

File Manager





Description:

Create a file manager application that allows users to perform various operations on files and directories, such as creating, deleting, copying, moving, searching, and editing files.

Features to Implement:

File Operations:

Create a new file.

Delete an existing file.

Copy a file from one location to another.

Rename the text file.

Move a file from one location to another.

Search for files by name or extension within a directory.

Edit the contents of a text file.

Directory Operations:

Create a new directory.

Delete an existing directory.

List the contents of a directory.

Rename the directory.

Exception Handling:

Handle exceptions gracefully, such as file not found, permission denied, etc.

Object Oriented Programming:

Define classes for files and directories to encapsulate their properties and behaviors.

Implement inheritance and polymorphism where applicable.

Multithreading:

Implement multithreading to perform file operations concurrently, improving performance, especially for large file operations like copying or moving.

Iterator and Generator:

Use iterators or generators to iterate over the contents of directories, allowing for efficient processing of large directory structures.

Modules:

- **os module:**

The os module in Python provides a portable way of using operating system dependent functionality. It offers a wide range of functions for interacting with the operating system, such as managing files and directories, working with processes, and accessing environment variables.

Here are some common functions and attributes in the os module:

File and Directory Operations:

`os.getcwd()`: Get the current working directory.

`os.chdir(path)`: Change the current working directory to the given path.

`os.listdir(path='.')`: List the contents of a directory.

`os.mkdir(path)`: Create a directory.

`os.makedirs(path)`: Recursively create directories.

`os.remove(path)`: Remove a file.

`os.rename(src, dst)`: Rename a file or directory.

`os.path.exists(path)`: Check if a path exists.

`os.path.isfile(path)`: Check if a path is a file.

`os.path.isdir(path)`: Check if a path is a directory.

`os.path.join(path1, path2, ...)`: Join one or more path components.

- **Shutil module:**

The `shutil` module in Python provides a higher-level interface for file operations that are more abstracted and easier to use compared to the lower-level file operations provided by the `os` module. It offers functions for copying files and directories, removing directories, and archiving operations.

Here are some common functionalities provided by the `shutil` module:

`shutil.copy(src, dst)`: Copies the file at the path `src` to the location `dst`.

`shutil.move(src, dst)`: Moves the file or directory from `src` to `dst`.

`shutil.rmtree(path)`: Deletes the directory tree rooted at `path`.

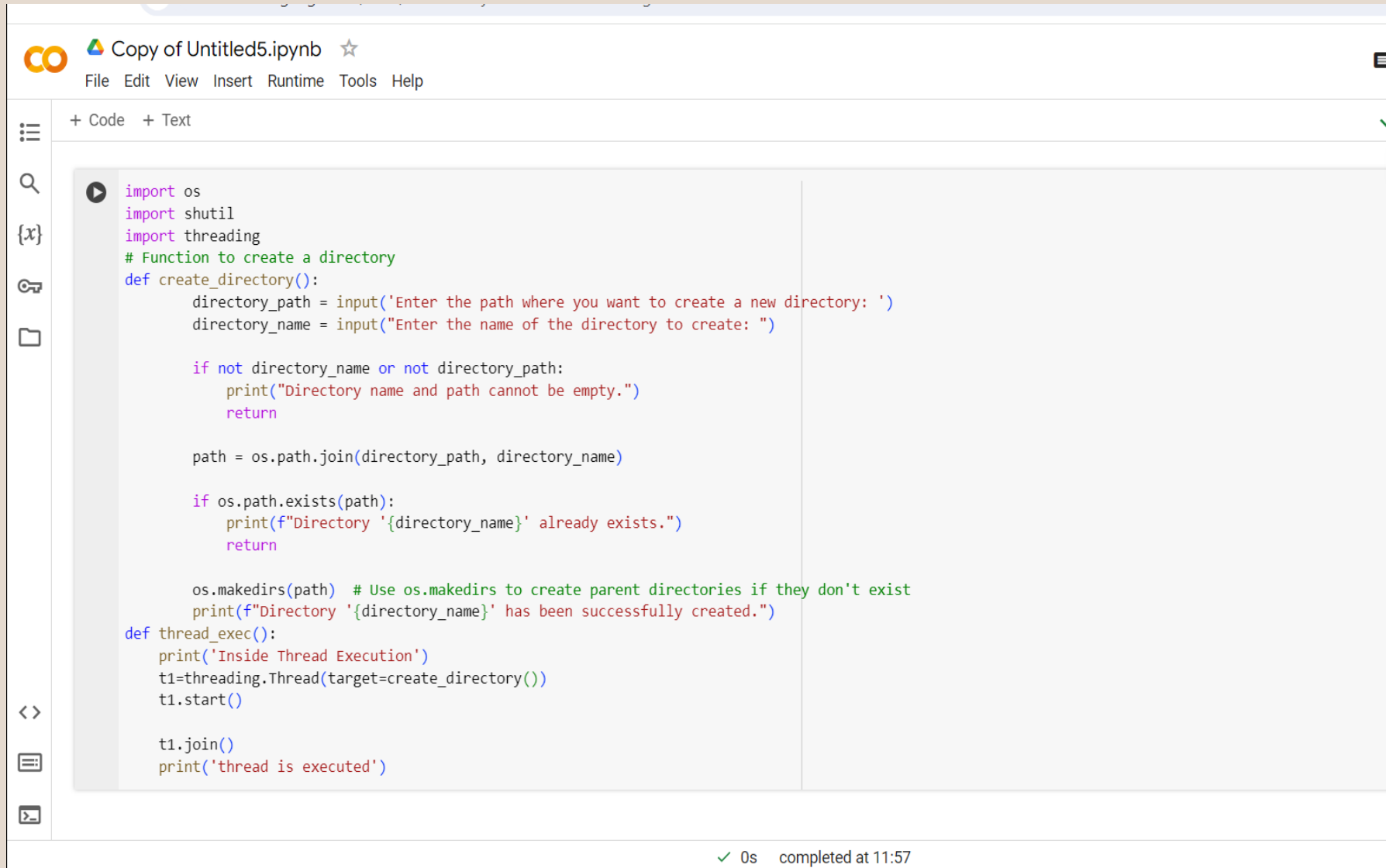
- **Threading module:**

The `threading` module in python provides a way to run multiple threads simultaneously within a single application. Threads are lighter-weight than processes , making them suitable for tasks easily.

Functions :

- **Create a Directory:** Create a new directory at the specified path.
- **Delete a Directory:** Delete an existing directory and its contents.
- **List Files in a Directory:** List files within a specified directory and their types.
- **Rename a Directory:** Rename an existing directory.
- **Move Files from One Directory to Another:** Move all files from one directory to another.
- **Create a File:** Create a new text file at the specified path.
- **Delete a File:** Delete an existing text file.
- **Edit a Text File:** Edit the content of an existing text file.
- **Copy Content from One Text File to Another:** Copy the content of one text file to another.
- **List Directories:** List directories in a specified path.

1.create_directory() function: It creates the directory in the specified path. If the directory exists it displays the message the directory already exists. The directory name and path you entered is not empty.



