1. Por un lado se dice que Git es un SCM, y por otro un VCS. ¿A qué hacen referencia estos acrónimos? Citar otros ejemplos de programas que hagan lo mismo que Git.

   .....

   .....

   .....

2. ¿Cuáles son los tres estados posibles de un archivo en Git?

   .....

   .....

   .....

3. ¿En qué archivo suele guardar Git su configuración? ¿Cómo podemos ver la configuración que está usando Git en la línea de comandos?

   .....

   .....

   .....

4. ¿Cómo podemos obtener ayuda de Git para algún comando en particular?

   .....

   .....

   .....

5. Crear una cuenta de usuario en GitHub: `https://github.com`. Usar un nombre de usuario adecuado para el mundo laboral. El perfil de GitHub es algo que miraría un futuro empleador si se dedican al desarrollo de *software*.

6. Completar los espacios en blanco.
   (a) _____ agrega archivos al próximo *commit*.
   (b) El comando `git` _____ muestra un registro con los *commits* de un _____.
   (c) _____ crea un repositorio en el directorio de trabajo.
   (d) Para subir un _____ local a uno remoto usamos `git` _____.
   (e) Las dos maneras de obtener un repositorio de Git son o usando _____ en un directorio, o _____ un repositorio existente con _____.
   (f) Si quiero hacer cambios en un _____ compartido por un equipo de trabajo es conveniente hacerlos en una _____.
   (g) Cuando fusionamos dos ramas puede haber un _____ si el mismo _____ fue modificado en ambas ramas.
   (h) Para resolver un conflicto de _____ hay que editar el _____ que genera el conflicto y marcarlo como resuelto con _____.
   (i) No podemos realizar un _____ a un remoto si nuestra copia local no está actualizada con los últimos _____ del repositorio remoto. En ese caso tendremos que realizar un _____ primero antes de subir nuestro trabajo.

7. Completar lo que falta en los siguientes comandos de Git.
   (a) `git add` _____ (agrega todos los archivos del directorio al *staging area*)

(b) `git status` _____ (versión abreviada del estado del repositorio)

(c) `git` _____ `-10` (ver los últimos 10 *commits*)

(d) `git commit` _____ (arregla el último *commit*)

(e) `git` _____ (agrega `index.html` y el directorio `css` al *index*)

(f) `git checkout` _____ (crea y cambia a una rama llamada `new-branch`)

(g) `git` _____ `"Juan Perez"` (configura el nombre de usuario a "Juan Perez")

(h) `git` _____ (muestra las URLs asociadas con cada remoto)

(i) `git` _____ (sincroniza la rama en la que estoy con la rama remota rastreada)

(j) `git` _____ (fusiona la rama `hotfix` con la rama en la que estoy parado)

8. ¿Qué comando de Git usarías para ver qué archivos están en el *staging area*?

   _____

9. ¿Cómo podemos ver un registro de los últimos 3 commits de un repositorio?

   _____

10. ¿Qué comando de Git usarías para agregar un archivo llamado `LICENSE` al próximo *commit*?

    _____

11. ¿Cómo podemos hacer un *commit* en Git en un solo comando si no hay archivos nuevos para rastrear?

    _____

12. ¿Cómo eliminar un remoto llamado `github`?

    _____

13. Borrar la rama llamada `hotfix`.

    _____

14. Mostrar un registro de *commits* de a uno por línea y dibujar un gráfico mostrando todas las ramas.

    _____

15. ¿Cómo cambiar el mensaje del último *commit*?.

    _____

16. ¿Cómo crear una rama llamada `create-user-form` y cambiar a esa rama en un sólo comando?

    _____

17. Subir la rama `main` del repositorio a un remoto llamado `heroku`. Setear la rama remota como "rastreada".

    _____

18. Dar una lista de todas las ramas, incluyendo ramas remotas rastreadas.

    _____

19. Suponiendo que acabamos de hacer un *commit* en un repositorio local. Dar los comandos necesarios para agregar un nuevo remoto llamado `work` que apunte a `https://github.com/user/repo.git`. Luego subir el repositorio local al remoto recién creado. El repositorio local tiene sólo una rama llamada `main`.

    .....

    .....

    .....

20. Dar los comandos para configurar la identidad en `git`.

21. Dar los comandos para hacer un *commit*, agregando todos los archivos del directorio de trabajo al mismo.

22. Estamos trabajando en el archivo `index.html` en la rama `master` de nuestra web y nos piden que arreglemos urgente un error de tipeo en la pagina `about.html` y subamos la nueva versión al remoto. Lo que está en `origin/master` es la rama de "producción". ¿Qué hacemos?

23. Crear un repositorio llamado `hola-mundo` en GitHub. Poner al menos cinco ejemplos de programas *"hello world"* en distintos lenguajes. Realizar un *commit* por cada ejemplo y agregar un `README.md`. En el `README` linkear cada ejemplo con su archivo correspondiente.

