1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas):
   - ¿Qué es GitHub?

   Es un servidor de repositorios de archivos que utiliza el control de versiones de Git.

   - ¿Cómo crear un repositorio en GitHub?

   Se inicia sesión en GitHub, hacemos clic en “Create new…”, luego en “New repository”. Ahí le asignamos un nombre, si queremos podemos agregar una descripción, indicamos si será público o privado, si deseamos que se inicialice con un archivo README.txt y también, opcionalmente, podemos indicar un template .gitignore (un archivo de texto que tiene los patrones de archivos o directorios de los que no se va a mantener un control de versión) y la licencia de nuestro repositorio (que se puede y que no se puede hacer con su contenido).

   - ¿Cómo crear una rama en Git?

   ```
   git branch “nombre_de_la_rama”
   ```

   - ¿Cómo cambiar a una rama en Git?

   ```
   git checkout nombre_de_la_rama
   ```

   - ¿Cómo fusionar ramas en Git?

   ```
   git checkout rama_destino
   git merge rama_a_fusionar
   ```

   - ¿Cómo crear un commit en Git?

   ```
   git add .
   git commit -m “Un comentario”
   ```

   - ¿Cómo enviar un commit a GitHub?

   ```
   git push origin nombre_de_la_rama
   ```

   - ¿Qué es un repositorio remoto?

   Un repositorio ubicado en un servidor de repositorios como GitHub.

   - ¿Cómo agregar un repositorio remoto a Git?

   ```
   git remote add nombre_remoto url_del_repositorio_remoto
   ```

- ¿Cómo empujar cambios a un repositorio remoto?

```
git push nombre_remoto nombre_rama_pushear
```

- ¿Cómo tirar de cambios de un repositorio remoto?

```
git pull nombre_remoto nombre_rama_pullear
```

- ¿Qué es un fork de repositorio?

Es una copia personal de un repositorio que no nos pertenece.

- ¿Cómo crear un fork de un repositorio?

Ingresamos a GitHub, accedemos al repositorio que queremos copiar, hacemos, hacemos clic en Fork y seguimos los pasos.

- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

Una vez realizados mis cambios y subidos a mi repositorio voy a poder verlo en el GitHub con un botón que se habilita que dice Compare & pull request, hacemos clic sobre este y se abre la pantalla para poder crearlo, seleccionamos sobre que repositorio queremos hacer el PR (por defecto el repositorio sobre el que se realizó el fork), agregaremos un título claro, una descripción de los cambios realizados y hacer clic en Create pull request.

- ¿Cómo aceptar una solicitud de extracción?

Entramos a GitHub, en el repositorio veremos el tab Pull requests con un número que indica los PRs pendientes, hacemos clic en esta y si tenemos permiso de escribir en el repositorio se nos habilitará el botón Merge pull request con el cual, en caso de querer incorporar los cambios, podemos hacer el merge al repositorio. Al hacer clic nos dejará incluir un comentario y debemos hacer clic en Confirm merge.

En caso de no querer incorporar los cambios del PR al repositorio simplemente podemos indicar el motivo del rechazo y hacer clic en el botón Close pull request.

- ¿Qué es una etiqueta en Git?

Es un punto que no cambia a lo largo del tiempo, a diferencia de los braches, y están asociados a un commit específico. Suelen utilizarse

para demarcar puntos importantes y que posteriormente pueden ser referenciados rápidamente por esta etiqueta.

- ¿Cómo crear una etiqueta en Git?

```
git tag nombre_del_tag
```

- ¿Cómo enviar una etiqueta a GitHub?

```
git push nombre_repositorio_remoto nombre_del_tag
```

- ¿Qué es un historial de Git?

Es un histórico de los los commits realizados en un repositorio y sirven para analizar y llevar un seguimiento de los cambios realizados.

- ¿Cómo ver el historial de Git?

Para ver el listado de commits: `git log`

Para ver los commits sobre una rama: `git log nombre_de_la_rama`

Para ver los commits sibre un archivo: `git log archivo.extension`

- ¿Cómo buscar en el historial de Git?

Podemos buscar por:

Comentario de commits: `git log --grep="texto_a_buscar"`

Autor: `git log --author="nombre_del_autor"`

Fechas: `git log --since="yyyy-MM-dd" --until="yyyy-MM-dd"`

Texto en archivo: `git log -S "texto_buscado"`

RegEx en archivo: `git log -G "regex"`

Estas formas de búsqueda pueden ser combinadas.

- ¿Cómo borrar el historial de Git?

```
git reset --modo_de_reset HEAD^
git reset --modo_de_reset HEAD^N
git reset --modo_de_reset hash_del_commit
```

- ¿Qué es un repositorio privado en GitHub?

