

Integración: Git, GitHub y VSC

Ing. Luis Guillermo Molero Suárez



Git y GitHub

¿Qué es Git?

Es un sistema de control de versiones, es distribuido, es decir que múltiples personas pueden trabajar en equipo, es open source y también se adapta a todo tipo de proyectos desde pequeños hasta grandes, además, se pueden fusionar archivos, guarda una línea de tiempo a lo largo de todo el proyecto. Maneja una interfaz tipo Bash. GIT, es el software de control de versiones en el que se basa GitHub

Sitio de descarga: <https://git-scm.com/downloads>

Como instalar Git: <https://www.youtube.com/watch?v=ExdLS6lZaAY>

¿Qué es Github?

A diferencia de Git, Github es un sitio web y un servicio en la nube que ayuda a los desarrolladores a almacenar y administrar su código, al igual que llevar un registro y control de cualquier cambio sobre este código. En otras palabras, es una plataforma de desarrollo colaborativo, o también llamada la red social de los desarrolladores donde se alojan los repositorios, el código se almacena de forma pública pero se puede hacer privado con una cuenta de pago.

La interfaz de GitHub es bastante fácil de usar para el desarrollador novato que quiera aprovechar las ventajas del Git. Sin GitHub, usar un Git generalmente requiere de un poco más de conocimientos de tecnología y uso de una línea de comando (Bash).

Sitio de descarga: <https://desktop.github.com/>

Como instalar Github: <https://www.youtube.com/watch?v=tN6tloweTU>



¿Qué Es una Versión de Control?

Una Versión de Control ayuda a los desarrolladores a llevar un registro y administrar cualquier cambio en el código del proyecto de software. A medida que crece este proyecto, la versión de control se vuelve esencial.

Con la bifurcación, un desarrollador duplica parte del código fuente (llamado repositorio). Este desarrollador, luego puede, de forma segura, hacer cambios a esa parte del código, sin afectar al resto del proyecto.

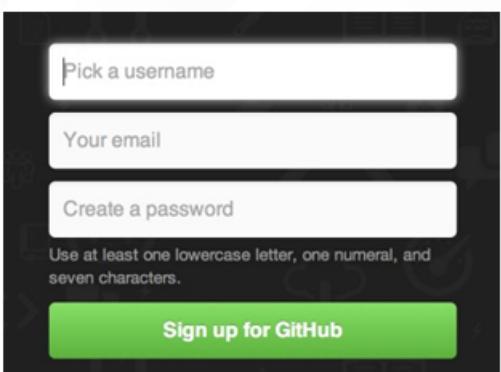
Luego, una vez que el desarrollador logre que su parte del código funcione de forma apropiada, esta persona podría fusionar este código al código fuente principal para hacerlo oficial. Todos estos cambios luego son registrados y pueden ser revertidos si es necesario.

Documentación de Github: <https://docs.github.com/es/github>

Documentación Git: <https://git-scm.com/book/es/v2>

Creación de una cuenta en GitHub

Lo primero que necesitas es una cuenta de usuario gratuita. Simplemente visita <https://github.com>, elige un nombre de usuario que no esté ya en uso, proporciona un correo y una contraseña, y pulsa el botón verde grande “Sign up for GitHub”.



Lo siguiente que verás es la página de precios para planes mejores, pero lo puedes ignorar por el momento. GitHub te enviará un correo para verificar la dirección que les has dado. Confirmar la dirección ahora, es bastante importante (como veremos después).

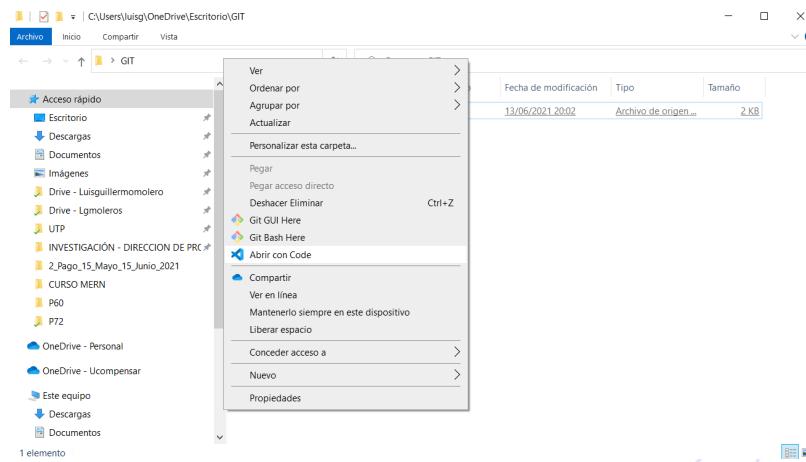
Para ampliar esta información: <https://n9.cl/nqu9>



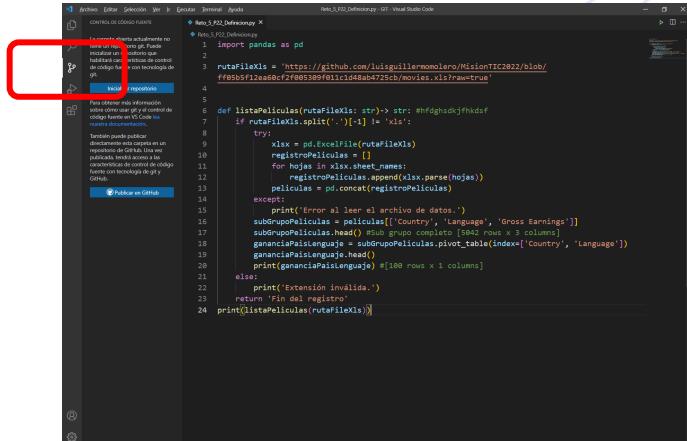
Visual Studio Code

Integración de GIT, GitHub y Visual Studio Code

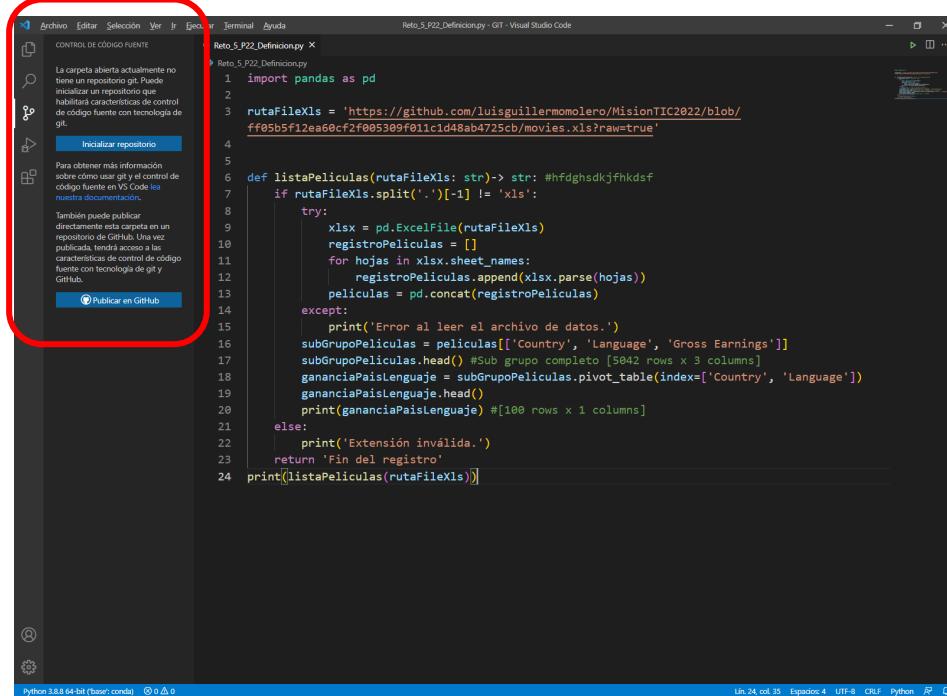
Paso 1: Abrimos nuestro proyecto en VSC, de la forma:



2.- Desde nuestra carpeta de código hacemos clic en el botón “Control de código fuente” de VSC .



