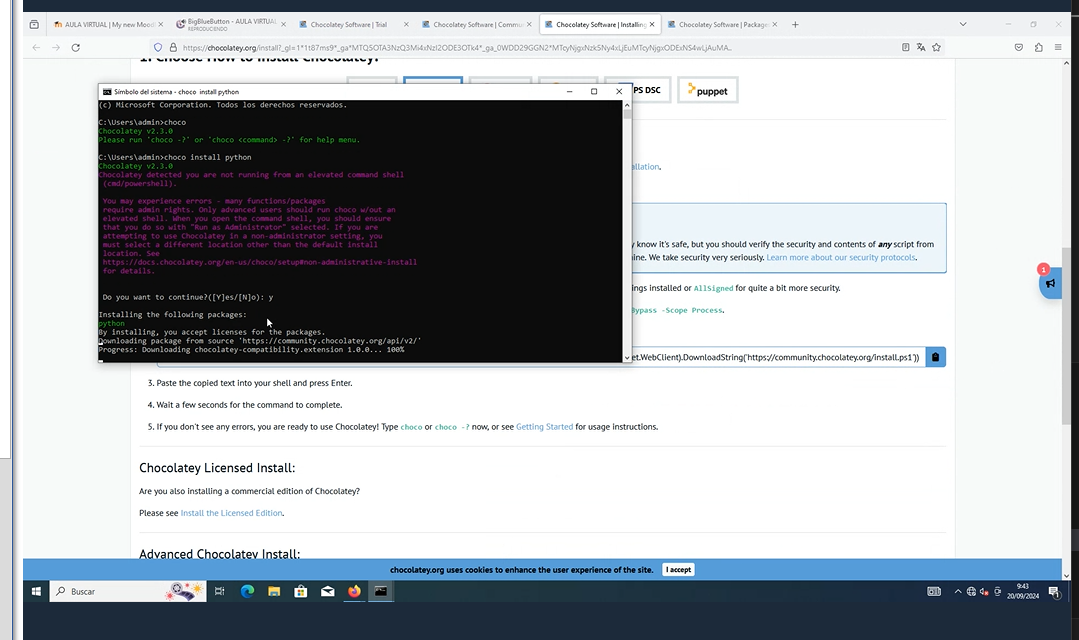
PYTHON 20\_09

CHOCOLATEY.ORG: para instalar programas en ordenadores y que sean las mismas vresiones en todos los ordenadores, se instala desde cmd y se puede automatizar.



find packages en commubnity, vs code>

>choco install vscode

Si ponemos en buscador Python:

>

Para instalar de powershell.

Hay que instalar chocolatey primero en el entorno.

Na pagina en Python es una gran cadena, se puede examinar con patrones regex por ej.

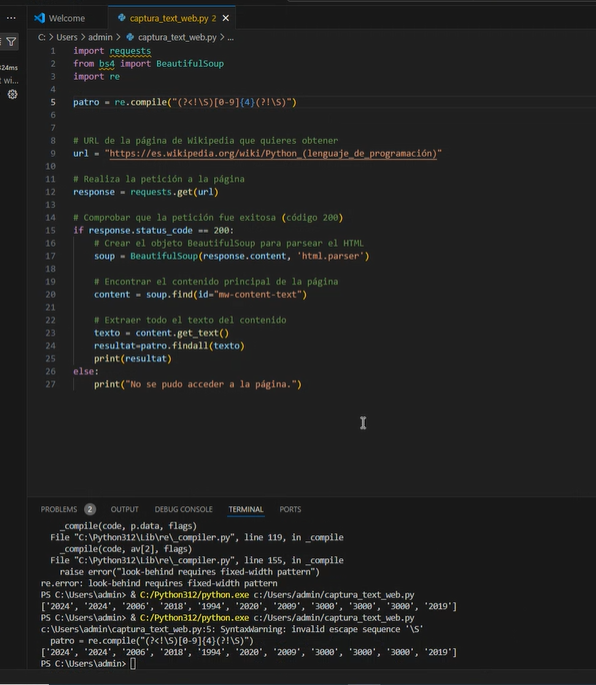
-----

En comando (cmd):

