Logo

Description automatically generatedLogo

Description automatically generated with medium confidence

*Practica: 03*

*Nombre de la Practica: Ejercicios de Pypass*

Instituto Politécnico Nacional.

Escuela Superior de Cómputo.

Licenciatura en ciencia de datos.

Nombre de la materia: Desarrollo de Aplicaciones para el Análisis de Datos

Grupo: 4AV1

Profesora: Sandra Luz Morales Guitron

Lopez Mendez Emiliano

INDICE

[INTRODUCCION 3](#_Toc179656781)

[DESAROLLO 3](#_Toc179656782)

[LISTAS 3](#_Toc179656783)

[DICCIONARIOS 26](#_Toc179656784)

[SETS 31](#_Toc179656785)

[FICHEROS 32](#_Toc179656786)

[CONCLUSIONES 35](#_Toc179656787)

# INTRODUCCION

En esta practica se llevo la instalación de un Editor de programación para poder programar con la versión mas actualizada de Python. Escogiendo la computadora que vamos a usar, el sistema operativa y el Editor que se usaran.

# DESAROLLO

Se llevo acabo el análisis y configuración de varios códigos ya adentro de pypass. Se modificaron para poder imprimir los resultados deseados y también para poder imprimir dichos programas se hizo un archivo aparte llamado args.py con las variables que se piden del main.

## LISTAS

1. max-value **El código encuentra y devuelve el valor máximo en una lista iterando a través de sus elementos y comparando cada uno con el máximo actual.**

A computer screen shot of a program

Description automatically generated

1. max-value-with-min **El código encuentra el valor máximo de una lista convirtiendo todos los números a sus valores opuestos (negativos), y luego encuentra el mínimo de esa lista de números opuestos, devolviendo su equivalente positivo como el máximo original.**

A screenshot of a computer program

Description automatically generated

1. min-value **El código encuentra y devuelve el valor mínimo en una lista iterando a través de sus elementos y comparando cada uno con el mínimo actual.**

A screenshot of a computer program

Description automatically generated

1. min-value-with-max **El código encuentra el valor mínimo de una lista convirtiendo los números a sus valores opuestos (negativos) y luego encontrando el máximo de esa lista de opuestos, devolviendo su valor equivalente positivo como el mínimo original.**

A screenshot of a computer program

Description automatically generated

1. remove-dups **El código elimina los elementos duplicados de una lista, manteniendo solo la primera aparición de cada número, y devuelve una nueva lista sin repeticiones.**

**A screen shot of a computer

Description automatically generated**

1. flatten-list **El código aplana una lista anidada, es decir, convierte cualquier sublista dentro de la lista original en una lista plana, manteniendo todos los elementos en un solo nivel.**

A screenshot of a computer program

Description automatically generated

1. remove-consecutive-dups **El código elimina los elementos consecutivos duplicados de una lista, devolviendo una nueva lista donde solo se mantiene una aparición de cada elemento consecutivo repetido.**

A computer screen shot of a program code

Description automatically generated

1. all-same **El código verifica si todos los elementos de una lista son iguales y devuelve `True` si lo son, o `False` si no lo son.**

A screen shot of a computer program

Description automatically generated

1. sum-diagonal **El código calcula la suma de los elementos de la diagonal principal de una matriz cuadrada (misma cantidad de filas y columnas). Si la matriz no es cuadrada, devuelve `None`.**

A screen shot of a computer program

Description automatically generated

1. powers2 **El código genera una lista de potencias de 2, desde \(2^0\) hasta \(2^n\), donde \(n\) es un valor dado, y devuelve esa lista. En este caso, se genera una lista de potencias de 2 hasta \(2^{10}\).**

A screen shot of a computer program

Description automatically generated

1. dec2bin **El código convierte un número decimal a su representación binaria y la devuelve como una cadena. En este caso, convierte el número decimal 56 en su equivalente binario "111000".**

A screenshot of a computer program

Description automatically generated

1. sum-mixed **El código convierte los elementos de una lista (que pueden ser cadenas o enteros) a enteros y luego devuelve la suma de todos esos valores. En este caso, convierte los elementos de la lista `['1', '2', 3, '4', 5]` a enteros y calcula la suma, que es 15**.

