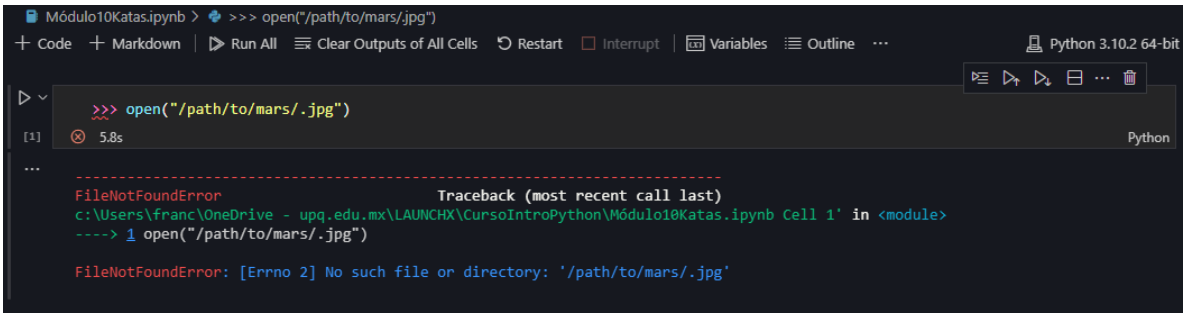
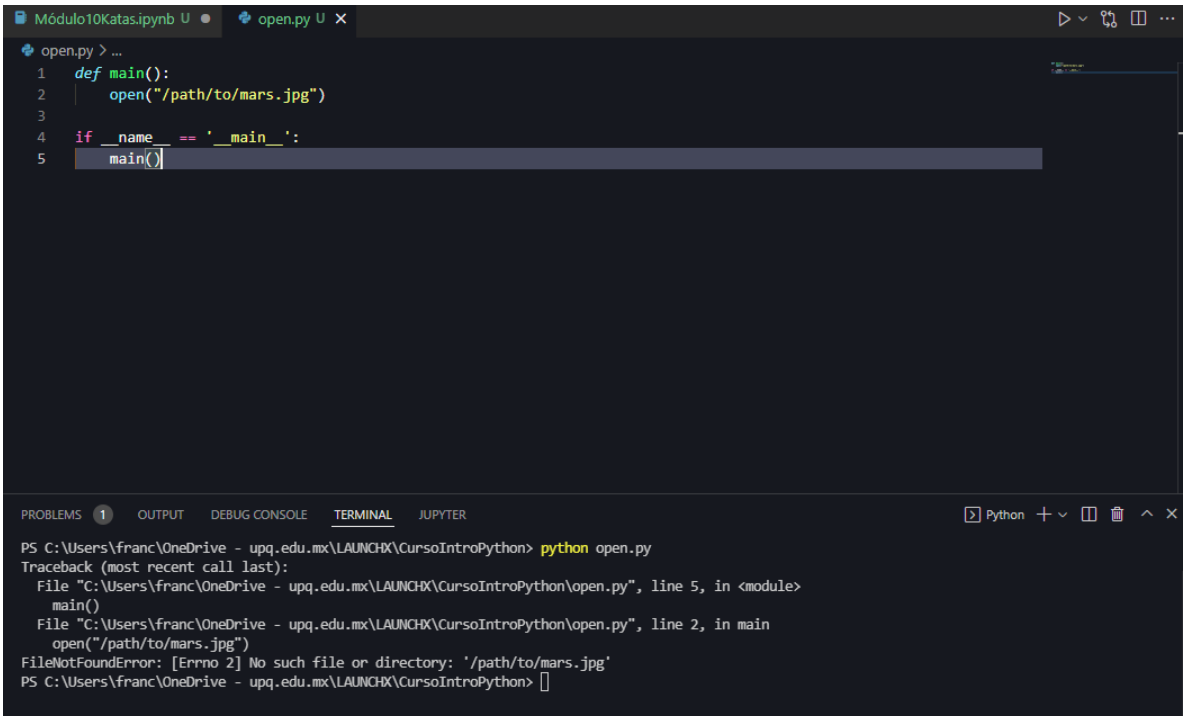
MÓDULO 10 – MANEJO DE ERRORES