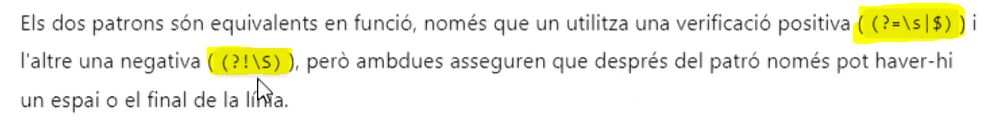
pip es instalador de Python

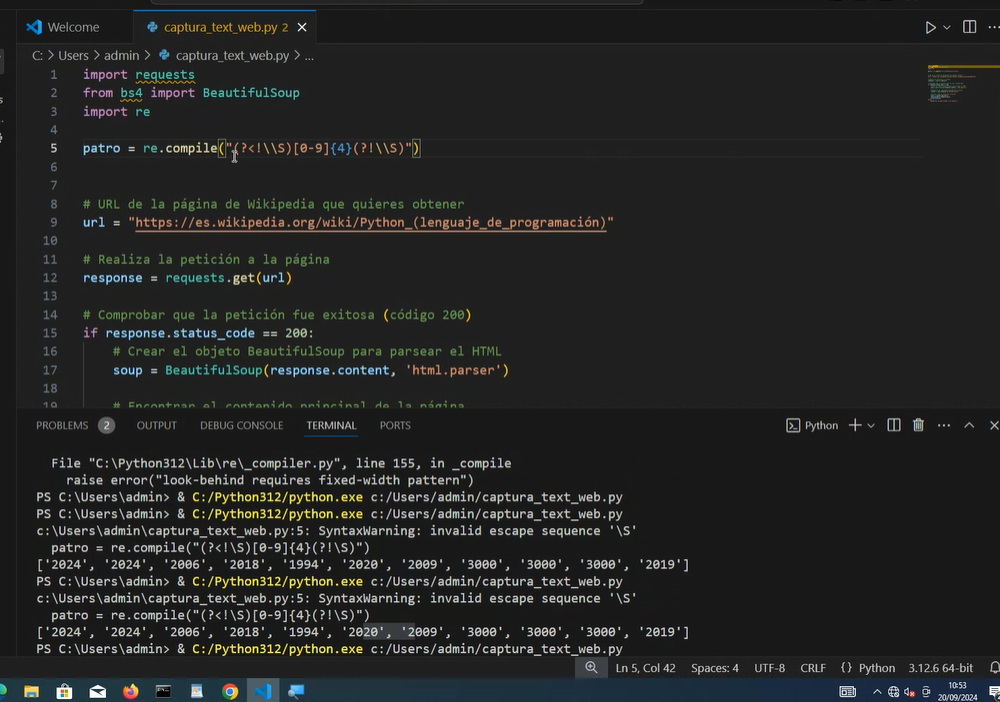
Pip install bs4   
(beautifull soup)

Agafa dates

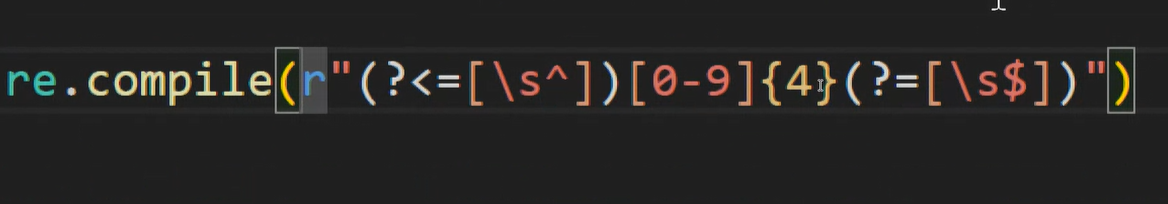


**(?<!S)**   
**(?<=\s|^)**  
look behind **negatiu (?<!S)**: equival al look behind **positiu** **(?<=\s|^)** verificat a la web regex101.



