pois o número de revisão do git é nessa hash SHA1, então mesmo que várias pessoas estejam usando o mesmo repositório, se o número de revisão for igual para todas elas, isso significa que todos estão trabalhando naquele mesmo conjunto de arquivos. Isso traz maior segurança, pois se qualquer alteração for realizada em qualquer bit que seja, ou mesmo durante a transferência dos arquivos, se ocorrer qualquer perda de bits, o git irá identificar.

17. Qual a palavra para indicar o último _commit_ em vez do valor de hash SHA1 correspondente?

Revisão

18. Quando se cria dois arquivos usando um editor de texto qualquer e, na sequência, executamos o comando **git add -u**, os dois arquivos criados passam de _untracked_ para _new file_?

Não, a terminação '-u' apenas modifica arquivos que já foram adicionados, com este comando, novos arquivos não serão adicionados.

19. Qual o efeito da execução dos dois comandos abaixo, nesta ordem, em um dado repositório?

git reset --soft HEAD~1

git reset --hard

O primeiro não irá mexer com o arquivo index ou com a working tree, porém irá reiniciar a head para o commit que for determinado. Todos os arquivos modificados ficarão como mudanças a serem commitadas.

Já no segundo caso, o índice e a working tree serão resetados e quaisquer mudanças nos arquivos já rastreados desde o commit indicado serão descartadas.

20. Após o emprego de um ambiente integrado de desenvolvimento (IDE), é comum a criação de arquivos e diretórios. Qual o comando que podemos empregar para remover arquivos e diretórios _untracked_?

git clean -fd

21. Qual o nome do arquivo no qual podemos inserir a indicação para o Git de arquivos e diretórios a serem ignorados?

.gitignore

22. Quando se cria o arquivo _MinhaClasse.class_ em um dado diretório e desejamos que arquivos com a extensão .class, como neste caso, sejam ignorados por todos os membros de uma equipe que estão contribuindo com um dado projeto, como devemos proceder?

Pergunta não está muito clara, mas caso queira ignorar todos os arquivos .class, deve-se colocar esta regra no arquivo .gitignore, a regra seria:

*.class

23. jQuery é uma famosa biblioteca em JavaScript. Consulte detalhes em [jQuery](http://jquery.com). O repositório correspondente encontra-se em [gitRep](https://github.com/jquery/jquery.git). Faça o clone deste repositório.

Realizado.

24. No repositório **jqueryrepo**, criado no passo anterior, qual o efeito do comando

git shortlog -sne?

Retorna as mensagens de log da saída padrão, mostrando somente o resumo da contagem do commit, ordenadas pelo número de commits por autor e mostrando o nome e email de cada autor. Exemplo:

```
cleverson.oliveira@NTB-TECI-007066 MINGW64 /c/Users/cleverson.oliveira.IPOG.000/
OneDrive - IPOG - INSTITUTO DE POS-GRADUAÇÃO E GRADUAÇÃO LTDA/Area de Trabalho/A
ULA01 (master)
$ git shortlog -sne
     2  Cleverson Oliveira <cleoepifanio2@gmail.com>

cleverson.oliveira@NTB-TECI-007066 MINGW64 /c/Users/cleverson.oliveira.IPOG.000/
OneDrive - IPOG - INSTITUTO DE POS-GRADUAÇÃO E GRADUAÇÃO LTDA/Area de Trabalho/A
ULA01 (master)
$
```

25. No repositório **jqueryrepo**, qual o efeito de **git remote -v**?

Mostra o endereço do repositório remoto.

origin https://github.com/jquery/jquery (fetch)

26. Um repositório Git pode ser etiquetado ao longo do tempo. Ou seja, _commits_ específicos podem ser “marcados” ou “etiquetados” para facilitar referências posteriores. Para listar todas as “etiquetas” (_tags_) estabelecidas para um dado repositório, qual comando deve ser executado?

git tag

27. Caso um dato repositório retorne muitas “marcas” ou “etiquetas” para o comando **git tag**, como retornar apenas aquelas que atendem a determinado padrão, por exemplo, iniciadas por 2.0?

