# Tutorial:

**Cómo presentar los trabajos prácticos**

---

**1. Crear el ambiente**

Descargar e instalar GitHub Desktop:

https://github.com/apps/desktop

Esta herramienta te permite gestionar los proyectos desde una interfaz gráfica.

También instala los archivos necesarios para que se puedan usar desde Visual Studio Code.

---

**2. Clonar (descargar el repositorio a la PC)**

Ir a la página:

https://github.com/AlejandroDiBattista/TUP-25-p3

Pulsar el botón verde y elegir `Open with GitHub Desktop`.

Si está correctamente instalado (algunas veces requiere reiniciar la máquina después de instalar), automáticamente abrirá y copiará el repositorio a tu disco (pedirá una autorización la primera vez).

Alternativamente, se puede hacer al revés:
Abrir GitHub Desktop, elegir del menú `File` la opción `Clone Repository` e ingresar la siguiente URL:

https://github.com/AlejandroDiBattista/TUP-25-p3.git

En ambos casos, fíjate detenidamente dónde guarda los archivos. (Normalmente sugiere \Documents\GitHub).

A todos los fines prácticos, tendrás un directorio con todo el material: las clases, los enunciados e incluso todos los trabajos prácticos de tus compañeros.

---

**3. Crear una rama para trabajar**

Las ramas son como una copia temporal de todos los archivos (en realidad solo copia los archivos que se modifican, así que no ocupan casi nada de espacio).

Ve al menú `Branch` y elige `New Branch...` e ingresa el nombre.

Sugiero que sea `<Legajo>-TP1`.

Ahora tendrás dos ramas (o copias del repositorio). Puedes pasar de una a otra usando la opción `Current Branch` en la pantalla. Si la despliegas, podrás cambiar de una rama a otra.

---

### 4. Abrir el repositorio para trabajar

Si vas al menú `Repository`, puedes elegir `Open in Visual Studio Code`.
Alternativamente, puedes abrir VS Code y dentro de él ir a `Archivo` y luego a `Abrir Carpeta...` para seleccionar el repositorio.

---

### 5. Programar el trabajo práctico

Ahora sí puedes trabajar en resolver el trabajo práctico.
Asegúrate de modificar solo los archivos de tu carpeta del trabajo práctico en cuestión.
La encontrarás dentro de la carpeta `TP`, luego tu `<legajo> - <nombre>` y dentro de esta, en la carpeta `TP1`.

Si después de hacer un cambio quieres volver a la rama principal (`Branch Main`), te pedirá que elijas entre:

- `Leave my Changes` (Dejar los cambios en tu rama)
- `Bring my Changes to Main` (Pasar los cambios a la rama principal)

Debes elegir la primera opción. Los cambios se guardarán para después.

En la rama principal puedes, por ejemplo, traer de nuevo los datos del repositorio para tener todo actualizado
(por ejemplo, las clases, las preguntas, etc.) haciendo `Repository` y luego `Fetch`.

Si vas a `View` y luego a `History`, puedes ver los cambios.

Si vuelves a tu rama eligiéndola en `Current Branch`, puede que no veas los cambios. No te preocupes, están copiados en un borrador.

Elige `Stashed Changes` y luego `Restore` para recuperarlos.

---

### 6. Presentar el trabajo

Una vez que hayas realizado el trabajo (fíjate en las instrucciones y modifica solo los archivos necesarios), te aparecerá una lista con todos los archivos modificados marcados.

Asegúrate de que esté marcado únicamente el archivo con la respuesta al ejercicio y confirma los cambios.

Luego, escribe una descripción y pulsa `Commit to <NombreBranch>`. Esto le indica al sistema que guarde los cambios
que realizaste en una "instantánea" de los archivos, es decir, un punto al que puedes volver si algo salió mal.
(Un detalle: podrías ir haciendo commits cada vez que completes una parte del trabajo para asegurarte de no perder nada).

Esto aún no envía los datos a la nube, solo marca los puntos a los que podrías regresar si hay un error.

Para que los cambios suban a GitHub, primero debes publicarlos haciendo `Repository` y luego `Push`.

Ahora estás listo para pedir que tus cambios se entreguen. Esto se llama `Pull Request` y significa que me vas a pedir que incorpore tus cambios al repositorio central. Este repositorio es de mi propiedad, por lo que no puedes subir los datos directamente, sino solicitarme que los incorpore.

Debes ir a `Repository` y elegir `Create Pull Request`. Esto te llevará al sitio web para completar la tarea.

La solicitud ya no forma parte del control del código fuente, sino que es la forma de administrar el proyecto. Por eso, desde la página web debes completar los datos.

Te pedirá un título y una descripción:

- En el título, pon nuevamente `TP1 – Legajo` y agrega tu nombre.
- En la descripción no hace falta que pongas nada, pero podrías usarla para comunicarme algunos problemas que tuviste, etc.

Es importante que pongas correctamente el título para poder identificar la petición.

Listo (o casi), ya publicaste la solución. Ahora aparecerá en mi listado de pendientes para que yo decida si la incorporo.

Puedes confirmar que todo está bien si vas al repositorio en la web, eliges la pestaña `Pull Requests` y verificas que el tuyo aparece en la lista.

Si no hay conflicto, es decir, si solo modificaste tus archivos, lo incorporaré directamente.

---

**7) Dejar el ambiente listo para la próxima**

Vuelve a la rama principal (`Branch Main`) y trae los datos nuevos haciendo `Repository` y luego `Fetch`. Así tendrás los últimos datos.

Cuando la solicitud de cambios que realizaste haya sido aceptada, al hacer `Fetch` aparecerán en el repositorio.

---

# ATENCIÓN:

**Si tuviste problemas para subir el trabajo...**

Si ya hiciste el trabajo y tuviste problemas para subirlo, puedes hacer una copia del archivo con la solución, borrar la carpeta completamente, realizar los pasos 1) al 4). En el paso 5), simplemente vuelve a copiar el archivo con la solución a tu carpeta, reemplazando el archivo original, y sigue con el paso 6). ¡Y listo!

---

# NOTA:

**Ejecución del ejercicio**

Normalmente, para ejecutar un programa debemos crear un proyecto y correrlo mediante `dotnet run` dentro de la carpeta del proyecto.

En este estado de la materia, creo conveniente trabajar de una forma más simple: ejecutar el archivo directamente sin crear un proyecto.

Para ello, debemos seguir estos pasos:

1. Instalar una utilidad para poder ejecutar "scripts" (pequeños programas para automatizar tareas):

```
dotnet tool install -g dotnet-script
```

2. Luego podrás correr el programa `ejercicio.cs` con:

```
dotnet script ./ejercicio.cs
```

estando dentro de la carpeta del ejercicio.

Siempre se puede ejecutar poniendo la ruta completa, pero en este caso considera que cuando los nombres incluyen espacios en blanco deben estar entre comillas:

```
dotnet script ".\65000 - Di Battista, Alejandro\tp1\ejercicio.cs"
```