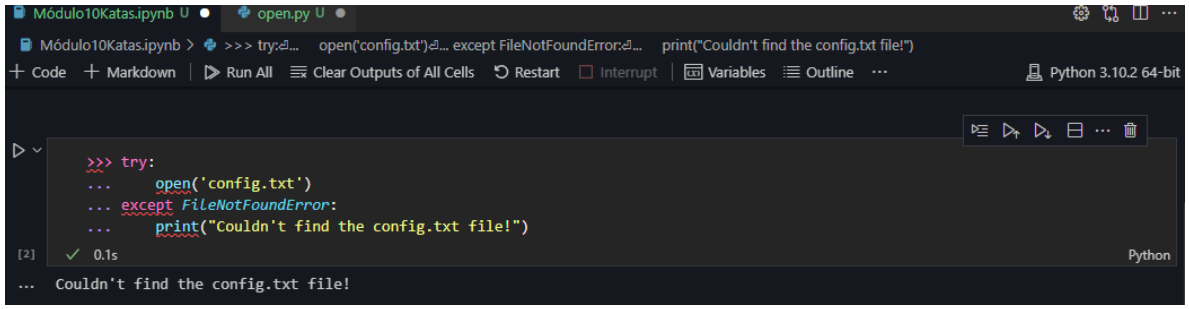
Tracebacks

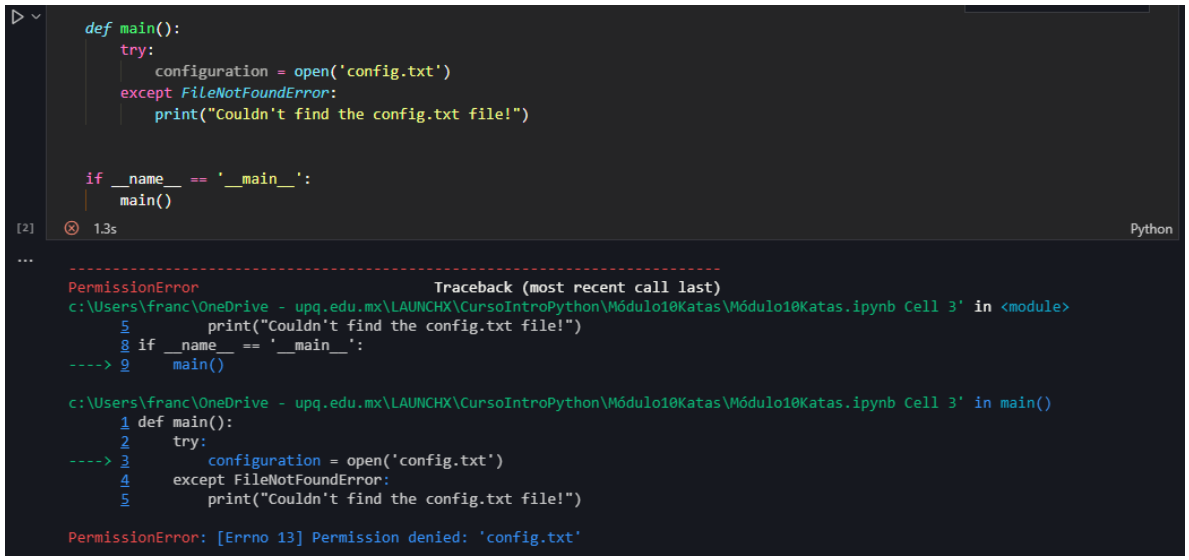


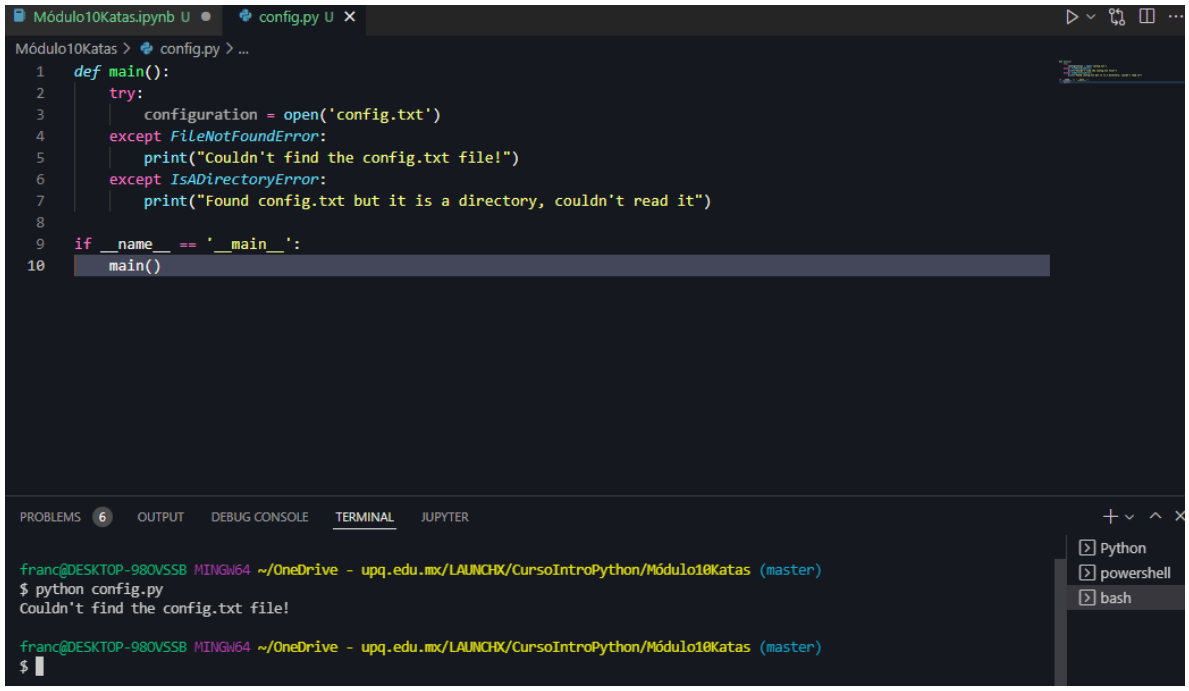


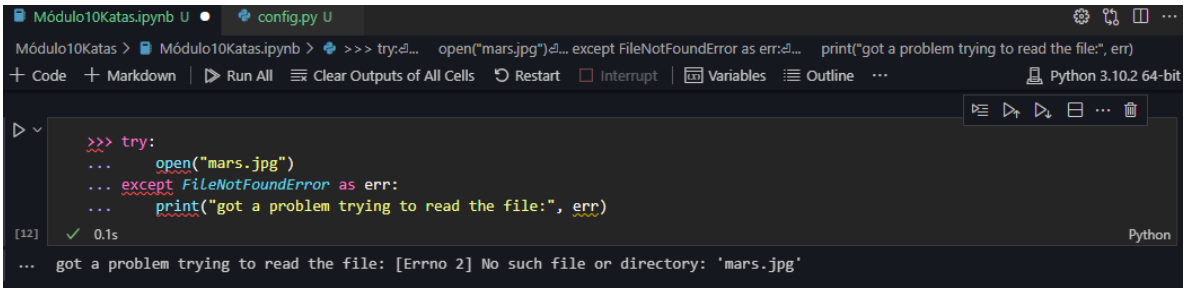
Controlando las excepciones

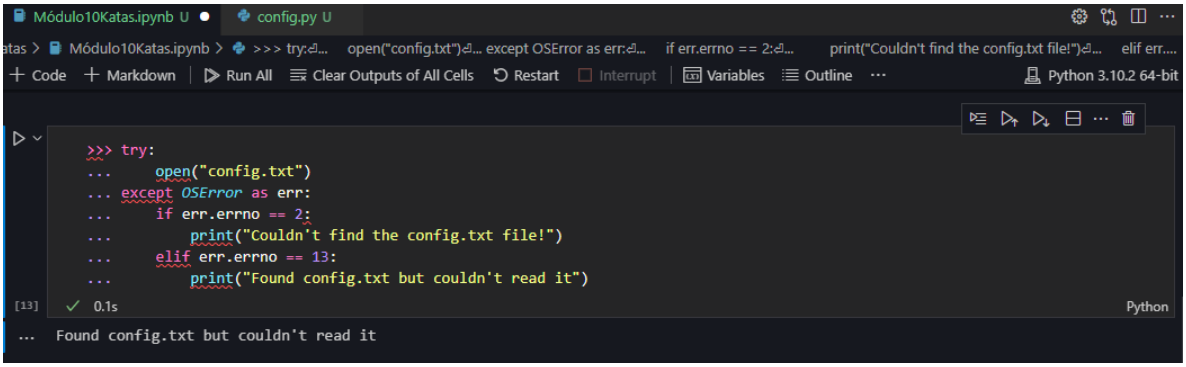
Try y Except de los bloques


Módulo10Katas.ipynb U
open.py U
Módulo10Katas.ipynb > >>> try:⏎... open('config.txt')⏎... except FileNotFoundError:⏎... print("Couldn't find the config.txt file!")
+ Code
+ Markdown
Run All
Clear Outputs of All Cells
Restart
Interrupt
Variables
Outline
Python 3.10.2 64-bit
>>> try:
... open('config.txt')
... except FileNotFoundError:
... print("Couldn't find the config.txt file!")
[2]
0.1s
Python
... Couldn't find the config.txt file!