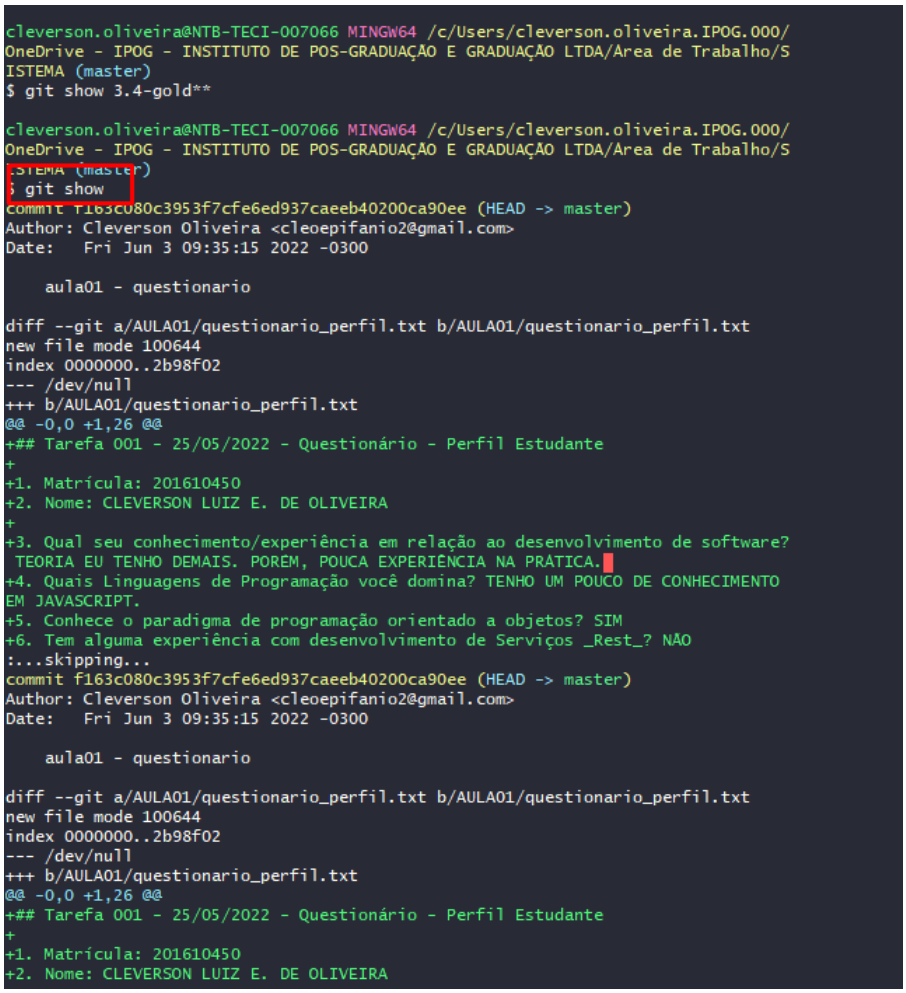
git tag -l "2.0*"

28. Qual o efeito do comando **git tag -a 3.4-gold -m “minha versão ouro”**?

Cria uma annotated tag ‘3.4-gold’ com a mensagem de tagging “minha versão ouro”

29. Após executado o comando acima, qual o efeito de **git show 3.4-gold**?

É possível ver os dados da tag, assim como o commit que foi etiquetado



30. O que o comando **git push origin 3.4-gold** teria como efeito?

Iria atualizar o repositório remoto com o repositório local sob a tag 3.4-gold, caso houvesse permissão para isso.

31. Após executar um commit, qual o efeito de **git commit --amend**?

O git commit --amend permite adicionar alterações ao último commit, por exemplo, na situação em que você esqueceu de adicionar alguma coisa ao último commit, seja um arquivo ou uma certa alteração aos arquivos.

32. Após executar **git add x.txt**, qual o efeito de **git reset HEAD x.txt**?

As mudanças no arquivo e adição não serão commitadas.

33. Após alterar o conteúdo de um arquivo committed em passo anterior, qual o efeito do comando **git checkout -- a.txt**?

Efeito porque o arquivo já foi nenhum commitado.

34. Qual a diferença entre os comandos **git reset HEAD a.txt** e **git checkout -- a.txt**?

O primeiro irá resetar a HEAD para o estado identificado para o arquivo x.txt, enquanto o segundo irá restaurar o arquivo a.txt da working tree.

35. Veja como interpretar o resultado de git diff [aqui](https://medium.com/therobinkim/how-to-read-a-git-diff-6c87a9dc47c5). Execute, em um dos seus projetos, o comando **git diff HEAD~1 HEAD** e certifique-se de que entende o resultado apresentado.