Nos aparece la siguiente interacción



```

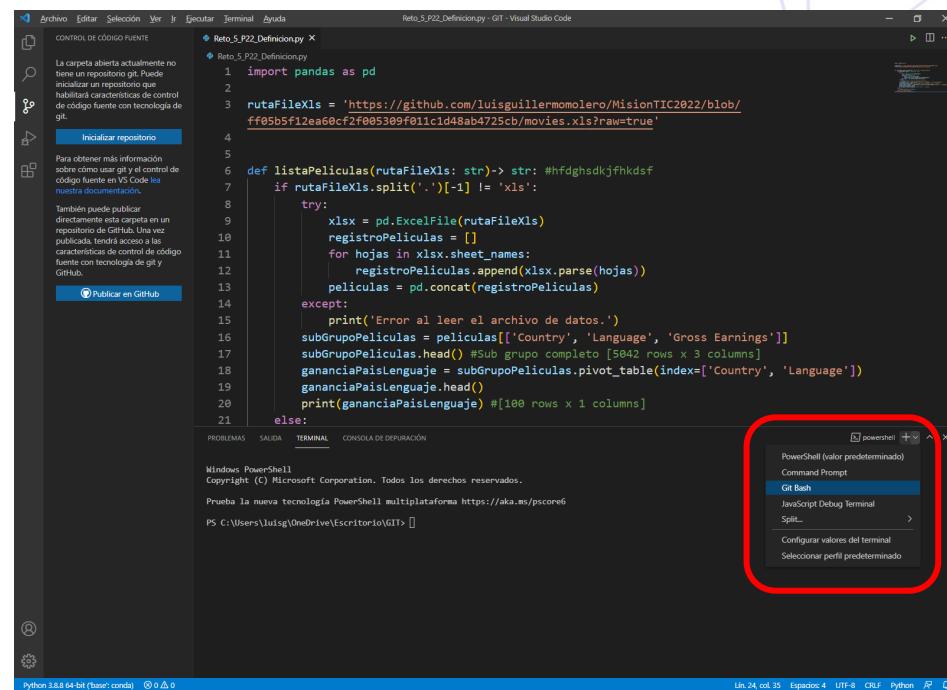
    Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
    CONTROL DE CÓDIGO FUENTE
    La carpeta abierta actualmente no tiene un repositorio git. Puede inicializar un repositorio que habilitará características de control de código fuente con tecnología de git.
    Iniciar repositorio
    Para obtener más información sobre cómo usar git y el control de código fuente en VS Code lea nuestra documentación.
    También puede publicar directamente esta carpeta en un repositorio de Github. Una vez publicada, tendrá acceso a las características de control de código fuente con tecnología de git y Github.
    Publicar en GitHub
    Reto_5_P22_Definicion.py
    1 import pandas as pd
    2
    3 rutaFileXls = 'https://github.com/luisguillermomolero/MisionTIC2022/blob/_ff05b5f12ea60cf2f005309f011c1d48ab4725cb/movies.xls?raw=true'
    4
    5
    6 def listaPeliculas(rutaFileXls: str)-> str: #ffdghsdkjfhhkdsf
    7     if rutaFileXls.split('.')[ -1 ] != 'xls':
    8         try:
    9             xlsx = pd.ExcelFile(rutaFileXls)
    10            registroPeliculas = []
    11            for hojas in xlsx.sheet_names:
    12                registroPeliculas.append(xlsx.parse(hojas))
    13            peliculas = pd.concat(registroPeliculas)
    14        except:
    15            print('Error al leer el archivo de datos.')
    16            subGrupoPeliculas = peliculas[['Country', 'Language', 'Gross Earnings']]
    17            subGrupoPeliculas.head() #Sub grupo completo [5042 rows x 3 columns]
    18            gananciaPaislenguaje = subGrupoPeliculas.pivot_table(index=['Country', 'Language'])
    19            gananciaPaislenguaje.head()
    20            print(gananciaPaislenguaje) #[100 rows x 1 columns]
    21        else:
    22            print('Extensión inválida.')
    23        return 'Fin del registro'
    24    print(listaPeliculas(rutaFileXls))

```

Python 3.8.6 64-bit (base:conda) ⚡ 0 ⚡ 0

Lín. 24, col. 35 Espacio: 4 UTF-8 CR LF Python ⚡ 0

3.- Abrimos la terminal de Git en VSC y escribimos git init



```

    Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
    CONTROL DE CÓDIGO FUENTE
    La carpeta abierta actualmente no tiene un repositorio git. Puede inicializar un repositorio que habilitará características de control de código fuente con tecnología de git.
    Iniciar repositorio
    Para obtener más información sobre cómo usar git y el control de código fuente en VS Code lea nuestra documentación.
    También puede publicar directamente esta carpeta en un repositorio de Github. Una vez publicada, tendrá acceso a las características de control de código fuente con tecnología de git y Github.
    Publicar en GitHub
    Reto_5_P22_Definicion.py
    1 import pandas as pd
    2
    3 rutaFileXls = 'https://github.com/luisguillermomolero/MisionTIC2022/blob/_ff05b5f12ea60cf2f005309f011c1d48ab4725cb/movies.xls?raw=true'
    4
    5
    6 def listaPeliculas(rutaFileXls: str)-> str: #ffdghsdkjfhhkdsf
    7     if rutaFileXls.split('.')[ -1 ] != 'xls':
    8         try:
    9             xlsx = pd.ExcelFile(rutaFileXls)
    10            registroPeliculas = []
    11            for hojas in xlsx.sheet_names:
    12                registroPeliculas.append(xlsx.parse(hojas))
    13            peliculas = pd.concat(registroPeliculas)
    14        except:
    15            print('Error al leer el archivo de datos.')
    16            subGrupoPeliculas = peliculas[['Country', 'Language', 'Gross Earnings']]
    17            subGrupoPeliculas.head() #Sub grupo completo [5042 rows x 3 columns]
    18            gananciaPaislenguaje = subGrupoPeliculas.pivot_table(index=['Country', 'Language'])
    19            gananciaPaislenguaje.head()
    20            print(gananciaPaislenguaje) #[100 rows x 1 columns]
    21        else:
    22            print('Extensión inválida.')
    23        return 'Fin del registro'
    24    print(listaPeliculas(rutaFileXls))

```

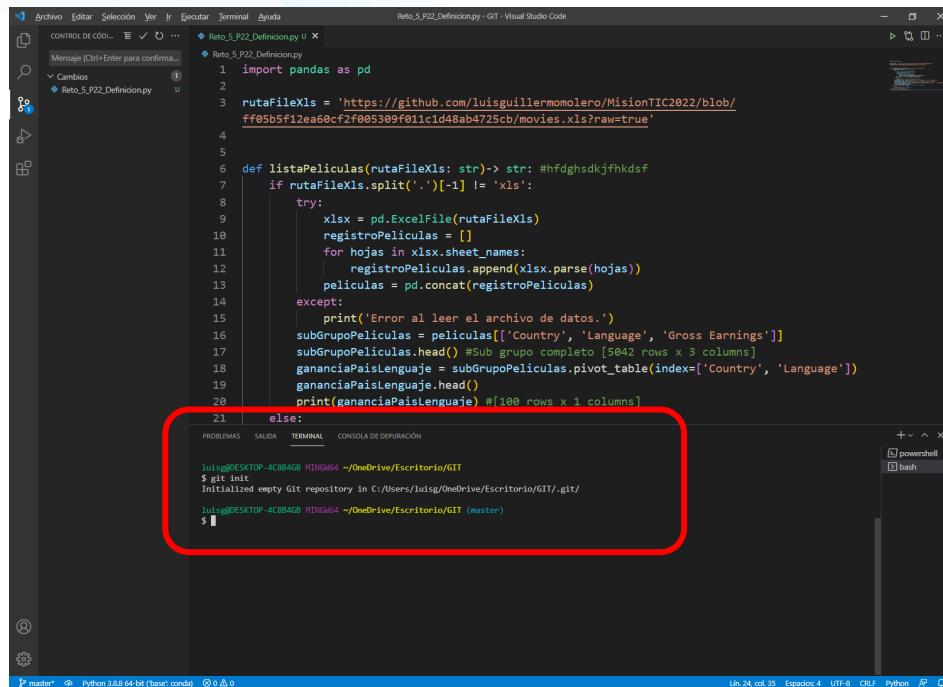
PROBLEMAS SALIDA TERMINAL CONSOLA DE DEPURACIÓN

Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.
Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/powershell
PS C:\Users\luisg\OneDrive\Escritorio\GIT> []