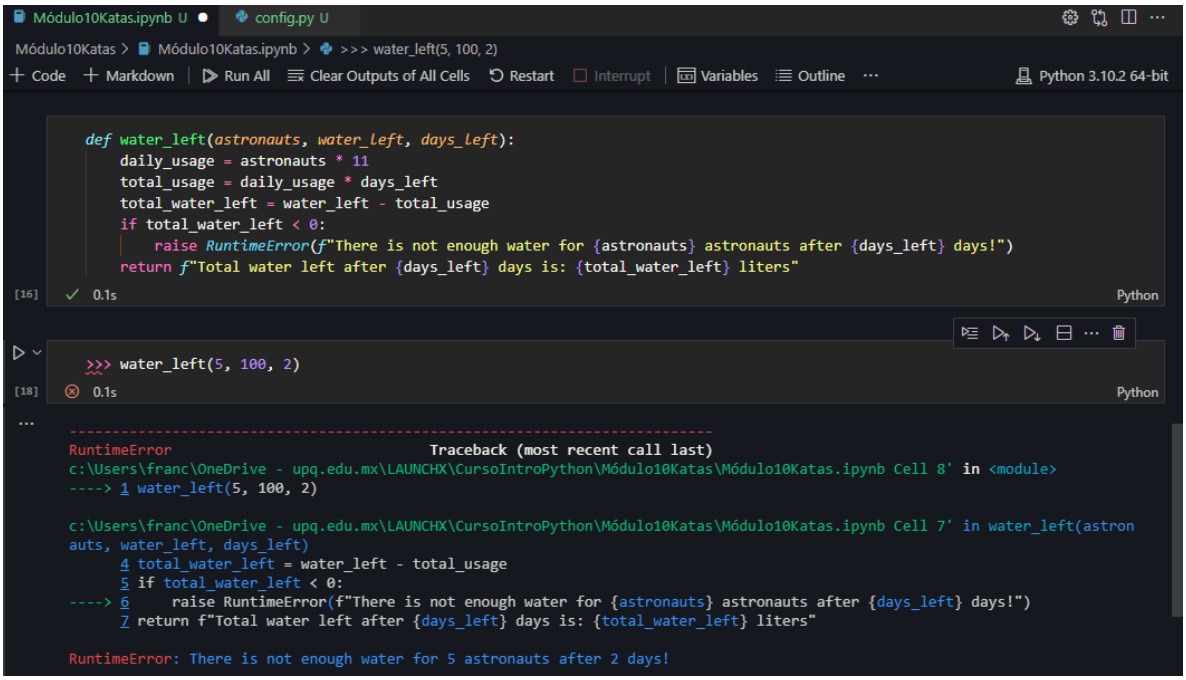

def main():
try:
configuration = open('config.txt')
except FileNotFoundError:
print("Couldn't find the config.txt file!")
if __name__ == '__main__':
main()
[2]
1.3s
Python
PermissionError Traceback (most recent call last)
c:\Users\franc\OneDrive - upq.edu.mx\LAUNCHX\CursoIntroPython\Módulo10Katas\Módulo10Katas.ipynb Cell 3' in <module>
5 print("Couldn't find the config.txt file!")
8 if __name__ == '__main__':
----> 9 main()
c:\Users\franc\OneDrive - upq.edu.mx\LAUNCHX\CursoIntroPython\Módulo10Katas\Módulo10Katas.ipynb Cell 3' in main()
1 def main():
2 try:
----> 3 configuration = open('config.txt')
4 except FileNotFoundError:
5 print("Couldn't find the config.txt file!")