La **r** es raw (para raw string) lo diferente aquí y quitamos el warning.  
O ponemos barras invertidas dobles para quitar el warning de la consola:  


**(?=\s|$)  
(?=[\s$])  
Este es look ahead.**

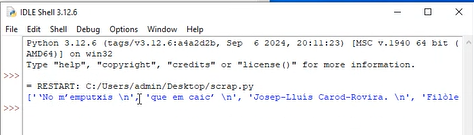
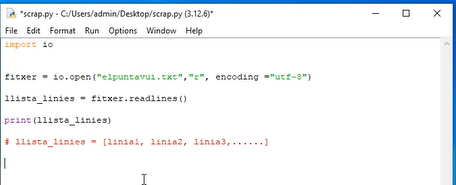


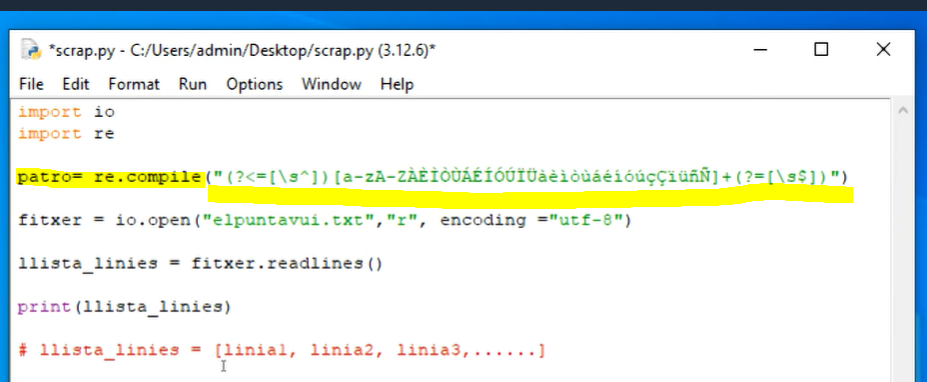
**Data mining y Top mining** (per buscar hashtags) **Scrapping:**

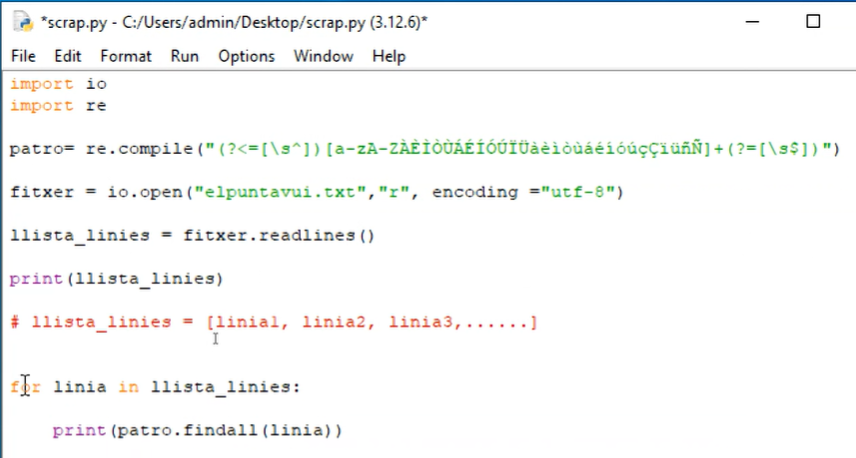
**Import io   
Io.open (arxiu.txt,”r”, encoding=”utf-8” )**

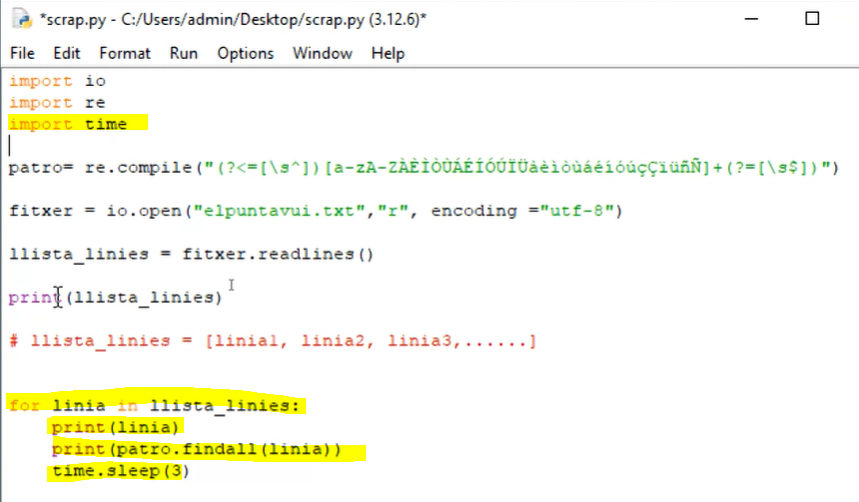
**r de read**

Ordenar per líneas: es una llista de líneas: **.readlines()**Cada element de la llista es una línea. Amb líneas es pot buscar els patrons.