The screenshot shows a Jupyter Notebook titled "Copy of Untitled5.ipynb". The notebook contains a single code cell with the following Python code:

```
import os
import shutil
import threading
# Function to create a directory
def create_directory():
    directory_path = input('Enter the path where you want to create a new directory: ')
    directory_name = input("Enter the name of the directory to create: ")

    if not directory_name or not directory_path:
        print("Directory name and path cannot be empty.")
        return

    path = os.path.join(directory_path, directory_name)

    if os.path.exists(path):
        print(f"Directory '{directory_name}' already exists.")
        return

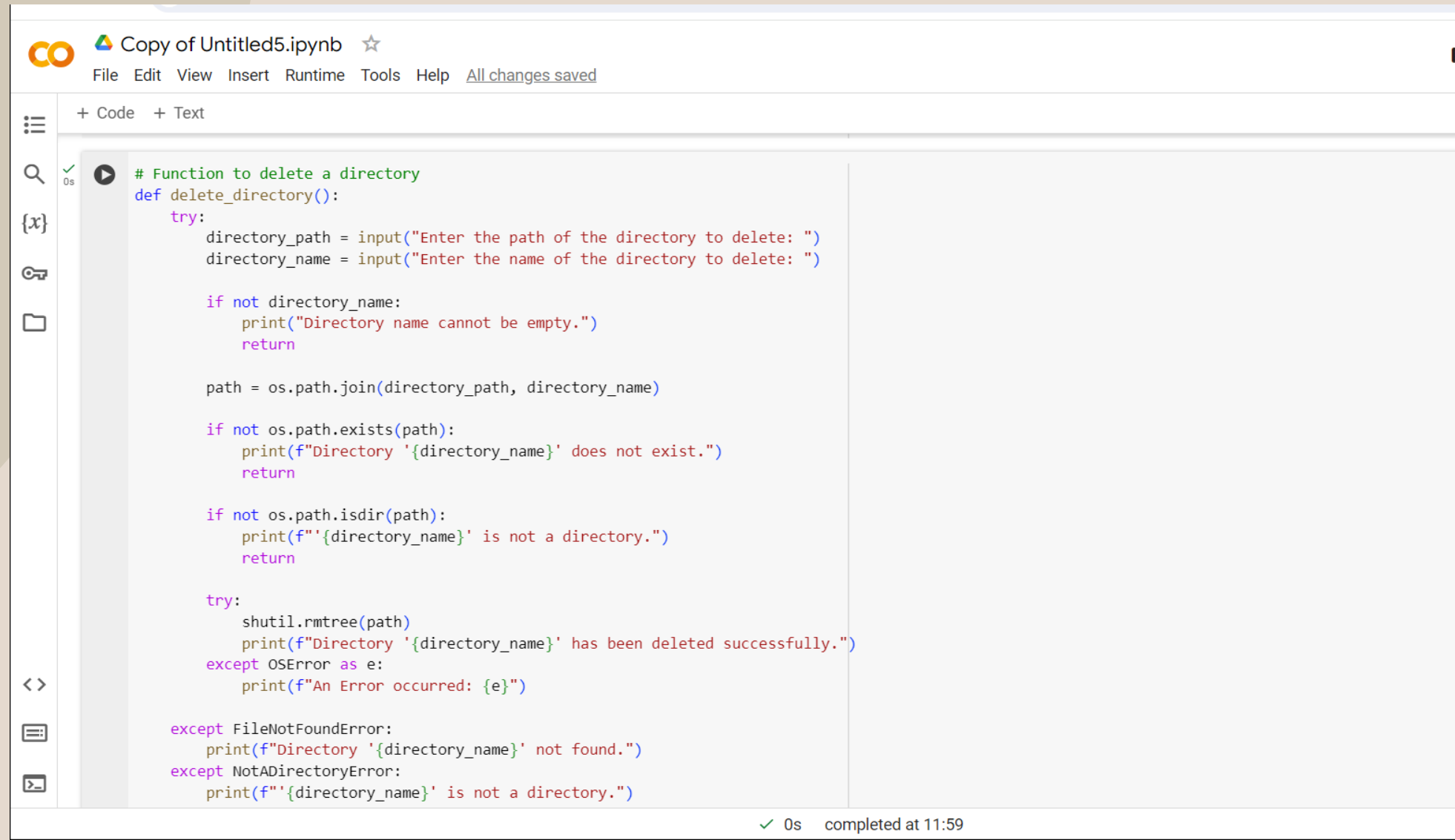
    os.makedirs(path) # Use os.makedirs to create parent directories if they don't exist
    print(f"Directory '{directory_name}' has been successfully created.")
def thread_exec():
    print('Inside Thread Execution')
    t1=threading.Thread(target=create_directory())
    t1.start()

    t1.join()
    print('thread is executed')
```

The code defines a `create_directory()` function that prompts the user for a path and a directory name. It checks if the directory already exists and uses `os.makedirs()` to create it, including parent directories. A separate `thread_exec()` function is also defined, which prints a message and starts a thread to execute `create_directory()`.

The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a sidebar with icons for file operations. The status bar at the bottom indicates "0s completed at 11:57".

2.delete_directory() function: This function deletes the directory that you entered .Directory name cannot be empty .It checks whether the directory exists or not .If the directory not found it displays that the directory not found. If you entered file instead of directory then it displays that it is not a directory..



The screenshot shows a Jupyter Notebook titled "Copy of Untitled5.ipynb". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a toolbar with icons for file operations. The main area contains a single code cell with the following Python code:

```
# Function to delete a directory
def delete_directory():
    try:
        directory_path = input("Enter the path of the directory to delete: ")
        directory_name = input("Enter the name of the directory to delete: ")

        if not directory_name:
            print("Directory name cannot be empty.")
            return

        path = os.path.join(directory_path, directory_name)

        if not os.path.exists(path):
            print(f"Directory '{directory_name}' does not exist.")
            return

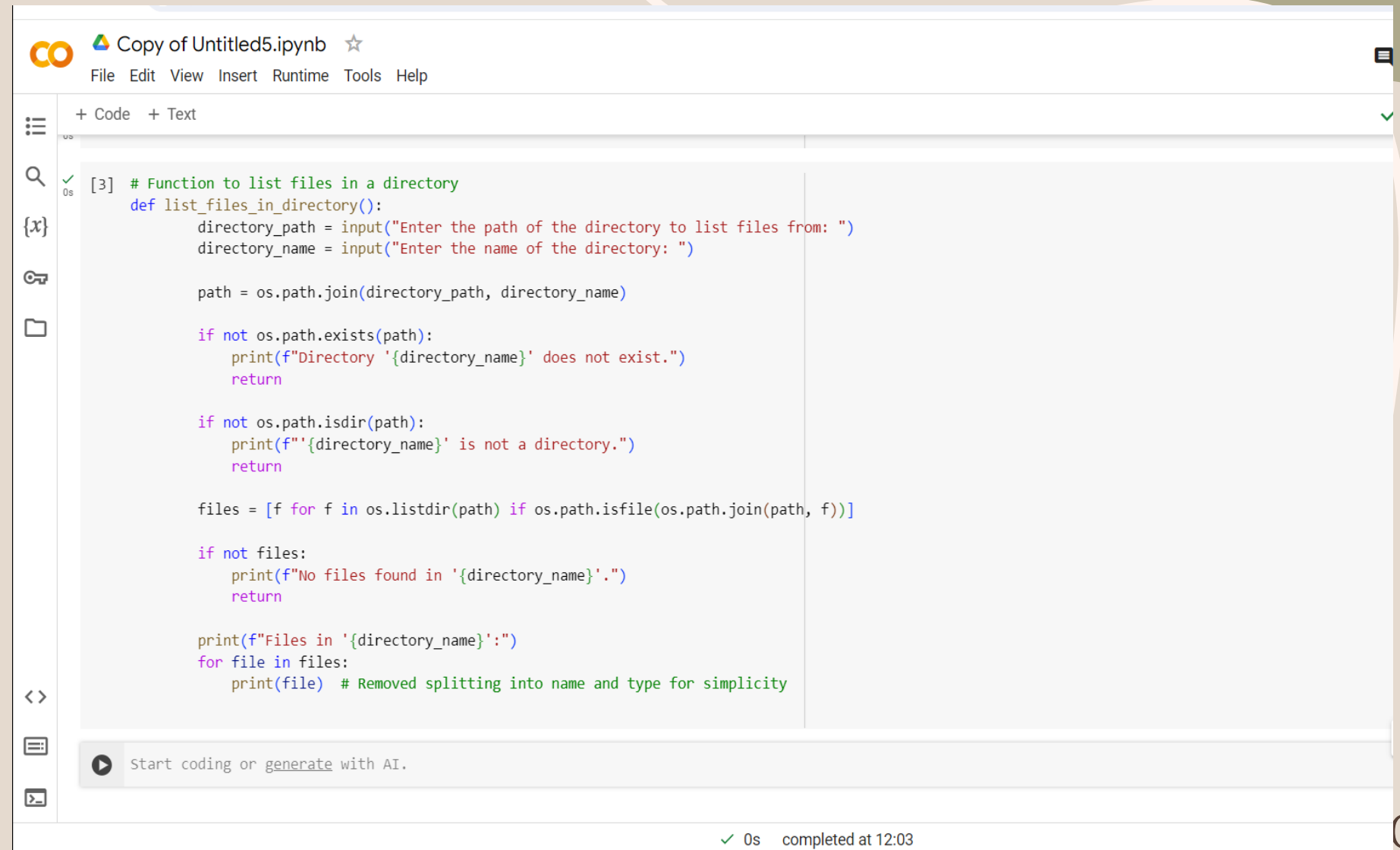
        if not os.path.isdir(path):
            print(f"'{directory_name}' is not a directory.")
            return

        try:
            shutil.rmtree(path)
            print(f"Directory '{directory_name}' has been deleted successfully.")
        except OSError as e:
            print(f"An Error occurred: {e}")

    except FileNotFoundError:
        print(f"Directory '{directory_name}' not found.")
    except NotADirectoryError:
        print(f"'{directory_name}' is not a directory.")
```

The status bar at the bottom indicates the code was executed successfully ("✓ 0s") and completed at 11:59.

3. `List_files_in_directory()` function::It displays the list of files in given directory .



