PROGRAMACIÓN I

Práctico 2: Git y GitHub

Resoluciones:

1) ▸ **¿Qué es GitHub?**

GitHub es una plataforma (en la nube) donde se pueden guardar proyectos usando Git (sistema de control de versiones, de código abierto) de manera eficiente . Este puede servir para trabajar en equipo, revisar los cambios del proyecto y proponer mejoras de este a través de comandos; además de mantener un historial y seguimiento del mismo.

▸ **¿Como crear un repositorio en GitHub?**

- Iniciar sesión en Github
- Hacer clic en New en la parte superior a la izquierda
- Poner un nombre en el repositorio
- Elegir que sea público o privado.
- (Opcional) Activar la opción “Initialize with a README”.
- Hacer clic en "Create repository".

▸ **¿Cómo crear una rama en Git?**

- Abrir la terminal y escribir: `git branch nombre-de-la-rama`

▸ **¿Cómo cambiar a una rama en Git?**

- Abrir la terminal e ir al proyecto
- Usar el comando: `git checkout nombre-de-la-rama`

▸ **¿Cómo fusionar ramas en Git?**

- Cambiarse a la rama y aplicar los cambios: `git checkout main`.
- Ejecutar : `git merge nombre-de-la-rama-a-fusionar`.
- Guardar los cambios: `git add archivo`

¿Cómo crear un commit en Git?

- Hacer cambios en los archivos
- Agregarlos al area de preparacion: `git add .` o `git add nombre-archivo`
- Ejecutar : `git commit -m "Descripción de los cambios"`

¿Cómo enviar un commit a GitHub?

- Tener un repositorio remoto (configurado)
- Usar el comando: `git push origin nombre-de-la-rama`

¿Qué es un repositorio remoto?

Es una copia del proyecto que está alojada en la nube. Este deja compartir y colaborar el proyecto entre diferentes personas a distancia.

¿Cómo agregar un repositorio remoto a Git?

- Copiar el URL del repositorio de GitHub
- Escribir en terminal: `git remote add origin https://github.com/usuario/repositorio.git`
- Verificar : `git remote -v`

¿Cómo empujar cambios a un repositorio remoto?

- Hacer el commit de los cambios locales
- Usar `git push origin nombre-de-la-rama`

¿Cómo tirar de cambios de un repositorio remoto?

- Estar en la rama correcta, y poner: `git checkout nombre-de-la-rama`
- Ejecutar: `git pull origin nombre-de-la-rama`

¿Qué es un fork de repositorio?

Es una copia de otro repositorio, pero dentro de tu cuenta. Sirve para hacer cambios sin afectar el proyecto original.

¿Cómo crear un fork de un repositorio?

- Entrar al repositorio original
- Hacer clic en Fork
- Elegir la cuenta y crear una copia del repositorio

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

- Subir los cambios al repositorio
- Ir a Github y hacer clic en “Compare & pull request”
- Completar un mensaje explicando los cambios
- Hacer clic en Create pull request

¿Cómo aceptar una solicitud de extracción?

- Ir al repositorio original.
- Ir a la pestaña Pull requests
- Seleccionar el PR recibido
- Revisar cambios
- Hacer clic en Merge pull request, y después en Confirm merge

¿Qué es una etiqueta en Git?

Es una marca que se pone en un commit específico para señalar una versión del proyecto

¿Cómo crear una etiqueta en Git?

Estar en el commit deseado, y escribir: `git tag nombre-etiqueta`

¿Cómo enviar una etiqueta a GitHub?

- Subir una sola etiqueta: `git push origin v1.0`
- Subir todas las etiquetas: `git push origin --tags`

¿Qué es un historial de Git?

Es el registro completo de todos los commits realizados en el repositorio, este muestra el autor, fecha y mensaje.

¿Cómo ver el historial de Git?

- Para ver los commits: `git log`

¿Cómo buscar en el historial de Git?

- Usar: `git log --grep="palabra clave"`

¿Cómo borrar el historial de Git?

- Eliminar el repositorios local de Git: `rm -rf .git`
- Inicializar : `git init`
- Subirlo como un proyecto

¿Qué es un repositorio privado en GitHub?

Es un repositorio solo visible para uno mismo y las personas que se inviten. Es ideal para trabajos individuales

¿Cómo crear un repositorio privado en GitHub?

Al crear un repositorio hay que seleccionar la opción Private antes de hacer clic en Create repository

¿Qué es un repositorio público en GitHub?

Es un proyecto visible para cualquiera. Puede ser clonado pero no todos pueden modificarlo

¿Cómo crear un repositorio público en GitHub?

De igual manera que hemos hecho con el privado, pero esta vez seleccionando la opción Public...