A screenshot of a computer screen

Description automatically generated

1. n-multiples **El código genera y muestra una lista con los primeros ( n ) múltiplos de un número ( x ). En este caso, para ( x = 2 ) y ( n = 4 ), el resultado es la lista `[2, 4, 6, 8]`, que son los primeros 4 múltiplos de 2.**

A screenshot of a computer program

Description automatically generated

1. drop-even **El código elimina los elementos en posiciones pares de una lista, manteniendo solo los que están en posiciones impares. En este ejemplo, para la lista `['U', 'V', 'W', 'X', 'Y']`, el resultado es `['V', 'X']`, que son los elementos en posiciones impares.**

A screen shot of a computer program

Description automatically generated

1. nth-power **El código toma una lista y un índice ( n ), y devuelve el valor del elemento en la posición ( n ) elevado a la potencia de ( n ). Si el índice ( n ) está fuera del rango de la lista, devuelve -1. En el ejemplo, para la lista `[6, 3, 8, 4]` y el índice 2, el resultado es ( 8^2 = 64 ).**

A screenshot of a computer program

Description automatically generated

1. name-initials **El código toma un nombre completo en formato "apellido, nombre", separa el apellido y el nombre, y luego genera las iniciales de ambos. Las iniciales del nombre son la primera letra del nombre en mayúscula seguida de un punto, y las iniciales del apellido se generan tomando la primera letra de cada palabra del apellido en mayúscula, también seguidas de un punto. En el ejemplo, para "van Rossum, Guido", el resultado es "G.V.R.".**

A screen shot of a computer program

Description automatically generated

1. non-consecutive **El código revisa una lista de números para encontrar el primer número que no sea consecutivo respecto al anterior. Si encuentra un número no consecutivo, lo devuelve. Si todos los números son consecutivos, devuelve `None`. En el ejemplo, para la lista `[1, 2, 4, 5]`, el resultado es `4` porque 4 no sigue a 3. Para la lista `[1, 2, 3, 4]`, devuelve `None` porque todos los números son consecutivos.**

A screen shot of a computer program

Description automatically generated

1. mult-reduce **El código utiliza la función `reduce` de la biblioteca `functools` para multiplicar todos los elementos de una lista. En este caso, toma la lista `[9, 3, 4, 2]` y calcula el producto de sus elementos, que es 216.**

A screenshot of a computer program

Description automatically generated

1. digit-rev-list **El código toma un número entero, lo convierte en una cadena de texto, invierte esa cadena y luego convierte cada dígito de la cadena invertida de vuelta a un entero. Finalmente, imprime la lista de dígitos invertidos. En el ejemplo, el número `56442` se convierte en la lista `[2, 4, 4, 6, 5]`.**

A screenshot of a computer program

Description automatically generated

1. time-plus-minutes **El código toma una hora en formato de cadena (`hora\_entrada`) y una cantidad de minutos, y luego suma esos minutos a la hora inicial utilizando la clase `timedelta` de la biblioteca `datetime`. Finalmente, convierte y muestra la nueva hora en formato `HH:MM`. En este ejemplo, a la hora `17:15` se le suman 240 minutos, resultando en la hora `21:15`.**

A screen shot of a computer program

Description automatically generated

1. add-positives **El código suma únicamente los valores positivos de una lista de números. En este ejemplo, la lista `[6, 3, 0, -1, -7, 5]` tiene los números positivos `6`, `3` y `5`, cuya suma es `14`.**

A screen shot of a computer

Description automatically generated

1. add-opposites **El código calcula la suma de los valores opuestos (negativos) de los elementos de una lista. En este caso, para la lista `[6, 3, 0, -1, -7, 5]`, el código toma los valores opuestos, es decir, `[-6, -3, 0, 1, 7, -5]`, y luego suma esos valores, obteniendo `-6` como resultado.**

A screenshot of a computer program

Description automatically generated

1. descending-numbers **El código genera una lista de números en orden descendente desde un valor \( n \) hasta 1. En este ejemplo, para \( n = 5 \), la lista generada es `[5, 4, 3, 2, 1]`.**

A screen shot of a computer

Description automatically generated

1. merge-sorted **El código combina dos listas ordenadas en una sola lista ordenada sin duplicados, comparando sus elementos y agregándolos a la nueva lista de manera eficiente.**

A screen shot of a computer program

Description automatically generated

1. trimmed-add **El código calcula la suma de los elementos de una lista excluyendo el valor máximo y mínimo, devolviendo 0 si la lista tiene menos de 3 elementos.**

A screenshot of a computer program

Description automatically generated

1. wolves **El código evalúa la posición de un lobo en una lista de animales y, dependiendo de su ubicación respecto a las ovejas, emite una advertencia indicando si el lobo está al final o si se va a comer a la oveja más cercana.**