PowerShell (valor predeterminado)
Command Prompt
Git Bash
JavaScript Debug Terminal
Split...
Configurar valores del terminal
Seleccionar perfil predeterminado

Python 3.8.6 64-bit (base:conda) ⚡ 0 ⚡ 0

Lín. 24, col. 35 Espacio: 4 UTF-8 CR LF Python ⚡ 0

```

import pandas as pd

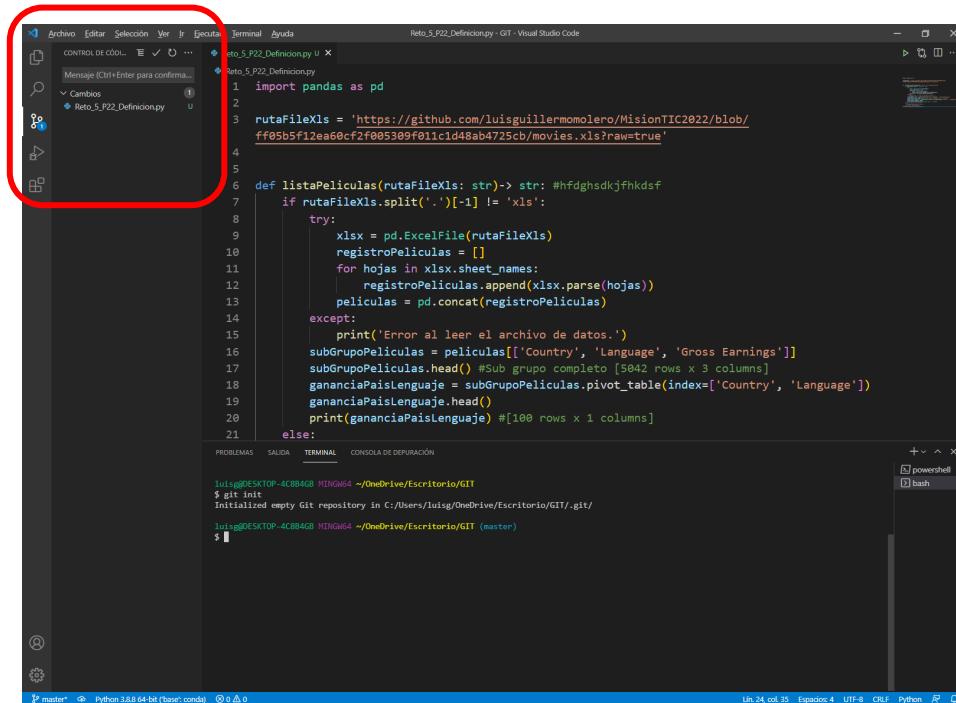
rutaFileXls = 'https://github.com/luisguillermomolero/MisionTIC2022/blob/_ff05b5f12ea60cf2f005309f011cid4ab4725cb/movies.xls?raw=true'

def listaPelículas(rutaFileXls: str) -> str: #hfdfghsdkjfhkdsf
    if rutaFileXls.split('.')[1] != 'xls':
        try:
            xlsx = pd.ExcelFile(rutaFileXls)
            registroPelículas = []
            for hojas in xlsx.sheet_names:
                registroPelículas.append(xlsx.parse(hojas))
            películas = pd.concat(registroPelículas)
        except:
            print('Error al leer el archivo de datos.')
    subGrupoPelículas = películas[['Country', 'Language', 'Gross Earnings']]
    subGrupoPelículas.head() #Sub grupo completo [5042 rows x 3 columns]
    gananciaPaislenguaje = subGrupoPelículas.pivot_table(index=['Country', 'Language'])
    gananciaPaislenguaje.head()
    print(gananciaPaislenguaje) #[100 rows x 1 columns]
else:
    print('Formato de archivo incorrecto.')

luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT
$ git init
Initialized empty Git repository in C:/Users/luisg/OneDrive/Escritorio/GIT/.git/
luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT (master)
$ 

```

Una vez ejecutado el comando, desaparece el área de interacción y ya podemos trabajar con el código fuente, ya que se inicializó un repositorio en local.



```

import pandas as pd

rutaFileXls = 'https://github.com/luisguillermomolero/MisionTIC2022/blob/_ff05b5f12ea60cf2f005309f011cid4ab4725cb/movies.xls?raw=true'

def listaPelículas(rutaFileXls: str) -> str: #hfdfghsdkjfhkdsf
    if rutaFileXls.split('.')[1] != 'xls':
        try:
            xlsx = pd.ExcelFile(rutaFileXls)
            registroPelículas = []
            for hojas in xlsx.sheet_names:
                registroPelículas.append(xlsx.parse(hojas))
            películas = pd.concat(registroPelículas)
        except:
            print('Error al leer el archivo de datos.')
    subGrupoPelículas = películas[['Country', 'Language', 'Gross Earnings']]
    subGrupoPelículas.head() #Sub grupo completo [5042 rows x 3 columns]
    gananciaPaislenguaje = subGrupoPelículas.pivot_table(index=['Country', 'Language'])
    gananciaPaislenguaje.head()
    print(gananciaPaislenguaje) #[100 rows x 1 columns]
else:
    print('Formato de archivo incorrecto.')

luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT
$ git init
Initialized empty Git repository in C:/Users/luisg/OneDrive/Escritorio/GIT/.git/
luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT (master)
$ 

```

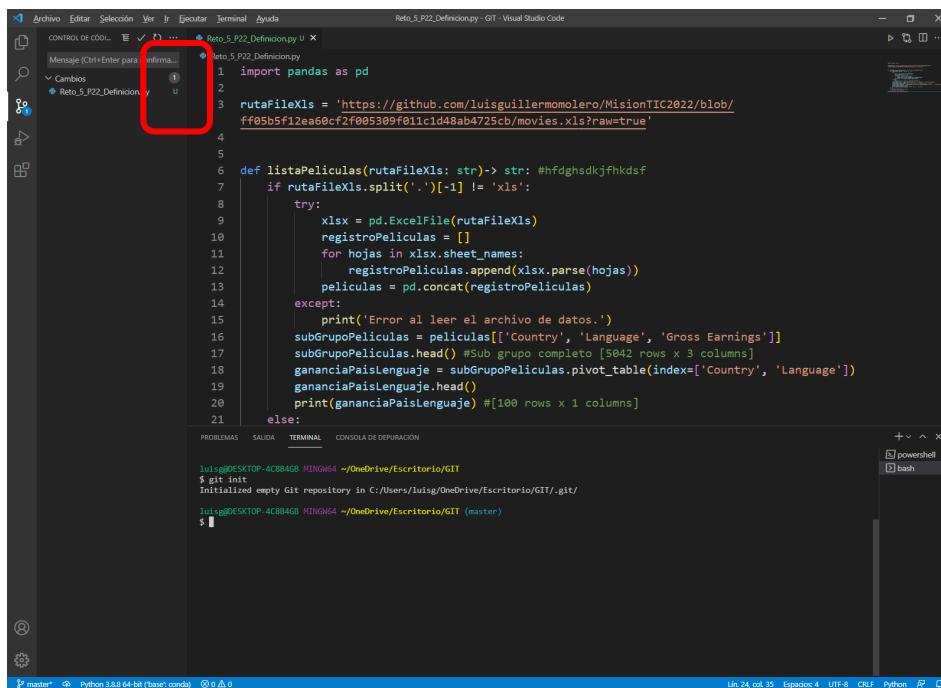


Podemos confirmar que ya está inicializado el repositorio de la siguiente manera:
Nos vamos a la carpeta donde se ubica el proyecto y observaremos una carpeta oculta .git
(Cada vez que queramos ocultar una carpeta colocamos el “.” antes del nombre).