Es un repositorio cuyo contenido solo es accesible para usuarios específicos.

- ¿Cómo crear un repositorio privado en GitHub?

Siguiendo los pasos establecidos en el segundo punto de esta actividad indicando aquí que es un repositorio privado.

- ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Iniciar sesión en GitHub, clic sobre nuestra foto de perfil, clic en "Settings", clic en "Repositories", localizamos el repositorio para el que queremos colaboración, clic en "X collaborators" (dónde X es la cantidad de colaboradores que ya tengamos), clic en "Invite a collaborator", buscamos la persona a invitar, clic sobre esta y por último clic en "Add XXX to this repositorie" (dónde XXX es el nombre del colaborador). Ya solo resta que el invitado acepte la invitación para que pueda clonar el repositorio y realizar cambios.

- ¿Qué es un repositorio público en GitHub?

Es un repositorio cuyo contenido es accesible y puede hacer fork para cualquier usuario que lo encuentre. Es comúnmente utilizado en proyectos open source.

- ¿Cómo crear un repositorio público en GitHub?

Siguiendo los pasos establecidos en el segundo punto de esta actividad indicando aquí que es un repositorio público.

- ¿Cómo compartir un repositorio público en GitHub?

Podemos compartir el enlace al repositorio, promocionarlo, o invitar a colaboradores específicos como en el caso de los repositorios privados.

2) Realizar la siguiente actividad:
   - Crear un repositorio.
     - Dale un nombre al repositorio.
     - Elije el repositorio sea público.
     - Inicializa el repositorio con un archivo.
   - Agregando un Archivo
     - Crea un archivo simple, por ejemplo, "mi-archivo.txt".
     - Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.
     - Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

- Creando Branchs
  - Crear una Branch
  - Realizar cambios o agregar un archivo
  - Subir la Branch

https://github.com/Casqui/Programacion_1_TP_02_Actividad_02

3) Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub

- Ve a GitHub e inicia sesión en tu cuenta.
- Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.
- Asigna un nombre al repositorio, por ejemplo, conflict-exercise.
- Opcionalmente, añade una descripción.
- Marca la opción "Initialize this repository with a README".
- Haz clic en "Create repository".

Paso 2: Clonar el repositorio a tu máquina local

- Copia la URL del repositorio (usualmente algo como https://github.com/tuusuario/conflict-exercise.git).
- Abre la terminal o línea de comandos en tu máquina.
- Clona el repositorio usando el comando:
  ```
  git clone https://github.com/tuusuario/conflict-exercise.git
  ```
- Entra en el directorio del repositorio:
  ```
  cd conflict-exercise
  ```

Paso 3: Crear una nueva rama y editar un archivo

- Crea una nueva rama llamada feature-branch:
  ```
  git checkout -b feature-branch
  ```
- Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo: Este es un cambio en la feature branch.
- Guarda los cambios y haz un commit:
  ```
  git add README.md
  git commit -m "Added a line in feature-branch"
  ```

Paso 4: Volver a la rama principal y editar el mismo archivo

- Cambia de vuelta a la rama principal (main):
  ```
  git checkout main
  ```
- Edita el archivo README.md de nuevo, añadiendo una línea diferente: Este es un cambio en la main branch.
- Guarda los cambios y haz un commit:

```
git add README.md
git commit -m "Added a line in main branch"
```

Paso 5: Hacer un merge y generar un conflicto

- Intenta hacer un merge de la feature-branch en la rama main:
  ```
  git merge feature-branch
  ```
- Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

Paso 6: Resolver el conflicto

- Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:
  ```
  <<<<<<< HEAD
  Este es un cambio en la main branch.
  =======
  Este es un cambio en la feature branch.
  >>>>>>> feature-branch
  ```
- Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.
- Edita el archivo para resolver el conflicto y guarda los cambios(Se debe borrar lo marcado en verde en el archivo donde estes solucionando el conflicto. Y se debe borrar la parte del texto que no se quiera dejar).
- Añade el archivo resuelto y completa el merge:
  ```
  git add README.md
  git commit -m "Resolved merge conflict"
  ```

Paso 7: Subir los cambios a GitHub

- Sube los cambios de la rama main al repositorio remoto en GitHub:
  ```
  git push origin main
  ```
- También sube la feature-branch si deseas:
  ```
  git push origin feature-branch
  ```

Paso 8: Verificar en GitHub

- Ve a tu repositorio en GitHub y revisa el archivo README.md para confirmar que los cambios se han subido correctamente.
- Puedes revisar el historial de commits para ver el conflicto y su resolución.

https://github.com/Casqui/OC_P1_TP2_A3