**Ara nem a per les paraules amb el patro regex:   
dintre del modul re s ha de posar la doble barra abans la s** 



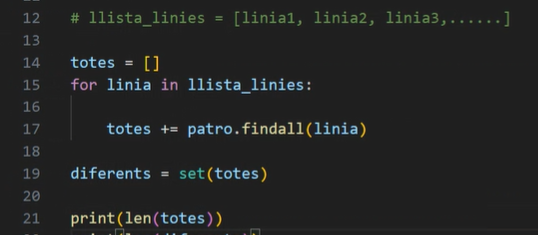


Usaremos Set.

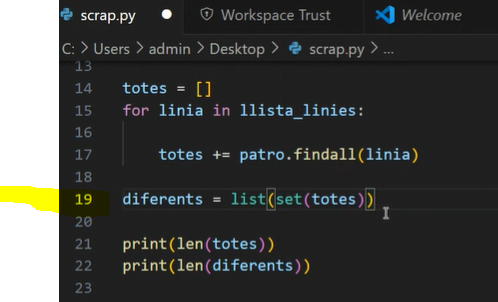
En Python, un **set** es una colección desordenada de elementos únicos. Esto significa que:

1. **No hay duplicados**: Los elementos de un set son únicos. Si intentas agregar un elemento que ya existe en el set, no se agregará.
2. **No tiene un orden específico**: A diferencia de las listas o las tuplas, los elementos de un set no están ordenados y no tienen un índice.
3. **Operaciones rápidas**: Los sets permiten realizar búsquedas, adiciones y eliminaciones de elementos de manera eficiente.

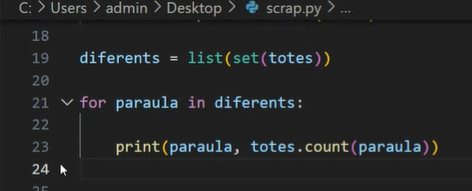
Puedes crear un set utilizando llaves {} o la función set().

