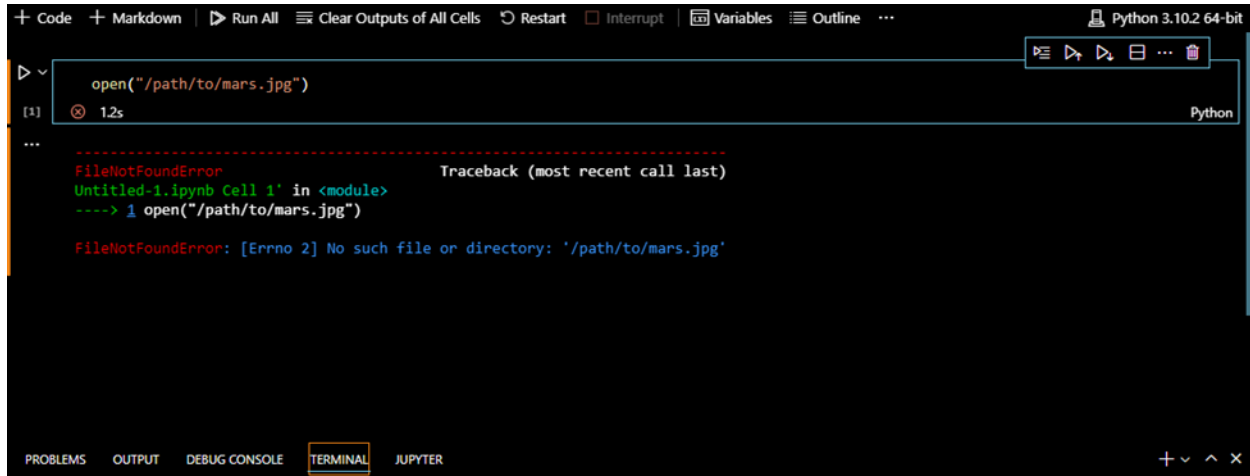


## KATA 10

### TRACEBACKS

Si intentamos en un notebook, abrir un archivo inexistente sucede lo siguiente:



The screenshot shows a Jupyter Notebook interface. The top bar includes tabs for Code, Markdown, Run All, Clear Outputs of All Cells, Restart, Interrupt, Variables, Outline, and a Python 3.10.2 64-bit icon. The main area contains a code cell with the following content:

```
open("/path/to/mars.jpg")
```

Below the code, the output shows a traceback for a `FileNotFoundError`:


```
FileNotFoundError                                Traceback (most recent call last)
Untitled-1.ipynb Cell 1 in <module>
----> 1 open("/path/to/mars.jpg")

FileNotFoundError: [Errno 2] No such file or directory: '/path/to/mars.jpg'
```

The bottom of the interface shows tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and JUPYTER.

Intenta crear un archivo de Python y asígnale el nombre `open.py`, con el contenido siguiente:

```
def main():
    open("/path/to/mars.jpg")
if __name__ == '__main__':
    main()
```



The screenshot shows a VS Code editor with a file named `open.py` open. The code in the file is:

```
1 def main():
2     open("/path/to/mars.jpg")
3
4 if __name__ == '__main__':
5     main()
```

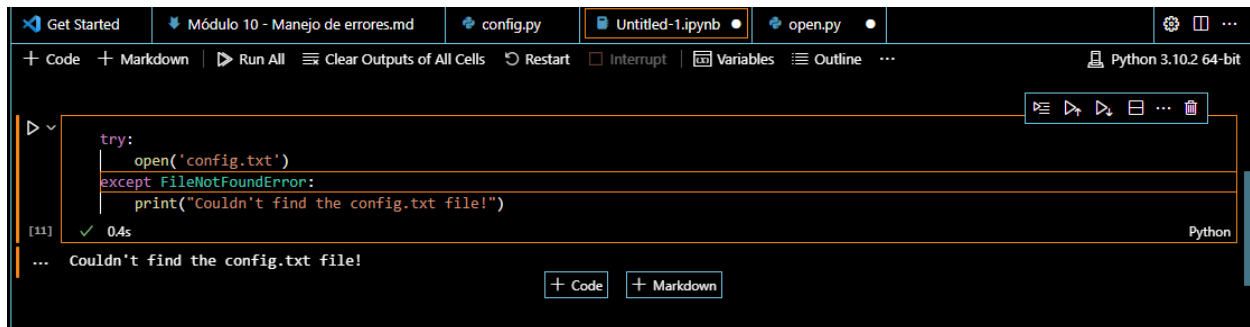
The bottom of the interface shows the TERMINAL tab, which displays the output of running the script:

```
C:\Users\alopez\Documents\memoriakingston\python>python open.py
Traceback (most recent call last):
  File "C:\Users\alopez\Documents\memoriakingston\python\open.py", line 5, in <module>
    main()
  File "C:\Users\alopez\Documents\memoriakingston\python\open.py", line 2, in main
    open("/path/to/mars.jpg")
FileNotFoundError: [Errno 2] No such file or directory: '/path/to/mars.jpg'
```

The bottom right of the interface shows tabs for powershell and cmd.

## Controlando excepciones

Si queremos controlar esa excepción, podemos hacerlo con un bloque try y except:



```
try:
    open('config.txt')
except FileNotFoundError:
    print("Couldn't find the config.txt file!")
```

[11] ✓ 0.4s Python

... Couldn't find the config.txt file!