Luego de verificar que tenemos control de cambios en el proyecto, revisemos los siguientes:

En nuestro proyecto, el archivo abierto aparece con el identificador “U” que significa sin seguimiento



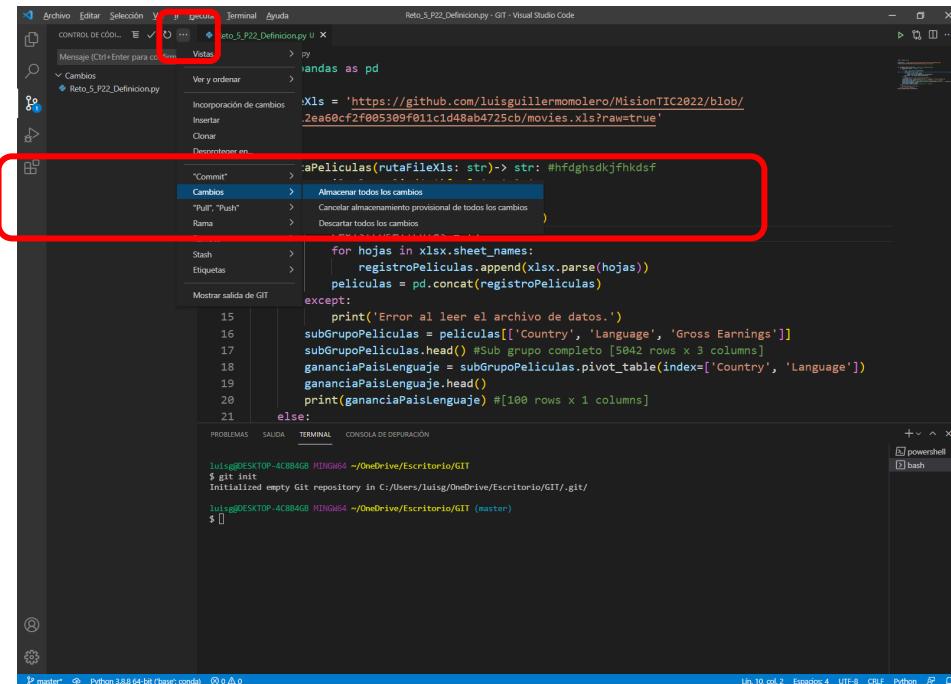
```

import pandas as pd
rutaFileXls = 'https://github.com/luisguillermomolero/MisionTIC2022/blob/ff05b5f12ea60cf2f0085309f011c1d4ab4725cb/movies.xls?raw=true'

def listaPeliculas(rutaFileXls: str) -> str: #hfdfghsdkjfhkdsf
    if rutaFileXls.split('.')[1] != 'xls':
        try:
            xlsx = pd.ExcelFile(rutaFileXls)
            registroPeliculas = []
            for hojas in xlsx.sheet_names:
                registroPeliculas.append(xlsx.parse(hojas))
            peliculas = pd.concat(registroPeliculas)
        except:
            print('Error al leer el archivo de datos.')
    subGrupoPeliculas = peliculas[['Country', 'Language', 'Gross Earnings']]
    subGrupoPeliculas.head() #Sub grupo completo [5042 rows x 3 columns]
    gananciaPaisLenguaje = subGrupoPeliculas.pivot_table(index=['Country', 'Language'])
    gananciaPaisLenguaje.head()
    print(gananciaPaisLenguaje) #[100 rows x 1 columns]
else:

```

4.- Entonces, hacemos clic en “almacenable todos los cambios”

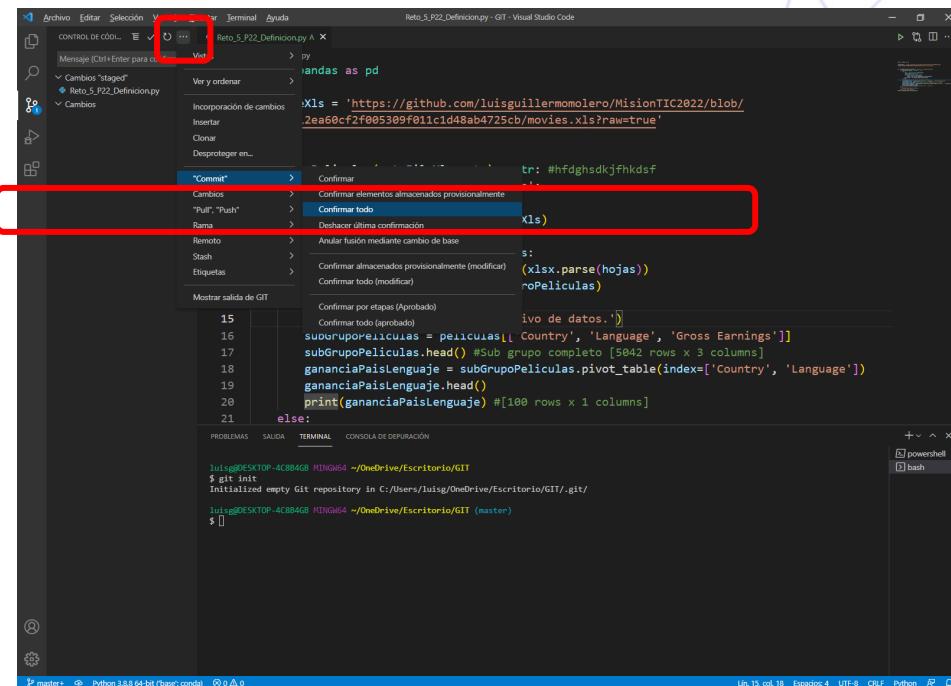


```

    Archivo Editar Selección Vídeo Terminal Ayuda
    CONTROL DE CÓDIGO: Reto_5_P22_Definicion.py x
    Mensaje (Ctrl+Enter para com... Vista
    Cambios > Reto_5_P22_Definicion.py
    Ver y ordenar > Vistas
    Incorporación de cambios
    Insertar
    Clonar
    Desproteger en...
    "Commit" > :pPelículas(rutaFileXls: str) -> str: #hfghsdkjfhkdsf
    Cambios > Almacenar todos los cambios
    "Pull", "Push"
    Rama
    Stash
    Etiquetas
    Mostrar salida de GIT
    PROBLEMAS SALIDA TERMINAL CONSOLA DE DEPURACIÓN
    Luisg@DESKTOP-4CB844B MINGW64 ~/OneDrive/Escritorio/GIT
    $ git init
    Initialized empty Git repository in C:/Users/luisg/OneDrive/Escritorio/GIT/.git/
    Luisg@DESKTOP-4CB844B MINGW64 ~/OneDrive/Escritorio/GIT (master)
    $ []
  
```

Una vez hecho esto, nos aparece el identificador “A” (Add).