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

- Entrar al repositorio e ir a Settings
- Luego ir a Collaborators e invitar a los usuarios usando su nombre en Github

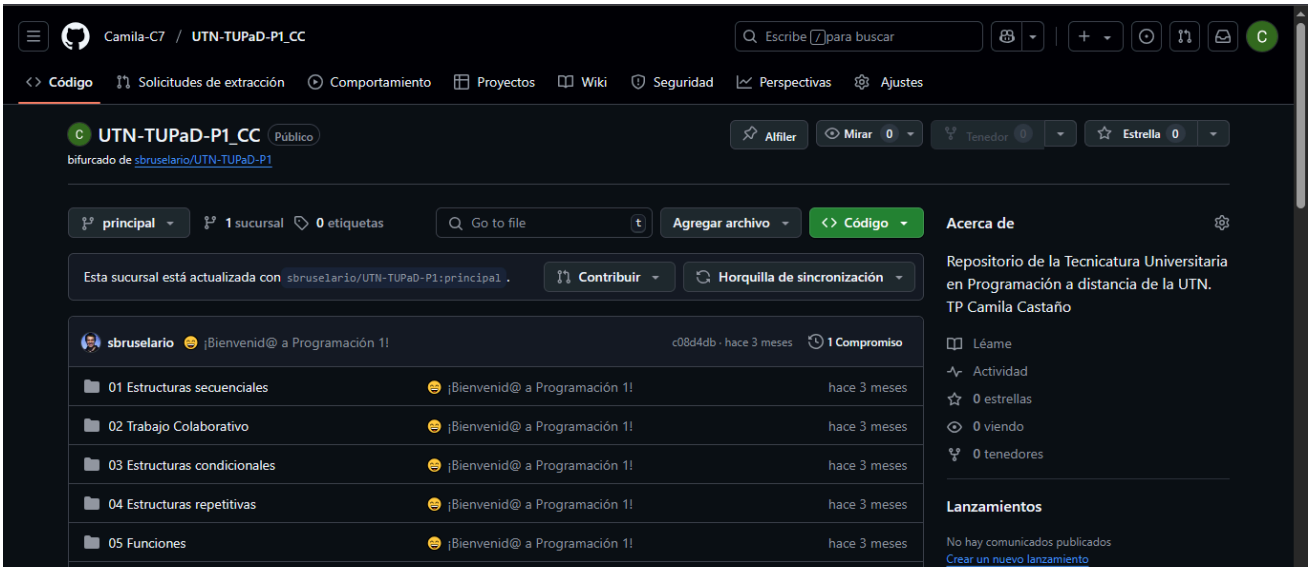
¿Cómo compartir un repositorio público en GitHub?

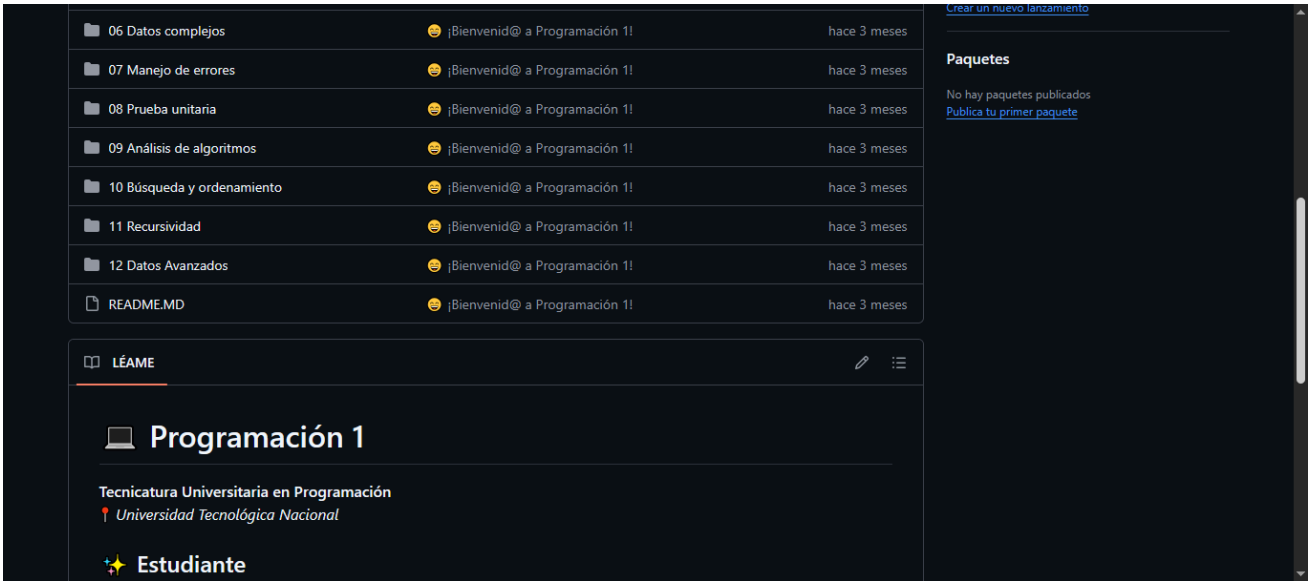
- Entrar al repositorio, copiar la URL del navegador
- Pegar la URL donde queramos compartirlo

Anexos de GitHub:

Panel con fork

Capturas de Fork:





2) Realizar la siguiente actividad:

• Crear un repositorio.
o Dale un nombre al repositorio.
o Elije el repositorio sea público.
o Inicializa el repositorio con un archivo.

• Agregando un Archivo
o Crea un archivo simple, por ejemplo, "mi-archivo.txt".
o Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.
o Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

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

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.