The screenshot shows a Jupyter Notebook titled "Copy of Untitled5.ipynb". The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". Below the menu is a toolbar with icons for file operations and a "Start coding or generate with AI." button. The main area displays a Python code cell with the following code:

```
[3] # Function to list files in a directory
def list_files_in_directory():
    directory_path = input("Enter the path of the directory to list files from: ")
    directory_name = input("Enter the name of the directory: ")

    path = os.path.join(directory_path, directory_name)

    if not os.path.exists(path):
        print(f"Directory '{directory_name}' does not exist.")
        return

    if not os.path.isdir(path):
        print(f"'{directory_name}' is not a directory.")
        return

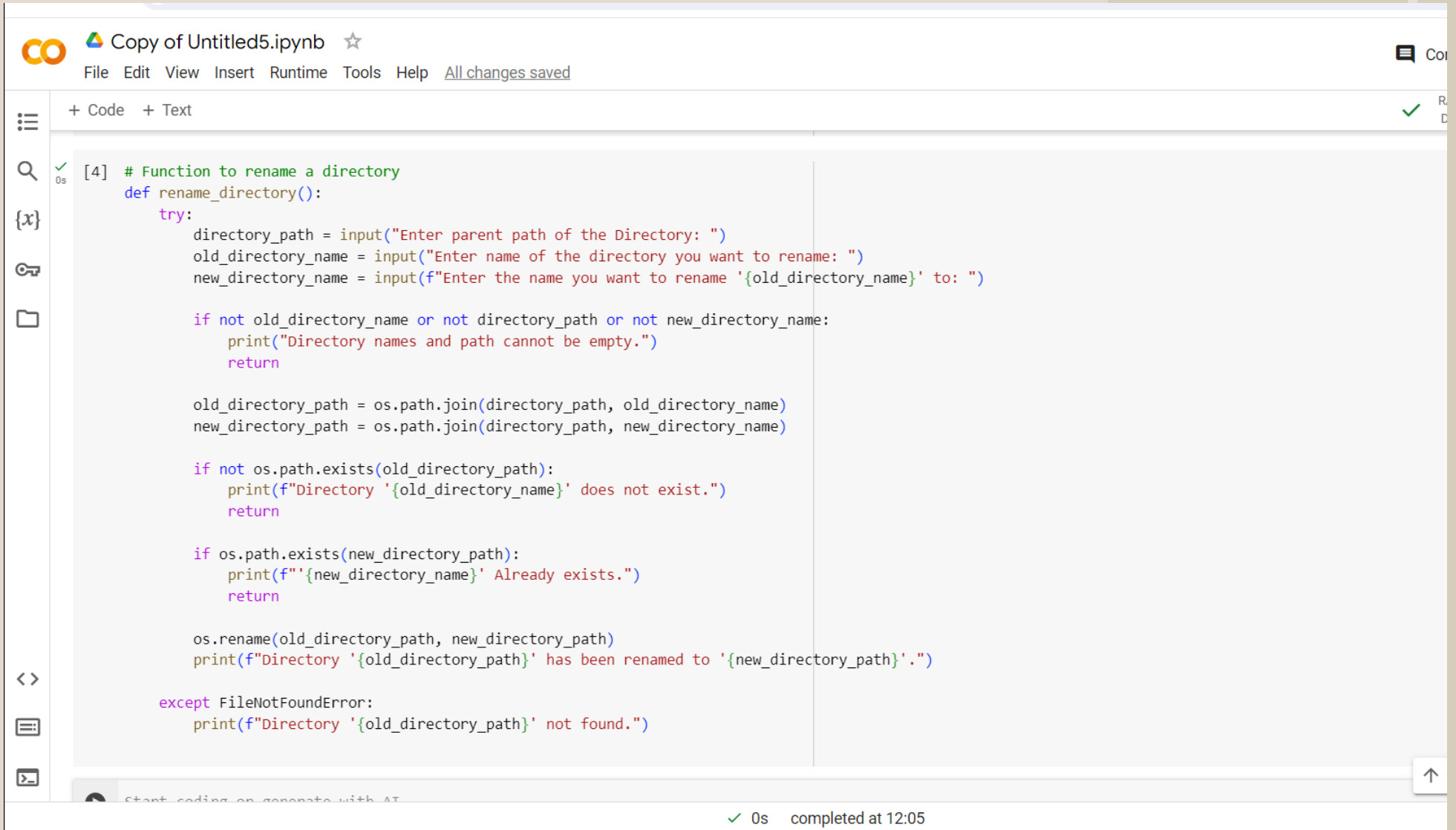
    files = [f for f in os.listdir(path) if os.path.isfile(os.path.join(path, f))]

    if not files:
        print(f"No files found in '{directory_name}'.")
        return

    print(f"Files in '{directory_name}':")
    for file in files:
        print(file) # Removed splitting into name and type for simplicity
```

The code cell is marked as "✓ 0s" and "completed at 12:03".

4.rename_directory() function: It renames the existing directory into the new name.



The screenshot shows a Jupyter Notebook titled "Copy of Untitled5.ipynb". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a status bar at the bottom indicating "0s completed at 12:05". The notebook contains a single code cell with the following Python code:

```
[4] # Function to rename a directory
def rename_directory():
    try:
        directory_path = input("Enter parent path of the Directory: ")
        old_directory_name = input("Enter name of the directory you want to rename: ")
        new_directory_name = input(f"Enter the name you want to rename '{old_directory_name}' to: ")

        if not old_directory_name or not directory_path or not new_directory_name:
            print("Directory names and path cannot be empty.")
            return

        old_directory_path = os.path.join(directory_path, old_directory_name)
        new_directory_path = os.path.join(directory_path, new_directory_name)

        if not os.path.exists(old_directory_path):
            print(f"Directory '{old_directory_name}' does not exist.")
            return

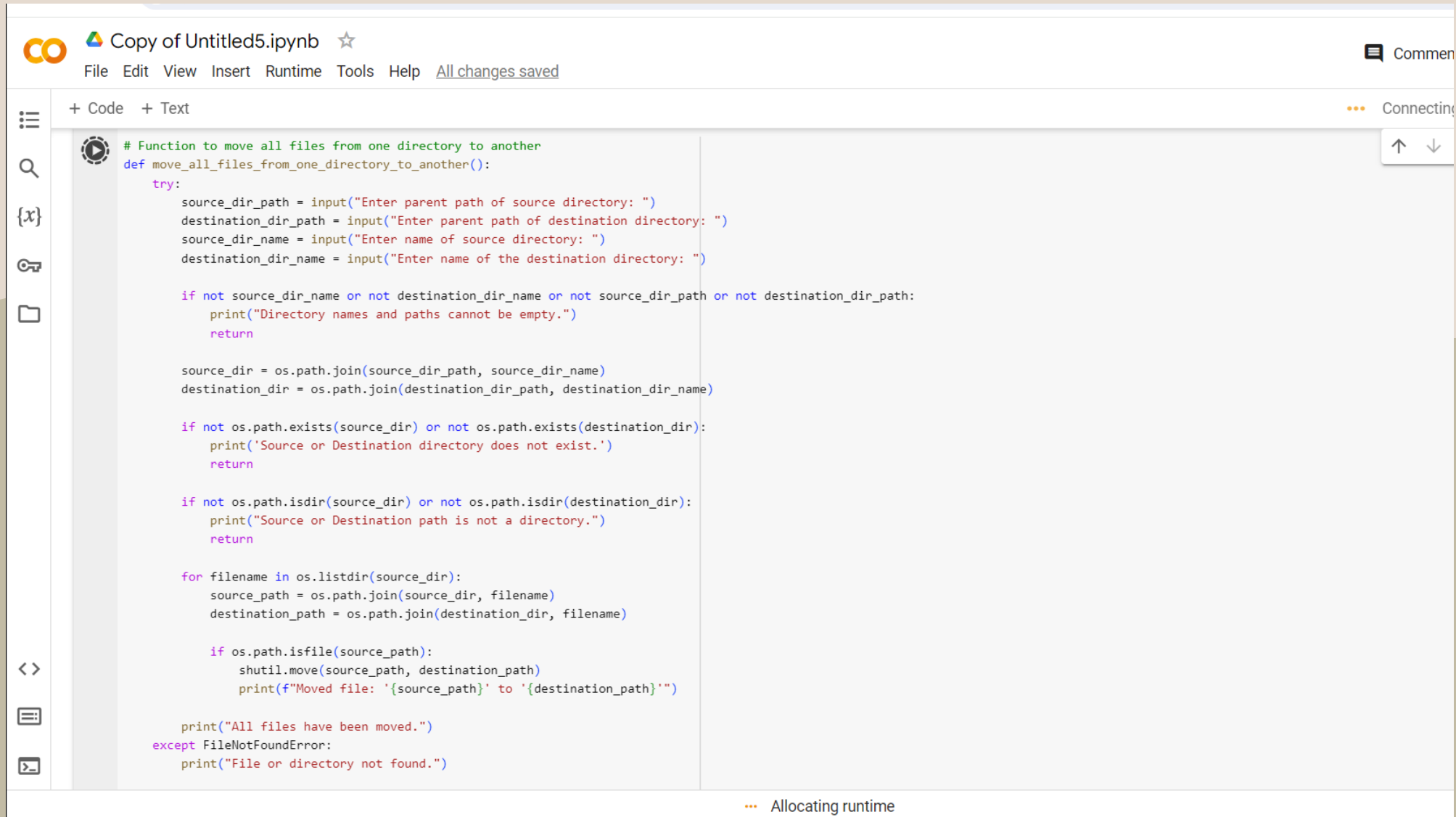
        if os.path.exists(new_directory_path):
            print(f"'{new_directory_name}' Already exists.")
            return

        os.rename(old_directory_path, new_directory_path)
        print(f"Directory '{old_directory_path}' has been renamed to '{new_directory_path}'.")

    except FileNotFoundError:
        print(f"Directory '{old_directory_path}' not found.")
```