PermissionError: [Errno 13] Permission denied: 'config.txt'

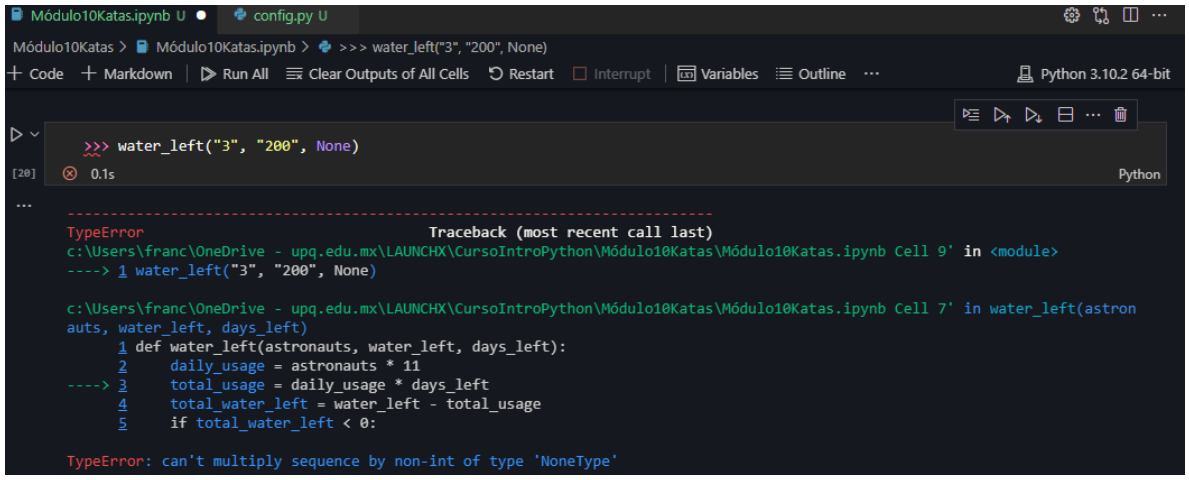

Módulo10Katas.ipynb U
config.py U
Módulo10Katas > config.py > ...
def main():
try:
configuration = open('config.txt')
except FileNotFoundError:
print("Couldn't find the config.txt file!")
except IsADirectoryError:
print("Found config.txt but it is a directory, couldn't read it")
if __name__ == '__main__':
main()
PROBLEMS 6
OUTPUT
DEBUG CONSOLE
TERMINAL
JUPYTER
Python
powershell
bash
franc@DESKTOP-98OVSSB MINGW64 ~/OneDrive - upq.edu.mx/LAUNCHX/CursoIntroPython/Módulo10Katas (master)
$ python config.py
Couldn't find the config.txt file!
franc@DESKTOP-98OVSSB MINGW64 ~/OneDrive - upq.edu.mx/LAUNCHX/CursoIntroPython/Módulo10Katas (master)
$