A screenshot of a computer program

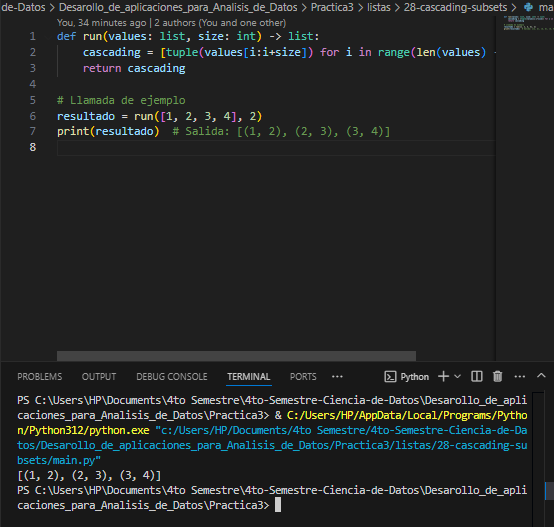
Description automatically generated

1. minmax **El código encuentra y devuelve el valor mínimo y máximo de una lista de números, devolviendo `None` en ambos casos si la lista está vacía.**

A screen shot of a computer program

Description automatically generated

1. cascading-subsets **El código crea y devuelve una lista de tuplas que contienen subconjuntos consecutivos de la lista original, cada uno con el tamaño especificado (`size`).**



1. diff-cuboid **El código calcula la diferencia absoluta entre los volúmenes de dos cuboides representados por dos listas y devuelve dicha diferencia.**

A screenshot of a computer program

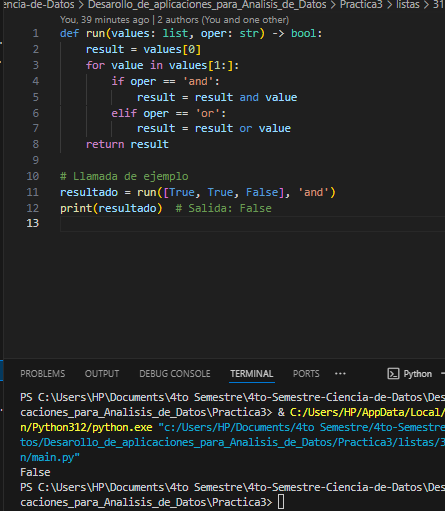
Description automatically generated

1. fl-strip **El código elimina el primer y último número de una cadena de números separados por comas y devuelve los números restantes como una cadena con espacios en lugar de comas.**

A screenshot of a computer program

Description automatically generated

1. logical-chain **El código aplica operaciones lógicas (`and` o `or`) a una lista de valores booleanos y devuelve el resultado final de la operación.**



1. first-unused-id **El código encuentra y devuelve el primer número entero positivo que no está presente en la lista dada.**

A screenshot of a computer screen

Description automatically generated

1. find-odds **El código filtra y devuelve una lista con solo los números impares de la lista original.**

A screenshot of a computer program

Description automatically generated

1. chemistry **El código verifica si una lista de números cumple con una serie de reglas lógicas específicas y devuelve `True` si todas las condiciones se cumplen o `False` en caso contrario.**

A screenshot of a computer program

Description automatically generated

1. next-next **El código busca un elemento específico en una lista y, si lo encuentra, devuelve el elemento siguiente a él; si no lo encuentra o está al final de la lista, devuelve `None`.**

A screen shot of a computer program

Description automatically generated

1. v-partition **El código divide una lista de números en dos listas: una con los valores menores que un valor de referencia (`ref\_value`) y otra con los valores mayores o iguales a ese valor de referencia.**

A screen shot of a computer program

Description automatically generated

1. attach-len **El código toma una cadena de texto, la divide en palabras, calcula la longitud de cada palabra, y devuelve una lista de cadenas donde cada palabra está seguida por su longitud. Por ejemplo, en el texto `"Hola a todos como estan"`, devuelve `['Hola:4', 'a:1', 'todos:5', 'como:4', 'estan:5']`.**

A screen shot of a computer program

Description automatically generated

1. reversing-words **El código invierte el orden de las palabras en una cadena de texto. Por ejemplo, para la entrada `"Hola como estan"`, devuelve `"estan como Hola"`.**

A screen shot of a computer program

Description automatically generated

1. barycenter **El código calcula el baricentro (centroide) de un triángulo dado por tres puntos ( A ), ( B ) y ( C ) en un plano. El baricentro se encuentra promediando las coordenadas ( x ) y ( y ) de los tres vértices. En el ejemplo, los puntos son ( A = [4, 6] ), ( B = [12, 4] ), y ( C = [10, 10] ), y el baricentro calculado es aproximadamente ( (8.67, 6.67) ).**

A screenshot of a computer program

Description automatically generated

1. flatten-list-deep **El código aplana una lista que puede contener sublistas de cualquier profundidad, es decir, convierte todas las sublistas anidadas en una única lista plana. En el ejemplo dado, la lista `[3, [6, [4, 7], 9], 12]` se convierte en `[3, 6, 4, 7, 9, 12]`.**