The code defines a function `rename_directory()` that prompts the user for a parent path, an old directory name, and a new directory name. It checks for empty inputs, the existence of the old directory, and the non-existence of the new directory. If all checks pass, it uses `os.rename()` to rename the directory and prints a confirmation message. It also includes a `FileNotFoundError` exception handler.

5.move_all_files_from_one_directory_to_another() function: it moves all files from the source directory to the destination directory.



The screenshot shows a Jupyter Notebook window titled "Copy of Untitled5.ipynb". The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help", along with a status bar indicating "All changes saved". On the left, there is a sidebar with icons for file explorer, search, and other notebook functions. The main area displays a Python code cell with the following code:

```
# Function to move all files from one directory to another
def move_all_files_from_one_directory_to_another():
    try:
        source_dir_path = input("Enter parent path of source directory: ")
        destination_dir_path = input("Enter parent path of destination directory: ")
        source_dir_name = input("Enter name of source directory: ")
        destination_dir_name = input("Enter name of the destination directory: ")

        if not source_dir_name or not destination_dir_name or not source_dir_path or not destination_dir_path:
            print("Directory names and paths cannot be empty.")
            return

        source_dir = os.path.join(source_dir_path, source_dir_name)
        destination_dir = os.path.join(destination_dir_path, destination_dir_name)

        if not os.path.exists(source_dir) or not os.path.exists(destination_dir):
            print('Source or Destination directory does not exist.')
            return

        if not os.path.isdir(source_dir) or not os.path.isdir(destination_dir):
            print("Source or Destination path is not a directory.")
            return

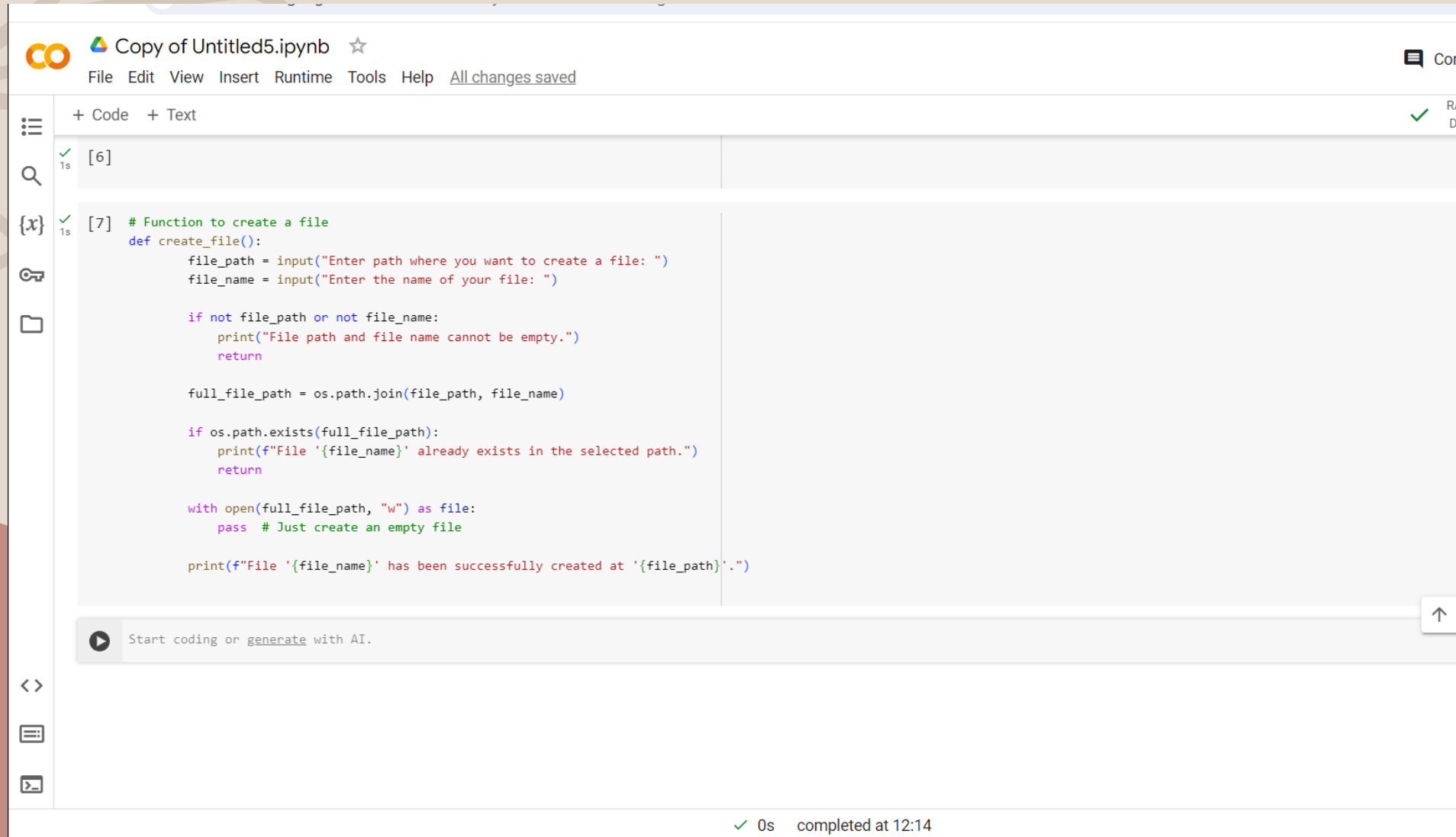
        for filename in os.listdir(source_dir):
            source_path = os.path.join(source_dir, filename)
            destination_path = os.path.join(destination_dir, filename)

            if os.path.isfile(source_path):
                shutil.move(source_path, destination_path)
                print(f"Moved file: '{source_path}' to '{destination_path}'")

        print("All files have been moved.")
    except FileNotFoundError:
        print("File or directory not found.")
```

At the bottom of the notebook, a status bar indicates "Allocating runtime".

6.create_file() function : this function creates file in the specified directory .