Vamos a crear un archivo de Python denominado config.py. El archivo tiene código que busca y lee el archivo de configuración del sistema de navegación:



```
1 def main():
2     try:
3         configuration = open('config.txt')
4     except FileNotFoundError:
5         print("Couldn't find the config.txt file!")
6
7
8 if __name__ == '__main__':
9     main()
10
11
12
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** JUPYTER

```
C:\Users\alopez\Documents\memoriakingston\python>python config.py
Traceback (most recent call last):
  File "C:\Users\alopez\Documents\memoriakingston\python\config.py", line 9, in <module>
    main()
  File "C:\Users\alopez\Documents\memoriakingston\python\config.py", line 3, in main
    configuration = open('config.txt')
PermissionError: [Errno 13] Permission denied: 'config.txt'

C:\Users\alopez\Documents\memoriakingston\python>
```

Una manera poco útil de controlar este error sería detectar todas las excepciones posibles para evitar un traceback:



The screenshot shows a Jupyter Notebook with a file explorer at the top displaying 'Módulo 10 - Manejo de errores.md', 'config.py', 'Untitled-1.ipynb', and 'open.py'. The main editor shows a Python script in 'config.py' with line numbers 1 to 12. The script defines a 'main()' function that attempts to open 'config.txt' and catches a general 'Exception'. Below the editor, the 'TERMINAL' tab is active, showing the command 'python config.py' and its output: 'Couldn't find the config.txt file!'. The right sidebar shows 'powershell' and 'cmd' tabs.

```
1 def main():
2     try:
3         configuration = open('config.txt')
4     except Exception:
5         print("Couldn't find the config.txt file!")
6
7
8 if __name__ == '__main__':
9     main()
10
11
12
```

C:\Users\alopez\Documents\memoriakingston\python>python config.py  
Couldn't find the config.txt file!  
C:\Users\alopez\Documents\memoriakingston\python>

Revertiremos la detección de `FileNotFoundException` y luego agregamos otro bloque `except` para detectar `PermissionError`:

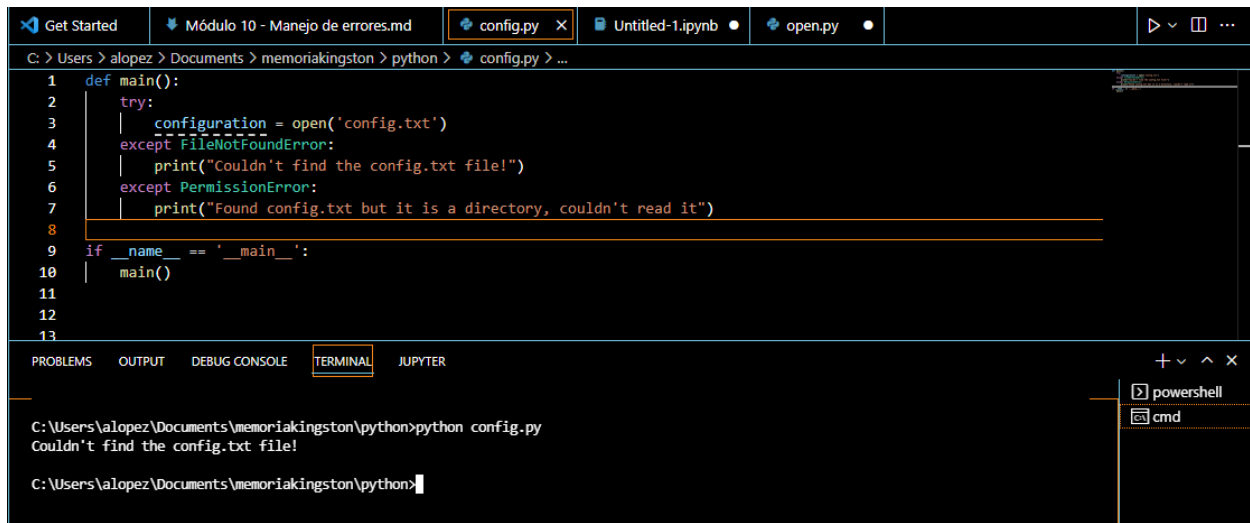


This screenshot shows the same Jupyter Notebook interface as the previous one, but with an updated 'config.py' script. The script now catches 'FileNotFoundException' and 'PermissionError' specifically. The terminal output shows the command 'python config.py' and the output: 'Found config.txt but it is a directory, couldn't read it'. The file explorer and sidebar remain the same.

```
1 def main():
2     try:
3         configuration = open('config.txt')
4     except FileNotFoundError:
5         print("Couldn't find the config.txt file!")
6     except PermissionError:
7         print("Found config.txt but it is a directory, couldn't read it")
8
9
10 if __name__ == '__main__':
11     main()
12
13
```

C:\Users\alopez\Documents\memoriakingston\python>python config.py  
Found config.txt but it is a directory, couldn't read it  
C:\Users\alopez\Documents\memoriakingston\python>

Eliminamos el archivo config.txt para asegurarnos de que se alcanza el primer bloque except en su lugar:



The screenshot shows a Jupyter Notebook window with the following components:

- File Explorer:** Shows the current directory structure: `C: > Users > alopez > Documents > memoriakingston > python > config.py > ...`. The file `config.py` is selected.
- Code Editor:** Contains the following Python code:

```
1 def main():
2     try:
3         configuration = open('config.txt')
4     except FileNotFoundError:
5         print("Couldn't find the config.txt file!")
6     except PermissionError:
7         print("Found config.txt but it is a directory, couldn't read it")
8
9 if __name__ == '__main__':
10     main()
11
12
13
```
- Terminal:** Shows the output of running the script:

```
C:\Users\alopez\Documents\memoriakingston\python>python config.py
Couldn't find the config.txt file!

C:\Users\alopez\Documents\memoriakingston\python>
```
- Taskbar:** Shows the `powershell` and `cmd` applications.