24. Agregar un colaborador al repositorio del ejercicio anterior y pedirle que suba algún ejemplo que todavía no tenemos. El colaborador debe realizar un *commit* agregando un nuevo archivo (el programa de ejemplo) y modificando el `README`.

25. Realizar un *fork* de algún repositorio `hola-mundo` de algún compañero que no nos dió acceso al mismo. Realizar un *pull request* proponiendo un nuevo ejemplo.

26. Estamos trabajando en la rama `new-feature`, estuvimos editando archivos pero de repente necesitamos volver a `master` y aplicar un parche de último momento a nuestra web. ¿Qué comandos de Git ejecutamos?

27. Estoy por hacer un *commit* con tres archivos: `index.js`, `package.json` y `.env`. Me arrepiento de los cambios en `.env` y decido hacer el *commit* pero dejando `.env` tal como estaba en el último *commit*. Los archivos ya están en el *index*. ¿Qué comandos ejecuto?

28. Crear un nuevo repositorio y dentro del mismo dos ramas más además de `master`. Crear un archivo diferente en cada rama y realizar un *commit* en cada una.

29. Indicar verdadero o falso.
    (a) Podemos usar comodines con `git add`. ____
    (b) Git guarda los cambios que hacemos a los archivos automáticamente. ____
    (c) El comando `git pull` sirve para que el repositorio en el que estamos trabajando se sincronize con el repositorio remoto. ____
    (d) Una línea en el `.gitignore` que diga `!a.out` significa que no estamos rastreando el archivo `a.out`. ____
    (e) *Staging area* e *index* son sinónimos. ____
    (f) Cuándo editamos un archivo que ya preparamos los cambios se reflejan automáticamente en el área de preparación. ____
    (g) La única manera de obtener un repositorio de Git es con `git init`. ____
    (h) Si dos personas trabajan en la misma rama del mismo remoto pero en distintos archivos es imposible generar un conflicto de *merge*. ____
    (i) Git no me deja cambiar de rama con el árbol de trabajo (*working tree*) modificado. ____
    (j) Los *forks* y los PRs son características de GitHub, no de Git. ____
    (k) Para colaborar en un proyecto al que no tenemos acceso de escritura podemos hacer un *pull request* y luego un *fork*. ____

30. ¿Qué comando sirve para pasar archivos del *staging area* al repositorio?
    ☐ `git add`
    ☐ `git stage`
    ☐ `git commit`
    ☐ `git restore`

31. El comando para mostrar la diferencia entre lo que está preparado (*staged*) y lo que está en el directorio de trabajo es
    ☐ `git diff`
    ☐ `git diff --staged`
    ☐ `git status -s`
    ☐ `git log --stat`

32. ¿Cuál de los siguientes comandos sirve para cambiar a una rama llamada `feature`?
    - ☐ `git switch feature`
    - ☐ `git checkout feature`
    - ☐ `git checkout -b feature`
    - ☐ Todas son correctas

33. ¿Qué comando usamos para bajar un repositorio de GitHub a nuestra computadora?
    - ☐ `git pull https://github.com/user/repo`
    - ☐ `git clone https://github.com/user/repo`
    - ☐ `git push https://github.com/user/repo`
    - ☐ `git download https://github.com/user/repo`

34. Los archivos que van a entrar en el próximo *commit* están en
    - ☐ cualquier lugar.
    - ☐ el directorio de trabajo.
    - ☐ el repositorio.
    - ☐ el *staging area*.

35. ¿Qué comando sirve para renombrar una rama en Git?
    - ☐ `git remote rm`
    - ☐ `git branch -D`
    - ☐ `git branch -m`
    - ☐ `git rebase`

36. ¿Qué comando sirve para mostrar el parche de cada *commit*?
    - ☐ `git log -p`
    - ☐ `git log`
    - ☐ `git remote -v`
    - ☐ `git branch`

37. ¿Qué es lo que hace el comando `git clone`?
    - ☐ Crea una copia local de un repositorio.
    - ☐ Crea un directorio de trabajo.
    - ☐ Crea una nueva rama.
    - ☐ Tanto la opción 1 como la 2.

38. ¿Qué comando sirve para quitar un archivo del *staging area*?
    - ☐ `git restore archivo`
    - ☐ `git restore --staged archivo`
    - ☐ `git rm archivo`
    - ☐ `git unstage archivo`

39. Estamos en la rama `feature` y recién hicimos un *commit*. Ahora queremos volver a `master` y fusionar ambas ramas. ¿Qué comando usamos?

    - ☐ `git checkout master; git feature merge`
    - ☐ `git checkout -b master`
    - ☐ `git merge feature`
    - ☐ `git switch master; git merge feature`

40. ¿Qué comando sirve para ver un registro de los parches introducidos por cada *commit* en el archivo `index.html`?

    - ☐ `git log -p index.html`
    - ☐ `git log --stat index.html`
    - ☐ `git log -p`
    - ☐ `git diff index.html`