The screenshot displays a Jupyter Notebook titled "Copy of Untitled5.ipynb". The interface includes a top menu bar with options: File, Edit, View, Insert, Runtime, Tools, Help, and a status indicator "All changes saved". Below the menu is a toolbar with icons for file operations and a "Start coding or generate with AI." button. The main area shows two code cells. The first cell, labeled "[6]", is empty. The second cell, labeled "[7]", contains a Python function definition for `create_file()`. The function prompts the user for a file path and name, checks if they are empty, and if not, prints an error message. It then constructs the full file path, checks if the file already exists, and if so, prints a message. Finally, it creates the file using `open()` in write mode and prints a success message.

```
[6]
```

```
[7] # Function to create a file
def create_file():
    file_path = input("Enter path where you want to create a file: ")
    file_name = input("Enter the name of your file: ")

    if not file_path or not file_name:
        print("File path and file name cannot be empty.")
        return

    full_file_path = os.path.join(file_path, file_name)

    if os.path.exists(full_file_path):
        print(f"File '{file_name}' already exists in the selected path.")
        return

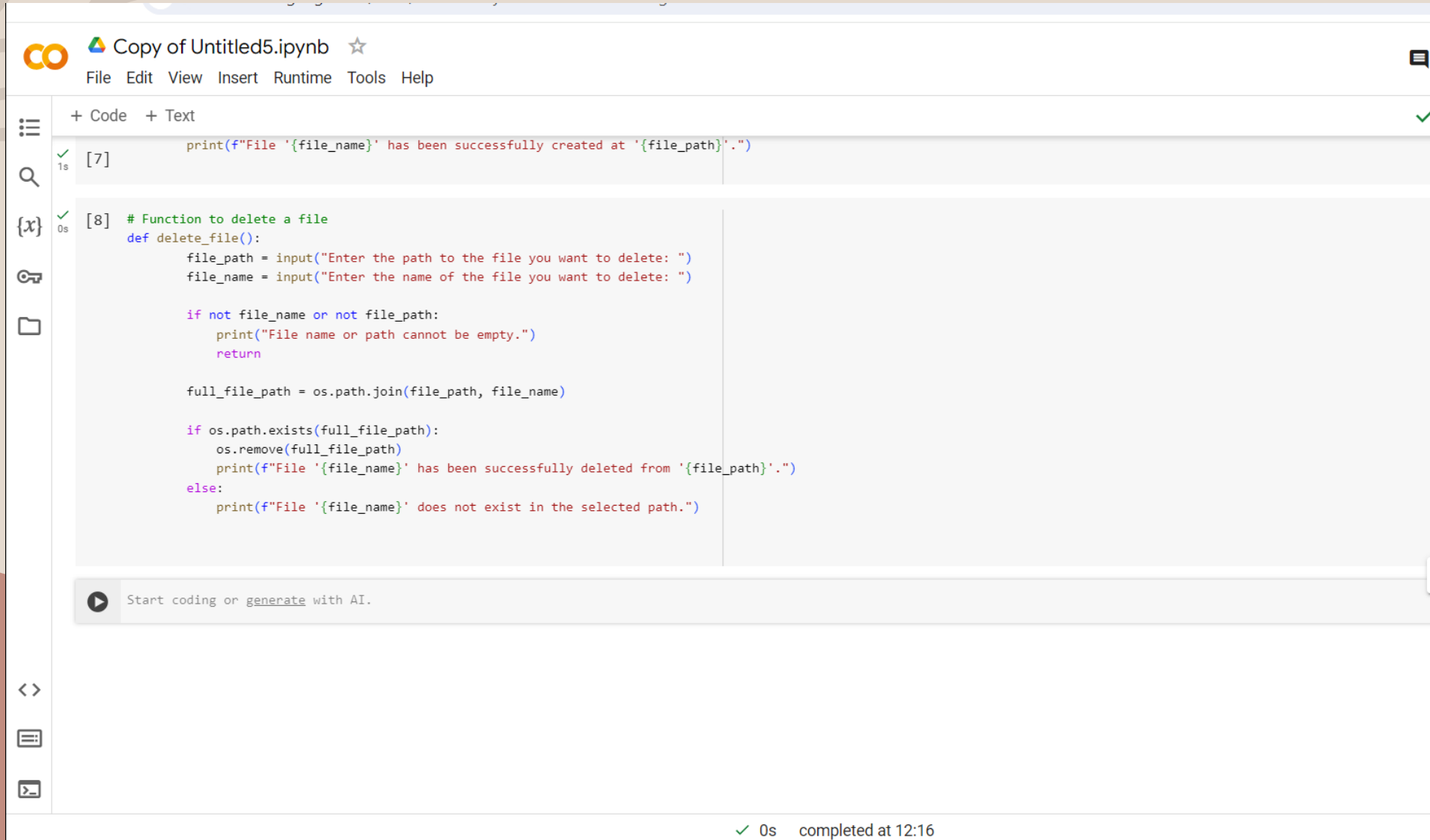
    with open(full_file_path, "w") as file:
        pass # Just create an empty file

    print(f"File '{file_name}' has been successfully created at '{file_path}'.")
```

Start coding or generate with AI.

0s completed at 12:14

7.delete_file() function: this function helps to delete the file from the specified directory



The screenshot shows a Jupyter Notebook window titled "Copy of Untitled5.ipynb". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a toolbar with icons for file operations. The notebook contains two code cells. The first cell, labeled [7], contains a single line of code: `print(f"File '{file_name}' has been successfully created at '{file_path}'")`. The second cell, labeled [8], contains a function definition for `delete_file()`. The function prompts the user for a file path and name, checks if they are empty, and then attempts to delete the file using `os.remove()`. It also includes a check to see if the file exists at the specified path. The status bar at the bottom indicates that the code was completed at 12:16.

```
[7] print(f"File '{file_name}' has been successfully created at '{file_path}'")

[8] # Function to delete a file
def delete_file():
    file_path = input("Enter the path to the file you want to delete: ")
    file_name = input("Enter the name of the file you want to delete: ")

    if not file_name or not file_path:
        print("File name or path cannot be empty.")
        return

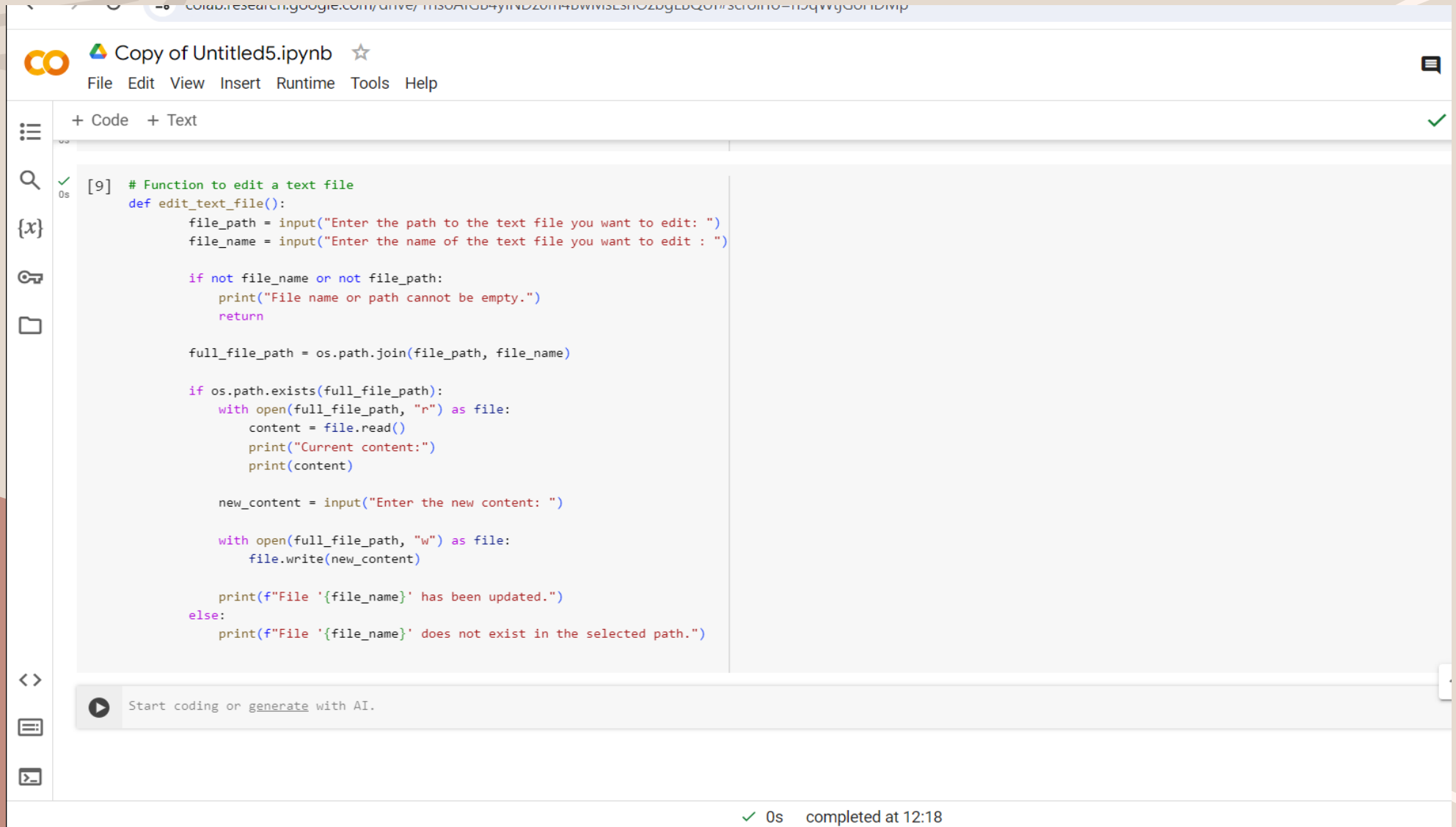
    full_file_path = os.path.join(file_path, file_name)

    if os.path.exists(full_file_path):
        os.remove(full_file_path)
        print(f"File '{file_name}' has been successfully deleted from '{file_path}'")
    else:
        print(f"File '{file_name}' does not exist in the selected path.")
```

Start coding or [generate](#) with AI.

✓ 0s completed at 12:16

8.edit_text_file() function : we can add text to the created file or else we can edit the text already containing in the file .