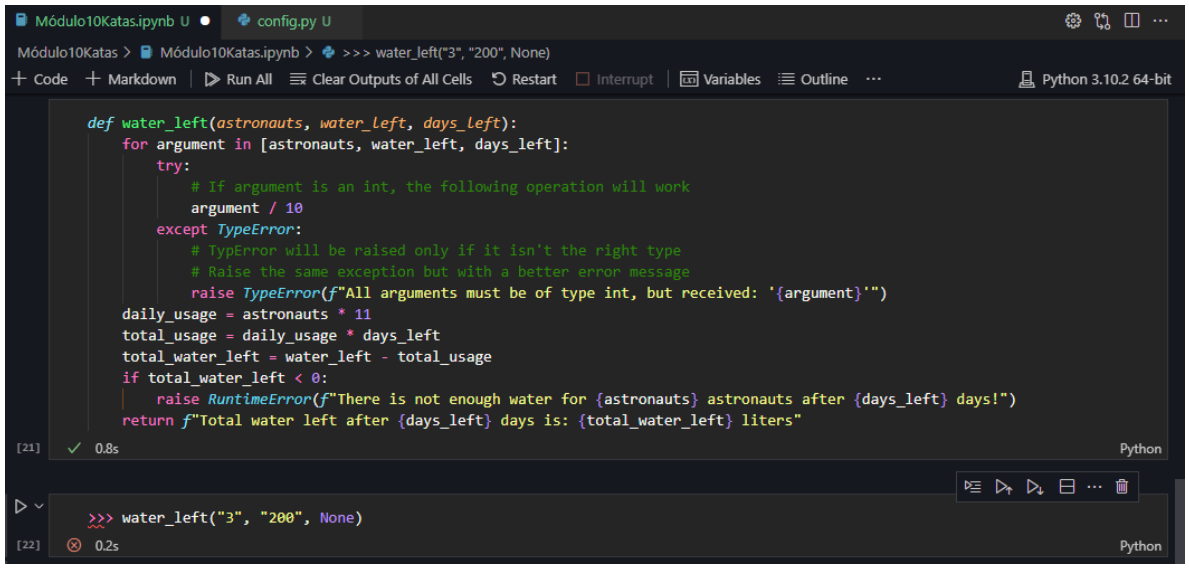


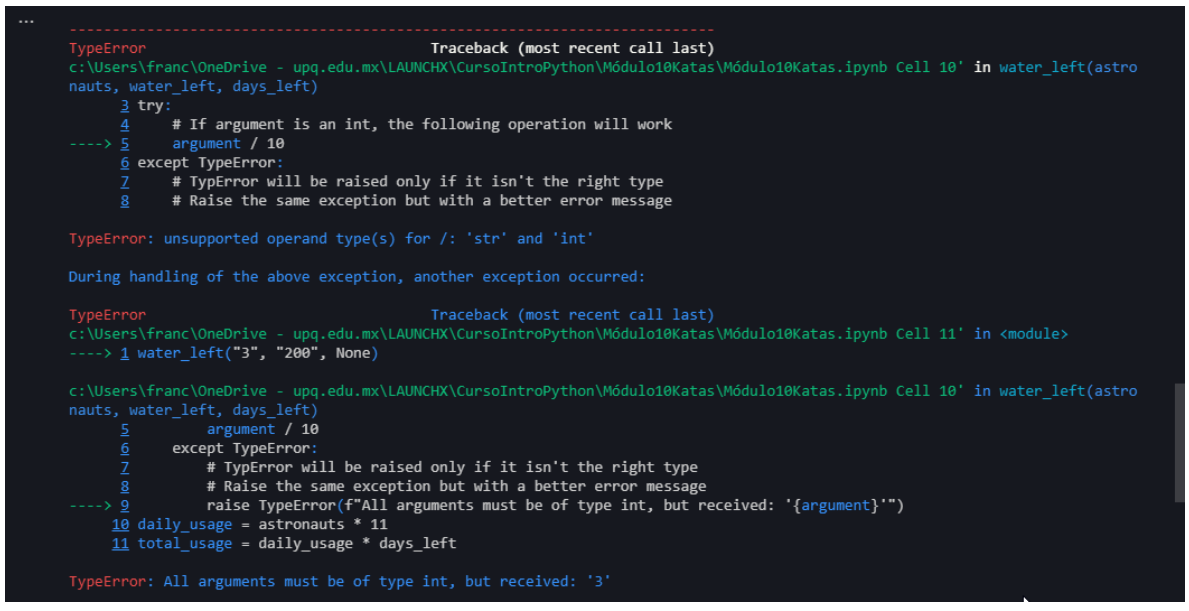


Generar excepciones




Módulo10Katas.ipynb U
config.py U
Módulo10Katas > Módulo10Katas.ipynb > >>> water_left("3", "200", None)
Code
Markdown
Run All
Clear Outputs of All Cells
Restart
Interrupt
Variables
Outline
Python 3.10.2 64-bit
>>> water_left("3", "200", None)
[20]
0.1s
Python
TypeError Traceback (most recent call last)
c:\Users\franc\OneDrive - upq.edu.mx\LAUNCHX\CursoIntroPython\Módulo10Katas\Módulo10Katas.ipynb Cell 9' in <module>
----> 1 water_left("3", "200", None)
c:\Users\franc\OneDrive - upq.edu.mx\LAUNCHX\CursoIntroPython\Módulo10Katas\Módulo10Katas.ipynb Cell 7' in water_left(astronauts, water_left, days_left)
1 def water_left(astronauts, water_left, days_left):
2 daily_usage = astronauts * 11
----> 3 total_usage = daily_usage * days_left
4 total_water_left = water_left - total_usage
5 if total_water_left < 0:
TypeError: can't multiply sequence by non-int of type 'NoneType'


Módulo10Katas.ipynb U
config.py U
Módulo10Katas > Módulo10Katas.ipynb > >>> water_left("3", "200", None)
Code
Markdown
Run All
Clear Outputs of All Cells
Restart
Interrupt
Variables
Outline
Python 3.10.2 64-bit
def water_left(astronauts, water_left, days_left):
for argument in [astronauts, water_left, days_left]:
try:
# If argument is an int, the following operation will work
argument / 10
except TypeError:
TypError will be raised only if it isn't the right type
Raise the same exception but with a better error message
raise TypeError(f"All arguments must be of type int, but received: '{argument}'")
daily_usage = astronauts * 11