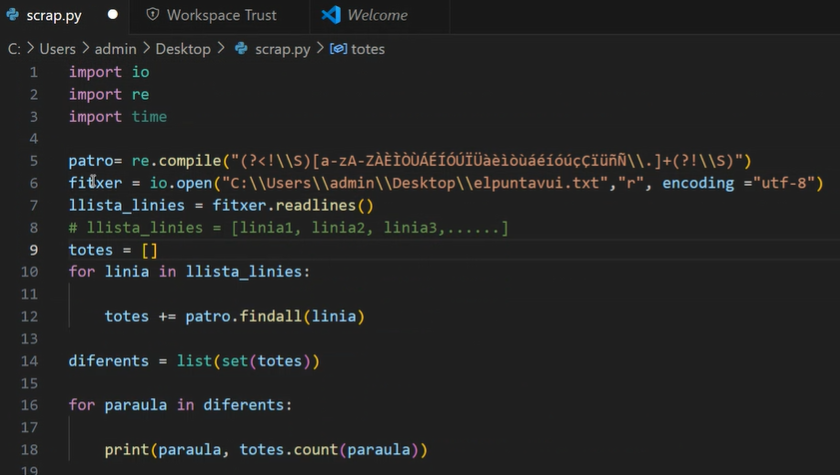


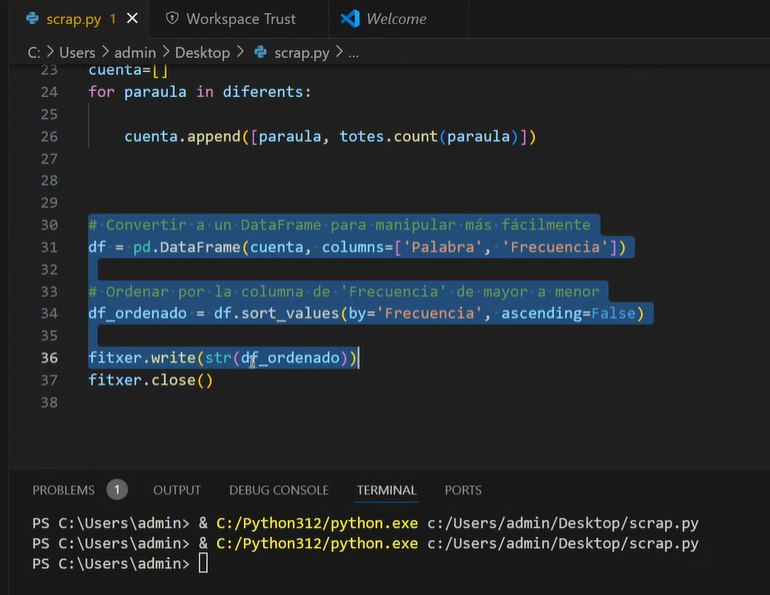
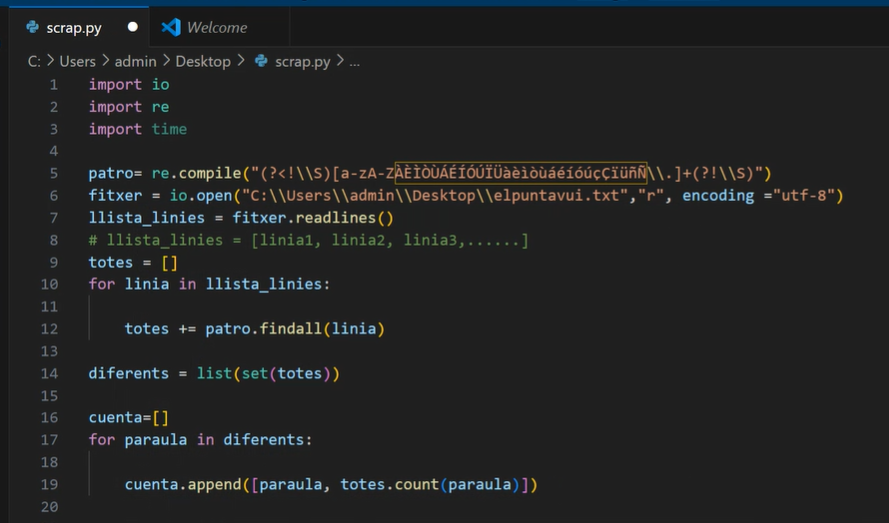
**Set no es una lista, hay qye pasarlo a lista: con índices ..**



**Es lo mismo a:**

 **imprime palabra y cuantas hay de la misma para todas.**

**SCRAP.py quedaría así:** 

**O PODEM FER:   
INICIAMOS cuenta list**

**Exlicacion del código por chatgpt:  
example of Python file: SCRAP**Código:  
import io

import re

import time

import pandas as pd

patro= re.compile("(?<!\\S)[a-zA-ZÀÈÌÒÙÁÉÍÓÚÏÜàèìòùáéíóúçÇïüñÑ\\.]+(?!\\S)")

fitxer = io.open("C:\\Users\\admin\\Desktop\\quintoelemento.txt","r", encoding ="utf-8")

llista\_linies = fitxer.readlines()

# llista\_linies = [linia1, linia2, linia3,......]

fitxer.close()

fitxer =io.open("C:\\Users\\admin\\Desktop\\resultado.txt","w",encoding="utf-8")

totes = []

for linia in llista\_linies:

totes += patro.findall(linia)

diferents = list(set(totes))

cuenta=[]

for paraula in diferents:

cuenta.append([paraula, totes.count(paraula)])

# Configuració per mostrar totes les files i columnes

pd.set\_option('display.max\_rows', None) # Mostrar totes les files

pd.set\_option('display.max\_columns', None) # Mostrar totes les columnes

pd.set\_option('display.expand\_frame\_repr', False) # Evitar que es trenqui en múltiples línies

# Us del Pandas Suggerit per la Chattie

df = pd.DataFrame(cuenta, columns=['Palabra', 'Frecuencia'])

# Ordenar por la columna de 'Frecuencia' de mayor a menor

df\_ordenado = df.sort\_values(by='Frecuencia', ascending=False)

fitxer.write(str(df\_ordenado))

fitxer.close()

The code you've written reads a text file, extracts words that match a given pattern (in this case, alphabetic words including accented characters), counts their frequency, and writes the resulting DataFrame with the word counts to an output file.