The screenshot shows a Jupyter Notebook titled "Copy of Untitled5.ipynb". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a toolbar with icons for file operations and execution. The main area displays a Python code cell with the following code:

```
[9] # Function to edit a text file
def edit_text_file():
    file_path = input("Enter the path to the text file you want to edit: ")
    file_name = input("Enter the name of the text file you want to edit : ")

    if not file_name or not file_path:
        print("File name or path cannot be empty.")
        return

    full_file_path = os.path.join(file_path, file_name)

    if os.path.exists(full_file_path):
        with open(full_file_path, "r") as file:
            content = file.read()
            print("Current content:")
            print(content)

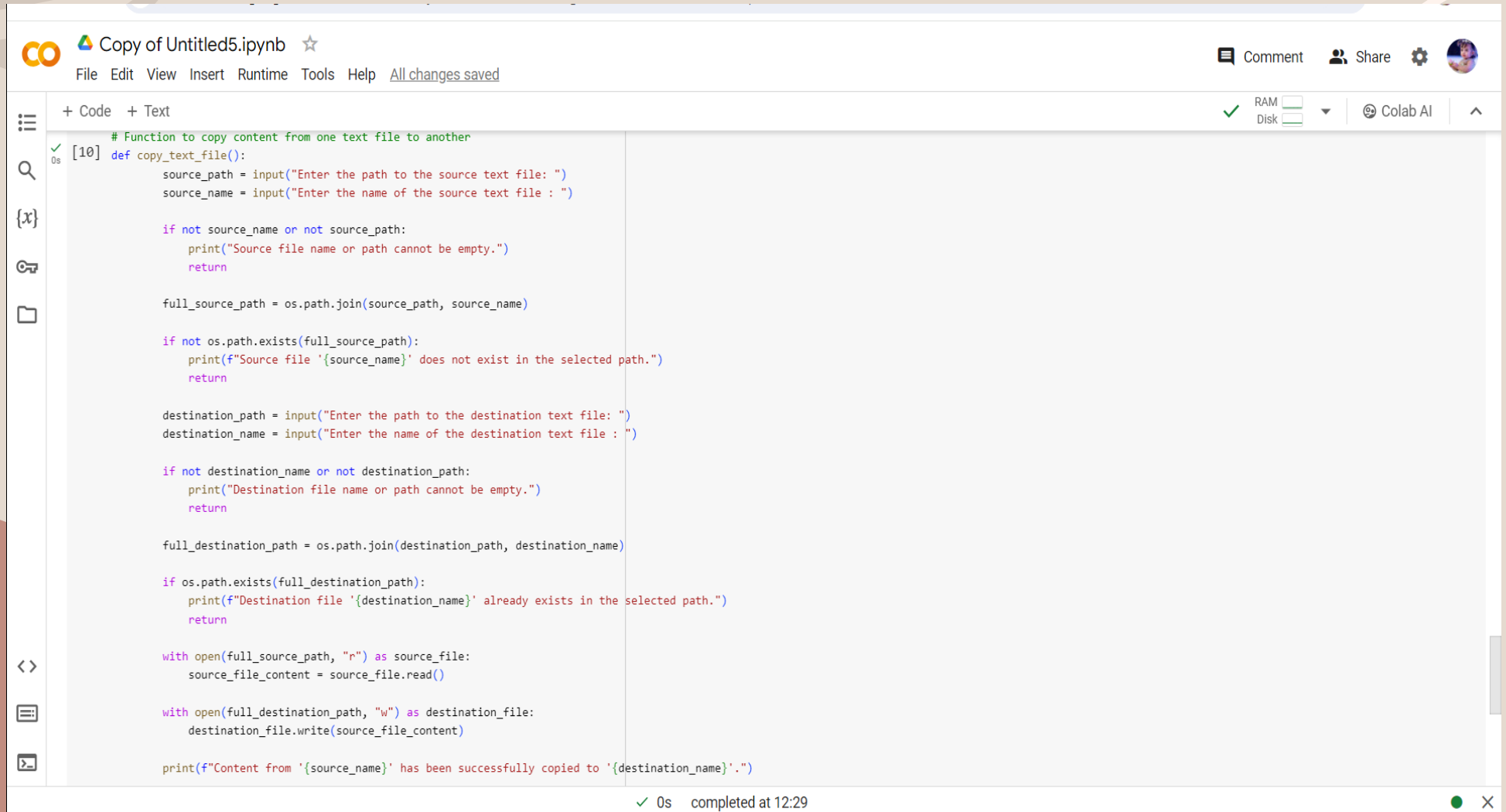
        new_content = input("Enter the new content: ")

        with open(full_file_path, "w") as file:
            file.write(new_content)

        print(f"File '{file_name}' has been updated.")
    else:
        print(f"File '{file_name}' does not exist in the selected path.")
```

At the bottom of the notebook, there is a status bar showing a green checkmark, "0s", and "completed at 12:18".

9.copy_text_file() function : it copies the text files from source file to destination file..



The screenshot shows a Google Colab notebook interface. The title bar reads "Copy of Untitled5.ipynb" with a star icon. Below the title bar is a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help", followed by a link "All changes saved". On the right side of the title bar are icons for "Comment", "Share", and a user profile. Below the menu bar is a toolbar with a green checkmark, "RAM" and "Disk" usage indicators, a dropdown arrow, "Colab AI", and an upward arrow. The main area of the notebook displays a Python function named `copy_text_file()`. The function takes no arguments and uses `input()` to get the source and destination file paths and names. It includes several conditional checks: if the source name or path is empty, if the source file does not exist, if the destination name or path is empty, and if the destination file already exists. If all checks pass, it opens the source file in read mode, reads its content, opens the destination file in write mode, and writes the content. Finally, it prints a success message. The status bar at the bottom shows a green checkmark, "0s", and "completed at 12:29".

```
# Function to copy content from one text file to another
[10] def copy_text_file():
    source_path = input("Enter the path to the source text file: ")
    source_name = input("Enter the name of the source text file : ")

    if not source_name or not source_path:
        print("Source file name or path cannot be empty.")
        return

    full_source_path = os.path.join(source_path, source_name)

    if not os.path.exists(full_source_path):
        print(f"Source file '{source_name}' does not exist in the selected path.")
        return

    destination_path = input("Enter the path to the destination text file: ")
    destination_name = input("Enter the name of the destination text file : ")

    if not destination_name or not destination_path:
        print("Destination file name or path cannot be empty.")
        return

    full_destination_path = os.path.join(destination_path, destination_name)

    if os.path.exists(full_destination_path):
        print(f"Destination file '{destination_name}' already exists in the selected path.")
        return


    with open(full_source_path, "r") as source_file:
        source_file_content = source_file.read()

    with open(full_destination_path, "w") as destination_file:
        destination_file.write(source_file_content)

    print(f"Content from '{source_name}' has been successfully copied to '{destination_name}'.")
```

✓ 0s completed at 12:29

10 . List_directories() : it displays the list of directories in the given directory.