total_usage = daily_usage * days_left
total_water_left = water_left - total_usage
if total_water_left < 0:
raise RuntimeError(f"There is not enough water for {astronauts} astronauts after {days_left} days!")
return f"Total water left after {days_left} days is: {total_water_left} liters"
[21]
0.8s
Python
>>> water_left("3", "200", None)
[22]
0.2s
Python


TypeError Traceback (most recent call last)
c:\Users\franc\OneDrive - upq.edu.mx\LAUNCHX\CursoIntroPython\Módulo10Katas\Módulo10Katas.ipynb Cell 10' in water_left(astronauts, water_left, days_left)
3 try:
4 # If argument is an int, the following operation will work
----> 5 argument / 10
6 except TypeError:
7 # TypError will be raised only if it isn't the right type
8 # Raise the same exception but with a better error message
TypeError: unsupported operand type(s) for /: 'str' and 'int'
During handling of the above exception, another exception occurred:
TypeError Traceback (most recent call last)
c:\Users\franc\OneDrive - upq.edu.mx\LAUNCHX\CursoIntroPython\Módulo10Katas\Módulo10Katas.ipynb Cell 11' in <module>
----> 1 water_left("3", "200", None)
c:\Users\franc\OneDrive - upq.edu.mx\LAUNCHX\CursoIntroPython\Módulo10Katas\Módulo10Katas.ipynb Cell 10' in water_left(astronauts, water_left, days_left)
5 argument / 10
6 except TypeError:
7 # TypError will be raised only if it isn't the right type
8 # Raise the same exception but with a better error message
----> 9 raise TypeError(f"All arguments must be of type int, but received: '{argument}'")
10 daily_usage = astronauts * 11
11 total_usage = daily_usage * days_left
TypeError: All arguments must be of type int, but received: '3'