4.- Como paso seguido, procedemos a “confirmar todo”

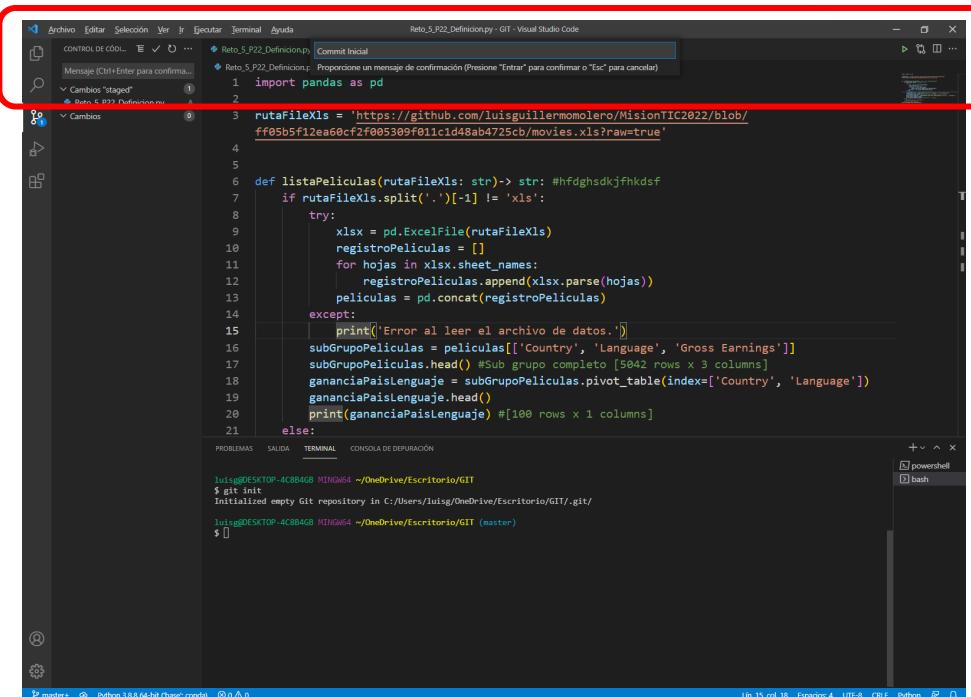


```

    Archivo Editar Selección Vídeo Terminal Ayuda
    CONTROL DE CÓDIGO: Reto_5_P22_Definicion.py A x
    Mensaje (Ctrl+Enter para com... Vista
    Cambios "staged" > Reto_5_P22_Definicion.py
    Cambios > Reto_5_P22_Definicion.py
    Ver y ordenar > Vistas
    Incorporación de cambios
    Insertar
    Clonar
    Desproteger en...
    "Commit" > :pPelículas(rutaFileXls: str) -> str: #hfghsdkjfhkdsf
    Cambios > Confirmar todo
    "Pull", "Push"
    Rama
    Remoto
    Stash
    Etiquetas
    Mostrar salida de GIT
    PROBLEMAS SALIDA TERMINAL CONSOLA DE DEPURACIÓN
    Luisg@DESKTOP-4CB844B MINGW64 ~/OneDrive/Escritorio/GIT
    $ git init
    Initialized empty Git repository in C:/Users/luisg/OneDrive/Escritorio/GIT/.git/
    Luisg@DESKTOP-4CB844B MINGW64 ~/OneDrive/Escritorio/GIT (master)
    $ []
  
```



5.- Nos abrirá la “paleta de comandos” y le colocamos un mensaje para el primer commit (Este mensaje debe hacerse tantas veces hagamos una actualización del proyecto para llevar el control de cambios), que para el caso de este ejemplo es: “Commit Inicial”.



```
import pandas as pd
rutaFileXls = 'https://github.com/luisguillermomolero/MisionTIC2022/blob/_ff05b5f12ea60fcf005309f011c1d48ab4725cb/movies.xls?raw=true'

def listaPeliculas(rutafilexls: str) -> str: #hfdfghsdfkjfhkdsf
    if rutafilexls.split('.')[1] != 'xls':
        try:
            xlsix = pd.ExcelFile(rutafilexls)
            registroPeliculas = []
            for hojas in xlsix.sheet_names:
                registroPeliculas.append(xlsix.parse(hojas))
            peliculas = pd.concat(registroPeliculas)
        except:
            print('Error al leer el archivo de datos.')
            subGrupoPeliculas = peliculas[['Country', 'Language', 'Gross Earnings']]
            subGrupoPeliculas.head() #Sub grupo completo [5042 rows x 3 columns]
            gananciaPaisLenguaje = subGrupoPeliculas.pivot_table(index=['Country', 'Language'])
            gananciaPaisLenguaje.head()
            print(gananciaPaisLenguaje) #[100 rows x 1 columns]
    else:
```

IMPORTANTE:

Si al momento de teclear <ENTER> para confirmar el mensaje del “commit” hecho genera un error de autenticación, introduzca en la terminal GIT las siguientes líneas de comandos:
git config --global user.email "tu_correo_de_GITHUB"
git config --global user.name "tu_usuario_de_GITHUB"

RECUERDA:

esta configuración de GIT, GitHub en VSC debe hacerse luego de haber instalado GIT en tu Pc y creado tu cuenta en GITHUB.

Paso seguido, veremos que no hay cambios que guardar en nuestro repositorio

```

import pandas as pd

rutaFileXls = 'https://github.com/luisguillermomolero/MisionTIC2022/blob/ff05b5f12ea60cf2f005309f011c1d48ab4725cb/movies.xls?raw=true'

def listaPelículas(rutaFileXls: str)-> str: #ffdfghsdkjfhkdsf
    if rutaFileXls.split('.')[ -1 ] != 'xls':
        try:
            xlsx = pd.ExcelFile(rutaFileXls)
            registroPelículas = []
            for hojas in xlsx.sheet_names:
                registroPelículas.append(xlsx.parse(hojas))
            películas = pd.concat(registroPelículas)
        except:
            print('Error al leer el archivo de datos.')
            subGrupoPelículas = películas[['Country', 'Language', 'Gross Earnings']]
            subGrupoPelículas.head() #Sub grupo completo [5042 rows x 3 columns]
            gananciaPaislenguaje = subGrupoPelículas.pivot_table(index=['Country', 'Language'])
            gananciaPaislenguaje.head()
            print(gananciaPaislenguaje) #[100 rows x 1 columns]
    else:

```

Una vez que hallamos modificado nuestro archivo del proyecto, aparecerá el identificador “M” en la ventana de “Control de cambios”

```

import pandas as pd

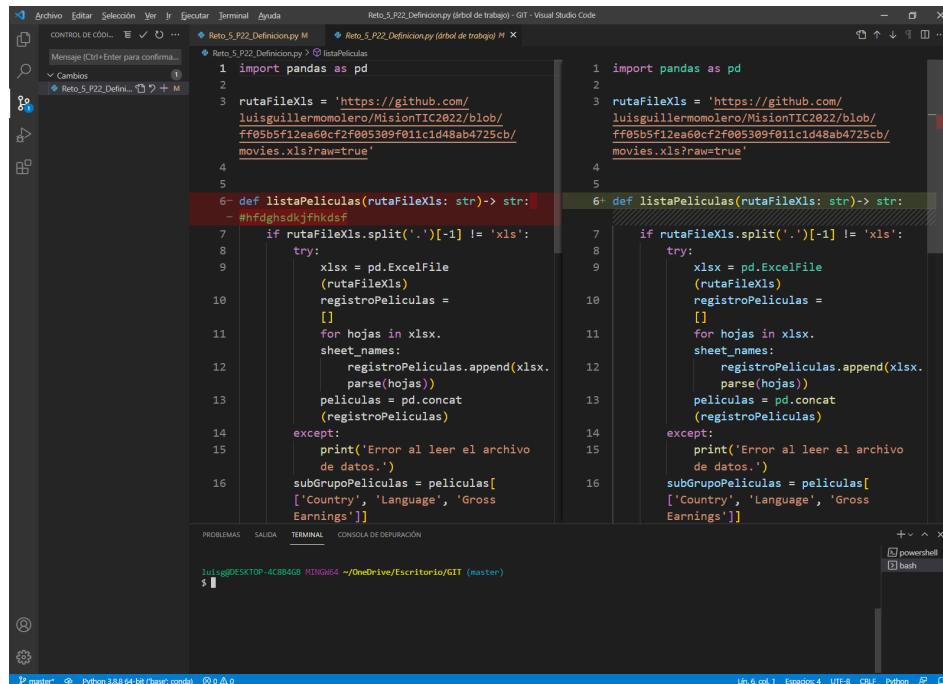
rutaFileXls = 'https://github.com/luisguillermomolero/MisionTIC2022/blob/ff05b5f12ea60cf2f005309f011c1d48ab4725cb/movies.xls?raw=true'

def listaPelículas(rutaFileXls: str)-> str:
    if rutaFileXls.split('.')[ -1 ] != 'xls':
        try:
            xlsx = pd.ExcelFile(rutaFileXls)
            registroPelículas = []
            for hojas in xlsx.sheet_names:
                registroPelículas.append(xlsx.parse(hojas))
            películas = pd.concat(registroPelículas)
        except:
            print('Error al leer el archivo de datos.')
            subGrupoPelículas = películas[['Country', 'Language', 'Gross Earnings']]
            subGrupoPelículas.head() #Sub grupo completo [5042 rows x 3 columns]
            gananciaPaislenguaje = subGrupoPelículas.pivot_table(index=['Country', 'Language'])
            gananciaPaislenguaje.head()
            print(gananciaPaislenguaje) #[100 rows x 1 columns]
    else:

```



En ese sentido, al hacer clic sobre el archivo modificado, aparecerá otra pantalla donde se observan los cambios.