The image shows a screenshot of a code editor window. The editor has a light gray background and a dark gray sidebar on the left. The sidebar contains icons for a file explorer, a search bar, and a terminal. The main area of the editor displays a Python script. The script defines a function `list_directories()` that takes no arguments. It prompts the user to enter a path using `input()`. It then checks if the path exists using `os.path.exists()`. If it exists, it lists the directories using `os.listdir()` and filters out files using `os.path.isdir()`. It prints the list of directories. If the path does not exist, it prints an error message. The script is executed, and the output is shown in the terminal at the bottom of the editor. The terminal shows the prompt `>` and the output `Directories in 'C:\Users\user\Documents':` followed by a list of directories: `1. Desktop`, `2. Downloads`, `3. Music`, `4. Pictures`, `5. Videos`, `6. Public`, `7. Templates`, `8. Desktop`, `9. Downloads`, `10. Music`, `11. Pictures`, `12. Videos`, `13. Public`, `14. Templates`. The status bar at the bottom of the editor shows a green checkmark, the text `0s`, and the time `completed at 7:16 PM`.

```
# Function to list directories in a path
def list_directories():
    path = input("Enter the path to list directories from: ")

    if os.path.exists(path):
        items = os.listdir(path)
        directories = [item for item in items if os.path.isdir(os.path.join(path, item))]

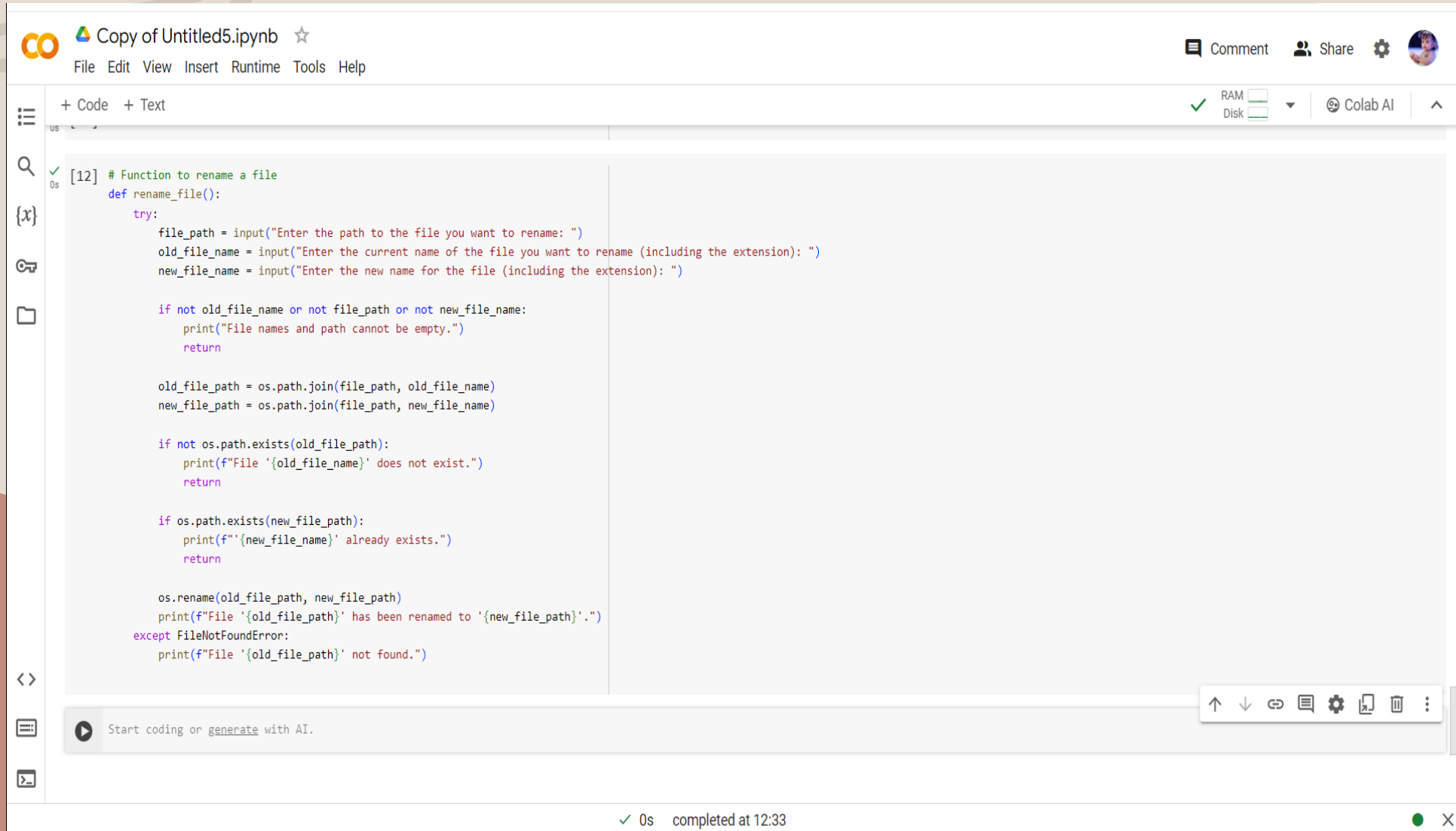
        if not directories:
            print(f"No directories found in '{path}'.")
        else:
            print(f"Directories in '{path}':")

            for directory in directories:
                print(directory)

    else:
        print(f"The specified path '{path}' does not exist.")
```

✓ 0s completed at 7:16 PM

11. rename_file() function :: it changes the name of the file from old name to new name.



The screenshot shows a Jupyter Notebook titled "Copy of Untitled5.ipynb" with a star icon. The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". On the right, there are buttons for "Comment", "Share", and a settings gear. Below the menu bar, there are tabs for "+ Code" and "+ Text". The main area displays a Python function named `rename_file()` with the following code:

```
[12] # Function to rename a file
def rename_file():
    try:
        file_path = input("Enter the path to the file you want to rename: ")
        old_file_name = input("Enter the current name of the file you want to rename (including the extension): ")
        new_file_name = input("Enter the new name for the file (including the extension): ")

        if not old_file_name or not file_path or not new_file_name:
            print("File names and path cannot be empty.")
            return

        old_file_path = os.path.join(file_path, old_file_name)
        new_file_path = os.path.join(file_path, new_file_name)

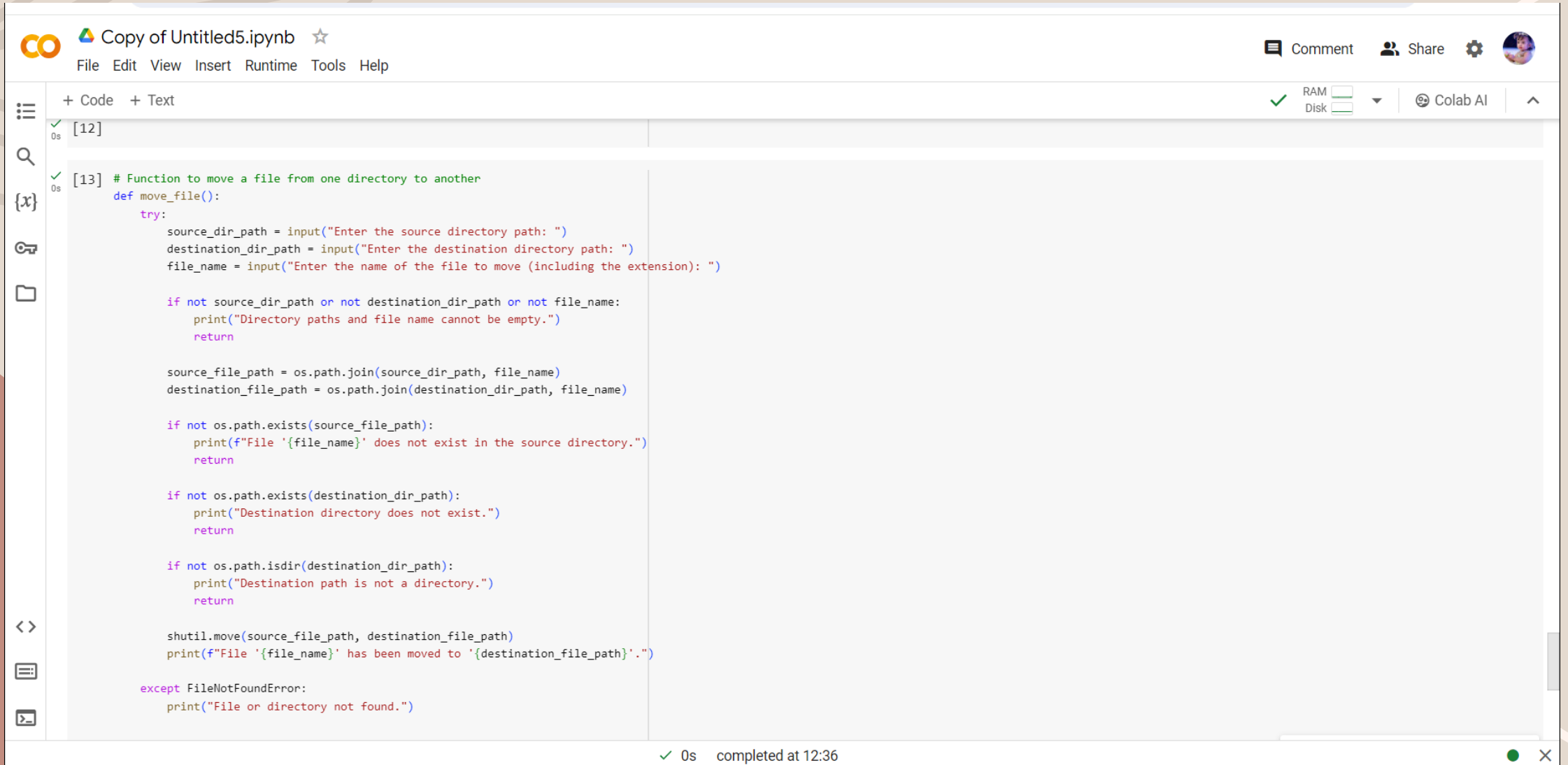
        if not os.path.exists(old_file_path):
            print(f"File '{old_file_name}' does not exist.")
            return

        if os.path.exists(new_file_path):
            print(f"'{new_file_name}' already exists.")
            return

        os.rename(old_file_path, new_file_path)
        print(f"File '{old_file_path}' has been renamed to '{new_file_path}'.")
    except FileNotFoundError:
        print(f"File '{old_file_path}' not found.")
```

At the bottom of the notebook, there is a status bar showing "✓ 0s completed at 12:33".