Here are some potential improvements or clarifications:

### Issues:

1. **Non-formatted output to file**: When writing the DataFrame df\_ordenado to the file, you're converting it to a string. This will output a less readable format. Instead, using df.to\_string() or df.to\_csv() would allow better control over the output formatting.
2. **Closing the file early**: You are appending the frequency results to the fitxer, but it's better to use the DataFrame methods for writing out the data, since you're using pandas.

### Improved Code:

### import io

### import re

### import time

### import pandas as pd

### # Regular expression to match words (including accented characters)

### patro = re.compile(r"(?<!\S)[a-zA-ZÀÈÌÒÙÁÉÍÓÚÏÜàèìòùáéíóúçÇïüñÑ\.]+(?!\S)")

### # Reading input file

### with io.open("C:\\Users\\admin\\Desktop\\quintoelemento.txt", "r", encoding="utf-8") as fitxer:

### llista\_linies = fitxer.readlines()

### # Collect all matches

### totes = []

### for linia in llista\_linies:

### totes += patro.findall(linia)

### # Getting unique words and counting occurrences

### diferents = list(set(totes))

### cuenta = [[paraula, totes.count(paraula)] for paraula in diferents]

### # Create DataFrame

### df = pd.DataFrame(cuenta, columns=['Palabra', 'Frecuencia'])

### # Sort DataFrame by 'Frecuencia' in descending order

### df\_ordenado = df.sort\_values(by='Frecuencia', ascending=False)

### # Write DataFrame to a file (better formatting with to\_string or to\_csv)

### with io.open("C:\\Users\\admin\\Desktop\\resultado.txt", "w", encoding="utf-8") as fitxer:

### fitxer.write(df\_ordenado.to\_string(index=False))

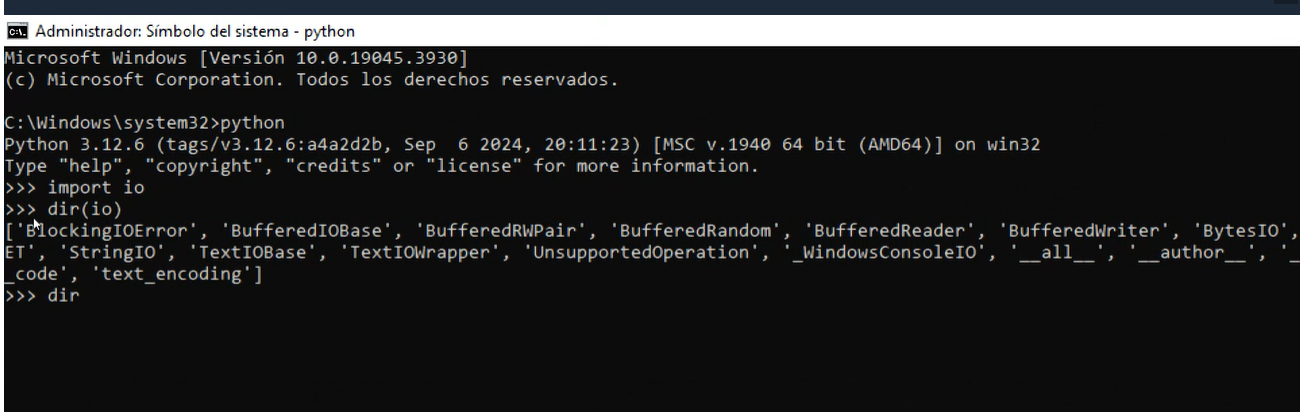
### # Optionally, if you want a CSV output instead of a text file

### # df\_ordenado.to\_csv("C:\\Users\\admin\\Desktop\\resultado.csv", index=False, sep=',', encoding='utf-8')

### Key changes:

1. **Used**with**for file handling**: This is a more Pythonic way to handle files as it automatically handles closing them.
2. to\_string()**method**: I used to\_string() to format the DataFrame more nicely in the output file. Optionally, you could switch to to\_csv() for a CSV format.
3. **Regular expression**: The patro pattern remains unchanged, but it's written using the raw string r"..." format to avoid the need to escape backslashes.
4. **Improved list comprehension**: The list comprehension for cuenta is a cleaner way of counting word occurrences.

This version will provide a clearer and better-formatted output in the resultado.txt.



**.io   
te dice todo de q va io.**

**Import io; help io >>ayudai**