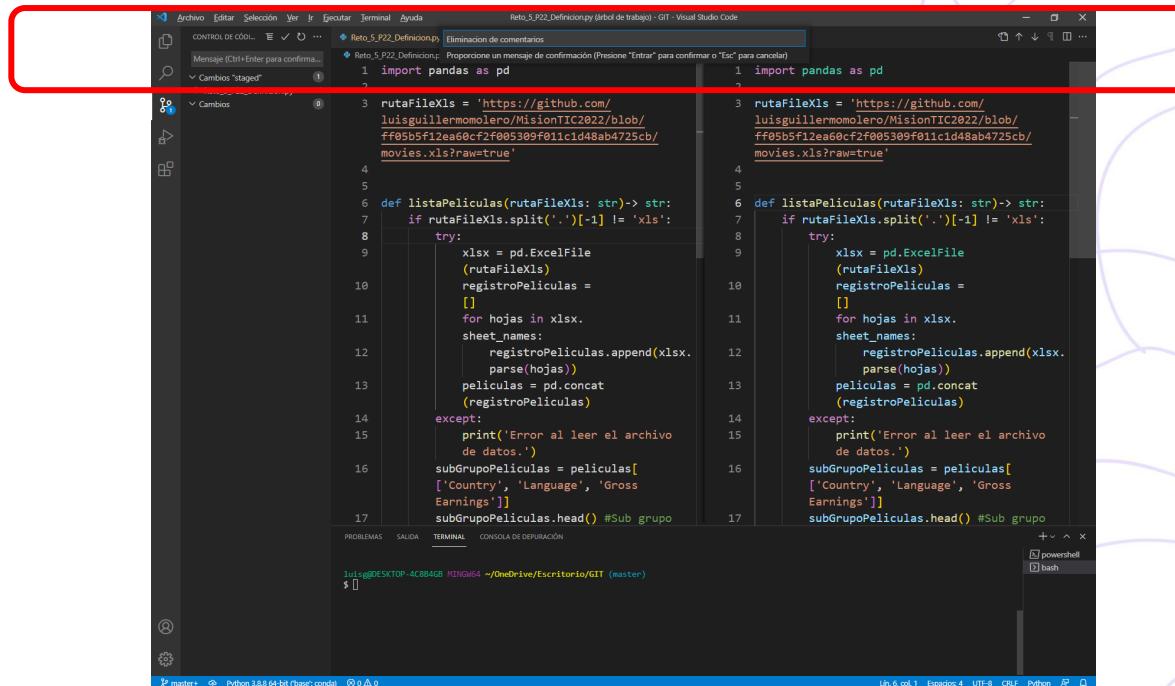


```

    Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
    CONTROL DE CÓDIGO... 🔍 🌐 Reto_5_P22_Definicion.py (árbol de trabajo) - GIT - Visual Studio Code
    Mensaje (Ctrl+Enter para confirmar...) Cambios
    Reto_5_P22_Definicion.py (árbol de trabajo) M
    1 import pandas as pd
    2
    3 rutaFileXls = 'https://github.com/
    luisguillermomolero/MisionTIC2022/blob/
    ff05b5f12ea60cf2f005309f011c1d48ab4725cb/
    movies.xls?raw=true'
    4
    5
    6- def listaPeliculas(rutaFileXls: str)-> str:
    - #ffdghnsdkjfhnksdf
    7-     if rutaFileXls.split('.')[1] != 'xls':
    8-         try:
    9-             xlsx = pd.ExcelFile
    10-                (rutaFileXls)
    11-                registroPeliculas =
    12-                    []
    13-                    for hojas in xlsx.
    sheet_names:
    14-                        registroPeliculas.append(xlsx.
    parse(hojas))
    15-                        peliculas = pd.concat
    16-                            (registroPeliculas)
    except:
    print('Error al leer el archivo
    de datos.')
    subGrupoPeliculas = peliculas[
    ['Country', 'Language', 'Gross
    Earnings']]
    1 import pandas as pd
    2
    3 rutaFileXls = 'https://github.com/
    luisguillermomolero/MisionTIC2022/blob/
    ff05b5f12ea60cf2f005309f011c1d48ab4725cb/
    movies.xls?raw=true'
    4
    5
    6 def listaPeliculas(rutaFileXls: str)-> str:
    7     if rutaFileXls.split('.')[1] != 'xls':
    8         try:
    9             xlsx = pd.ExcelFile
    10                (rutaFileXls)
    11                registroPeliculas =
    12                    []
    13                    for hojas in xlsx.
    sheet_names:
    14                        registroPeliculas.append(xlsx.
    parse(hojas))
    15                        peliculas = pd.concat
    16                            (registroPeliculas)
    except:
    print('Error al leer el archivo
    de datos.')
    subGrupoPeliculas = peliculas[
    ['Country', 'Language', 'Gross
    Earnings']]

```

Una vez observado estos cambios y estar conforme con ellos, nos vamos de nuevo a “almacenar todos los cambios” y luego procedemos a “confirmar todo “de nuevo (pasos anteriores). Para este ejemplo el mensaje será “Eliminación de comentarios”



Eliminación de comentarios

Proporcione un mensaje de confirmación (Presione "Entrar" para confirmar o "Esc" para cancelar)

```

    Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
    CONTROL DE CÓDIGO... 🔍 🌐 Reto_5_P22_Definicion.py (árbol de trabajo) - GIT - Visual Studio Code
    Mensaje (Ctrl+Enter para confirmar...) Cambios
    Reto_5_P22_Definicion.py (árbol de trabajo) M
    1 import pandas as pd
    2
    3 rutaFileXls = 'https://github.com/
    luisguillermomolero/MisionTIC2022/blob/
    ff05b5f12ea60cf2f005309f011c1d48ab4725cb/
    movies.xls?raw=true'
    4
    5
    6- def listaPeliculas(rutaFileXls: str)-> str:
    - #ffdghnsdkjfhnksdf
    7-     if rutaFileXls.split('.')[1] != 'xls':
    8-         try:
    9-             xlsx = pd.ExcelFile
    10-                (rutaFileXls)
    11-                registroPeliculas =
    12-                    []
    13-                    for hojas in xlsx.
    sheet_names:
    14-                        registroPeliculas.append(xlsx.
    parse(hojas))
    15-                        peliculas = pd.concat
    16-                            (registroPeliculas)
    except:
    print('Error al leer el archivo
    de datos.')
    subGrupoPeliculas = peliculas[
    ['Country', 'Language', 'Gross
    Earnings']]
    1 import pandas as pd
    2
    3 rutaFileXls = 'https://github.com/
    luisguillermomolero/MisionTIC2022/blob/
    ff05b5f12ea60cf2f005309f011c1d48ab4725cb/
    movies.xls?raw=true'
    4
    5
    6 def listaPeliculas(rutaFileXls: str)-> str:
    7     if rutaFileXls.split('.')[1] != 'xls':
    8         try:
    9             xlsx = pd.ExcelFile
    10                (rutaFileXls)
    11                registroPeliculas =
    12                    []
    13                    for hojas in xlsx.
    sheet_names:
    14                        registroPeliculas.append(xlsx.
    parse(hojas))
    15                        peliculas = pd.concat
    16                            (registroPeliculas)
    except:
    print('Error al leer el archivo
    de datos.')
    subGrupoPeliculas = peliculas[
    ['Country', 'Language', 'Gross
    Earnings']]

```

Finalmente, confirmamos que no se vean cambios en nuestro “Control de cambios”.