12. move_file() function: this function moves all the files from source to destination .



The screenshot shows a Google Colab notebook titled "Copy of Untitled5.ipynb". The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". On the right, there are buttons for "Comment", "Share", and a settings gear. Below the menu, a toolbar shows "+ Code" and "+ Text" tabs, along with RAM and Disk usage indicators, and a "Colab AI" button. The left sidebar contains icons for file management and search. The main code area displays a Python function named `move_file()` that prompts the user for source and destination directory paths and a file name. It includes several conditional checks: if any input is empty, it prints an error and returns; if the source file path does not exist, it prints an error and returns; if the destination directory does not exist, it prints an error and returns; if the destination path is not a directory, it prints an error and returns. If all checks pass, it uses `shutil.move()` to move the file and prints a confirmation message. An `except` block handles `FileNotFoundError` by printing "File or directory not found." The status bar at the bottom indicates the code was "completed at 12:36".

```
[12]

[13] # Function to move a file from one directory to another
def move_file():
    try:
        source_dir_path = input("Enter the source directory path: ")
        destination_dir_path = input("Enter the destination directory path: ")
        file_name = input("Enter the name of the file to move (including the extension): ")

        if not source_dir_path or not destination_dir_path or not file_name:
            print("Directory paths and file name cannot be empty.")
            return

        source_file_path = os.path.join(source_dir_path, file_name)
        destination_file_path = os.path.join(destination_dir_path, file_name)

        if not os.path.exists(source_file_path):
            print(f"File '{file_name}' does not exist in the source directory.")
            return

        if not os.path.exists(destination_dir_path):
            print("Destination directory does not exist.")
            return

        if not os.path.isdir(destination_dir_path):
            print("Destination path is not a directory.")
            return

        shutil.move(source_file_path, destination_file_path)
        print(f"File '{file_name}' has been moved to '{destination_file_path}'.")

    except FileNotFoundError:
        print("File or directory not found.")
```

✓ 0s completed at 12:36

Main() class : Here all the functions are set as menu . When we enter the desired choice from the menu then the corresponding function called and executes the statements..

```
# Main menu function
class main():
    while True:
        print("\nMenu:")
        print("1. Create a Directory")
        print("2. Delete a Directory")
        print("3. List Files in a Directory")
        print("4. Rename a Directory")
        print("5. Move Files from One Directory to Another")
        print("6. Create a File")
        print("7. Delete a File")
        print("8. Edit a Text File")
        print("9. Copy Content from One Text File to Another")
        print("10. List Directories")
        print("11. Rename a File")
        print("12. Move a File")
        print("13. Quit")

        choice = input("Enter Your Choice: ")
        if choice == '1':
            thread_exec()
        elif choice == '2':
            delete_directory()
        elif choice == '3':
            list_files_in_directory()
        elif choice == '4':
            rename_directory()
        elif choice == '5':
            move_all_files_from_one_directory_to_another()
        elif choice == '6':
            create_file()
        elif choice == '7':
            delete_file()
        elif choice == '8':
            edit_text_file()
        elif choice == '9':
            copy_text_file()
        elif choice == '10':
            list_directories()
        elif choice == '11':
            rename_file()
        elif choice == '12':
            move_file()
        elif choice == '13':
            break
        else:
            print("Invalid choice. Please select a valid option.")

if __name__ == "__main__":
    M = main()
```

This is menu when the main class executed .
It asks for the choice from the menu . If you
enter your choice it will executes the
corresponding function.

```
... Menu:
1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit
Enter Your Choice: 
```

When we enter 1 as choice from menu then thread executes the create directory function and it creates the directory in the path where you specified .

Menu:

1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit

Enter Your Choice: 1

Inside Thread Execution

Enter the path where you want to create a new directory: /content/

Enter the name of the directory to create: Demo

Directory 'Demo' has been successfully created.

thread is executed

When you enter 2 as choice from menu then it calls the delete directory function and deletes the directory from the specified path .

Menu:


1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit

Enter Your Choice: 2

Enter the path of the directory to delete: /content/

Enter the name of the directory to delete: Demo

Directory 'Demo' has been deleted successfully.



When you enter 3 as choice from menu then it calls the list files in a directory function and it displays all files from specified directory.


```
Menu:
1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit
Enter Your Choice: 3
Enter the path of the directory to list files from: /content/
Enter the name of the directory: sample_data
Files in 'sample_data':
anscombe.json
README.md
File1
california_housing_train.csv
mnist_train_small.csv
mnist_test.csv
california_housing_test.csv
```


When you enter 4 as choice from menu then it calls rename directory function and it changes the name from old to new name of the directory.

```
Menu:
1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit
Enter Your Choice: 4
Enter parent path of the Directory: /content/
Enter name of the directory you want to rename: Demo
Enter the name you want to rename 'Demo' to: Demo1
Directory '/content/Demo' has been renamed to '/content/Demo1'.
```

When you enter 5 as choice from menu it calls move files from one directory to another function and moves all files from one source directory to destination directory .

```
Menu:
1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit
Enter Your Choice: 5
Enter parent path of source directory: /content/
Enter parent path of destination directory: /content/
Enter name of source directory: sample_data
Enter name of the destination directory: Demo1
Moved file: '/content/sample_data/anscombe.json' to '/content/Demo1/anscombe.json'
Moved file: '/content/sample_data/README.md' to '/content/Demo1/README.md'
Moved file: '/content/sample_data/File1' to '/content/Demo1/File1'
Moved file: '/content/sample_data/california_housing_train.csv' to '/content/Demo1/california_housing_train.csv'
Moved file: '/content/sample_data/mnist_train_small.csv' to '/content/Demo1/mnist_train_small.csv'
Moved file: '/content/sample_data/mnist_test.csv' to '/content/Demo1/mnist_test.csv'
Moved file: '/content/sample_data/california_housing_test.csv' to '/content/Demo1/california_housing_test.csv'
All files have been moved.
```



When you enter 6 as choice from menu then it calls the create file function and creates a file in the specified path.

Menu:

1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit

Enter Your Choice: 6

Enter path where you want to create a file: /content/Demo1

Enter the name of your file: File2

File 'File2' has been successfully created at '/content/Demo1'.

When you enter 7 as choice from menu then it will call delete file function and then it deletes the file from the specified directory.

Menu:

1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit

Enter Your Choice: 7

Enter the path to the file you want to delete: /content/Demo1

Enter the name of the file you want to delete: File1

File 'File1' has been successfully deleted from '/content/Demo1'.

Menu:

When you enter 8 as choice from menu then it calls edit text file function and it updates the file with the updated text.

Menu:

1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit

Enter Your Choice: 8

Enter the path to the text file you want to edit: /content/Demo1

Enter the name of the text file you want to edit : File2

Current content:

Enter the new content: This is python project

File 'File2' has been updated.

When you enter 9 as choice from menu then it calls copy content from one file to another function and it copies the files from source directory to destination directory.

Menu:

1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit

Enter Your Choice: 9

Enter the path to the source text file: /content/sample_data

Enter the name of the source text file : california_housing_test.csv

Enter the path to the destination text file: /content/sample_data

Enter the name of the destination text file : File1

Content from 'california_housing_test.csv' has been successfully copied to 'File1'.

When you enter 10 as choice from menu then it calls list directories function and it displays all the directories from the given directory..

```
Menu:
1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit
Enter Your Choice: 10
Enter the path to list directories from: /bin/
Directories in '/bin/':
X11
```

When you enter 11 as choice from menu then it calls the rename file function and it changes the name of the file from old name to new name.

Menu:

1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit

Enter Your Choice: 11

Enter the path to the file you want to rename: /content/sample_data

Enter the current name of the file you want to rename (including the extension): README.md

Enter the new name for the file (including the extension): readme

File '/content/sample_data/README.md' has been renamed to '/content/sample_data/readme'.

When you enter 12 as choice from menu then it calls move a file function and it moves files from source to destination directory.

Menu:

1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit

Enter Your Choice: 12

Enter the source directory path: /content/Demo1

Enter the destination directory path: /content/sample_data

Enter the name of the file to move (including the extension): File2

File 'File2' has been moved to '/content/sample_data/File2'.

When you enter 13 as choice from menu then quits from the execution .

Menu:

1. Create a Directory
2. Delete a Directory
3. List Files in a Directory
4. Rename a Directory
5. Move Files from One Directory to Another
6. Create a File
7. Delete a File
8. Edit a Text File
9. Copy Content from One Text File to Another
10. List Directories
11. Rename a File
12. Move a File
13. Quit

Enter Your Choice: 13



thank you