#MisiónTIC2022

RECUERDA:

Todos estos cambios se están haciendo de manera local, ahora vamos a configurar nuestro GITHUB para poder hospedar estos cambios en la nube.

1.- Nos vamos a nuestra cuenta de GITHUB en la web y creamos un “Nuevo Repositorio”

The screenshot shows a GitHub user profile for 'luisguillermomolero'. It displays a circular profile picture, a summary of contributions ('9 contributions in the last year'), and a 'Contribution activity' chart. A red box highlights the top right corner of the screen, where a context menu is open. The menu options include 'New repository', 'Import repository', 'New gist', 'New organization', and 'New project'. The 'New repository' option is highlighted with a blue box.

Le colocamos un nombre, una descripción a nuestro repositorio y hacemos clic en “Create repository”

The screenshot shows the 'Create a new repository' form on GitHub. The 'Repository name' field is filled with 'Repositorio_Prueba_GIT_VSC' and has a green checkmark. The 'Description (optional)' field contains the text 'Repositorio de prueba para configurar control de cambios en VSC'. The 'Public' radio button is selected. At the bottom, a red box highlights the 'Create repository' button. The entire form is set against a background of light purple curved lines.

2.- Una vez creado el repositorio, copiamos las siguientes líneas de código que se encuentra en GitHub, en nuestra terminal de GIT de VSC.



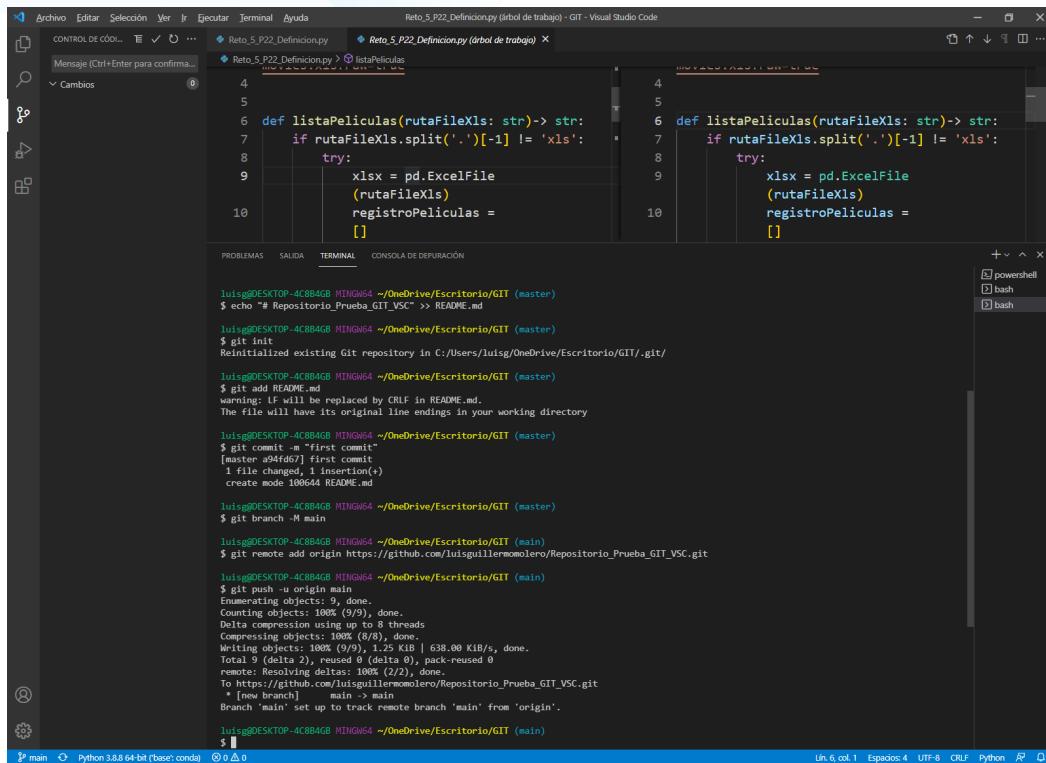
The screenshot shows a GitHub repository page for 'luisguillermomolero / Repositorio_Prueba_GIT_VSC'. The 'Code' tab is selected. A red box highlights the 'Quick setup — if you've done this kind of thing before' section, which contains a command-line script for initializing a new repository:

```
echo "# Repositorio_Prueba_GIT_VSC" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/luisguillermomolero/Repositorio_Prueba_GIT_VSC.git
git push -u origin main
```

Below this, there are sections for pushing an existing repository and importing code from another repository.

```
echo "# Repositorio_Prueba_GIT_VSC" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin
https://github.com/luisguillermomolero/Repositorio_Prueba_GIT_VSC.git
git push -u origin main
```



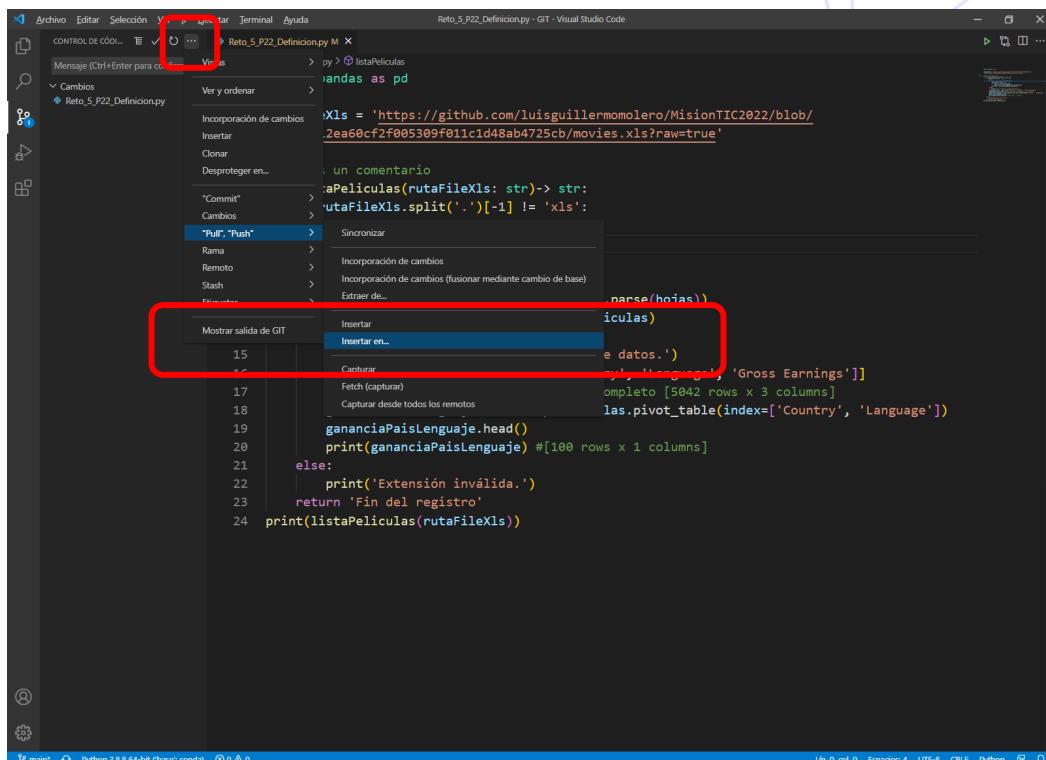
```

def listaPelículas(rutaFileXls: str) -> str:
    if rutaFileXls.split('.')[ -1 ] != 'xls':
        try:
            xlsx = pd.ExcelFile(rutaFileXls)
            registroPelículas = []
        except:
            print('Extensión inválida.')
            return 'Fin del registro'
    else:
        print('Extensión válida.')
        datos = pd.read_excel(rutaFileXls)
        gananciaPaisLenguaje = datos.groupby(['Country', 'Language']).sum()
        print(gananciaPaisLenguaje.head())
        print(gananciaPaisLenguaje #[100 rows x 1 columns])
        else:
            print('Extensión inválida.')
            return 'Fin del registro'
    print(listaPelículas(rutaFileXls))

```

luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT (master)
\$ echo "# Repositorio_Prueba_GIT_VSC" > README.md
\$ git init
Reinitialized existing Git repository in C:/Users/luisg/OneDrive/Escritorio/GIT/.git/
luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT (master)
\$ git add README.md
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory
luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT (master)
\$ git commit -m "first commit"
[master 949fd67] first commit
1 file changed, 1 insertion(+)
create mode 100644 README.md
luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT (main)
\$ git branch -M main
luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT (main)
\$ git remote add origin https://github.com/luisguillermomolero/Repositorio_Prueba_GIT_VSC.git
luisg@DESKTOP-4C8B4GB MINGW64 ~/OneDrive/Escritorio/GIT (main)
\$ git push -u origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compressing: 100% (0/0), done.
Compressing objects: 100% (0/0), done.
Writing objects: 100% (9/9), 1.2k KiB | 638.00 KiB/s, done.
Total 9 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/luisguillermomolero/Repositorio_Prueba_GIT_VSC.git
 * [new branch] main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.

Luego de que eventualmente se realice algún cambio en nuestro código nos vamos a “Insertar en”



Retorno de la ejecución:

```

def listaPelículas(rutaFileXls: str) -> str:
    if rutaFileXls.split('.')[ -1 ] != 'xls':
        try:
            xlsx = pd.ExcelFile(rutaFileXls)
            registroPelículas = []
        except:
            print('Extensión inválida.')
            return 'Fin del registro'
    else:
        print('Extensión válida.')
        datos = pd.read_excel(rutaFileXls)
        gananciaPaisLenguaje = datos.groupby(['Country', 'Language']).sum()
        print(gananciaPaisLenguaje.head())
        print(gananciaPaisLenguaje #[100 rows x 1 columns])
        else:
            print('Extensión inválida.')
            return 'Fin del registro'
    print(listaPelículas(rutaFileXls))

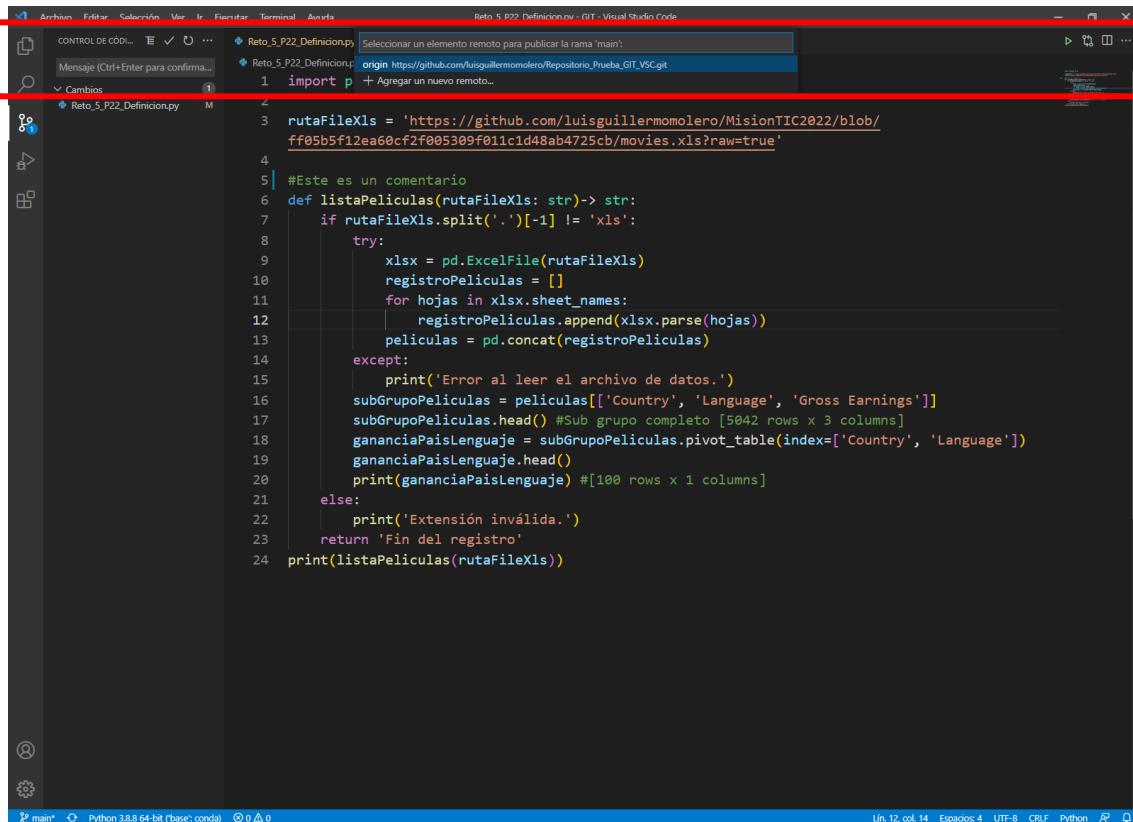
```

Mostrar salida de GIT

Insertar en...



Y en la “paleta de comandos” nos listara los repositorios donde podremos actualizar nuestro proyecto.

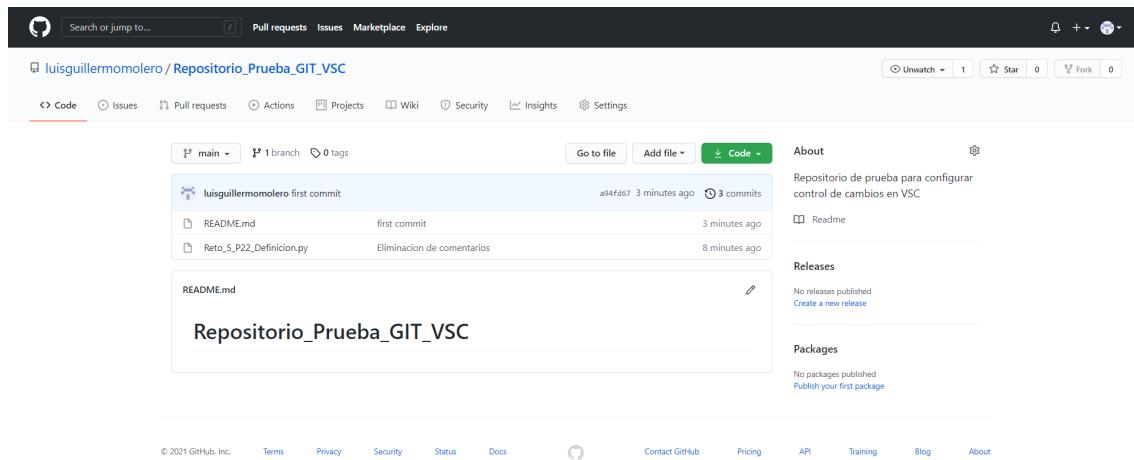


The screenshot shows a Visual Studio Code interface. In the top right corner of the terminal window, there is a red box highlighting a dropdown menu with the text "Seleccionar un elemento remoto para publicar la rama 'main'". Below the terminal, the code editor displays a Python script named "Reto_5_P22_Definicion.py". The script imports pandas and defines a function "listaPelículas" that reads an Excel file from a GitHub blob URL and prints the results. The GitHub URL in the code is:

```
rutaFileXls = 'https://github.com/luisguillermomolero/MisionTIC2022/blob/ff05b5f12ea60cf2f005309f011c1d48ab4725cb/movies.xls?raw=true'
```



Seleccionamos nuestro repositorio y listo, nos aparecerá en nuestro repositorio GitHub



IMPORTANTE:

Los pasos de “Almacenar todos los cambios”, “Confirmar todo” e “Insertar en” deben llevarse a cabo luego de una modificación en el código.