A screenshot of a computer program

Description automatically generated

1. first-duplicated **El código encuentra y devuelve el primer valor duplicado en una lista. Utiliza un conjunto para rastrear los elementos que ya han aparecido y, en cuanto encuentra un valor repetido, lo devuelve. En el ejemplo dado, el primer valor repetido es `3`.**

A screen shot of a computer program

Description automatically generated

1. fill-values **El código completa una lista insertando los números faltantes en secuencia. Si algún número en la secuencia no está en la lista, lo agrega en su lugar correspondiente. En el ejemplo dado, la lista `[0, 1, 3, 5]` se completa para incluir los números faltantes, resultando en `[0, 1, 2, 3, 4, 5]`.**

A screenshot of a computer program

Description automatically generated

1. frange **El código genera una lista de números flotantes que empieza en un valor dado (`start`), termina en otro valor (`stop`), y avanza en incrementos definidos por un paso (`step`). En este ejemplo, la lista generada desde `0` hasta `1` con un paso de `0.21` es `[0, 0.21, 0.42, 0.63, 0.84]`.**

A screen shot of a computer program

Description automatically generated

1. mul-matrix **El código realiza la multiplicación de dos matrices AAA y BBB. Primero verifica si el número** **de columnas de AAA coincide con el número de filas de BBB, ya que esta es una condición necesaria para realizar la multiplicación de matrices. Si la condición no se cumple, devuelve None. Si la condición se cumple, el código calcula el producto de matrices y devuelve el resultado.** A screenshot of a computer program

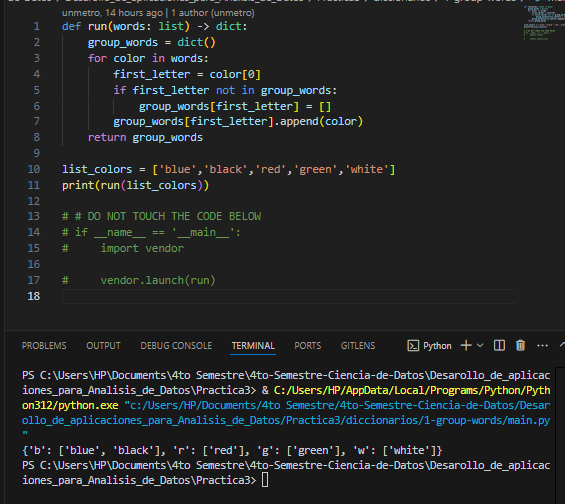
   Description automatically generated

A screenshot of a computer program

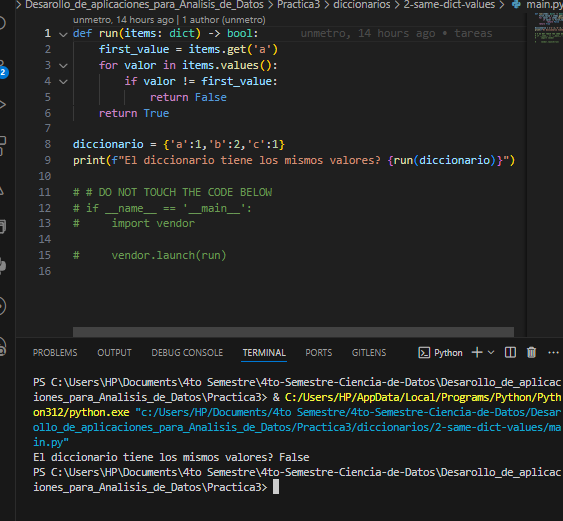
Description automatically generated

## DICCIONARIOS

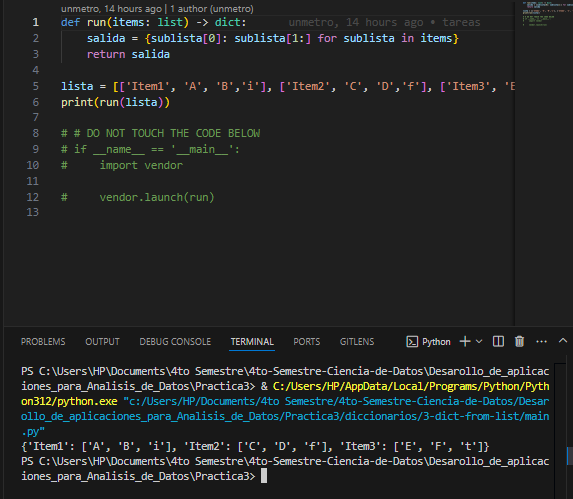
1. group-words **El código agrupa palabras de una lista según su primera letra y devuelve un diccionario donde las claves son las primeras letras y los valores son listas de palabras que comienzan con esa letra.**



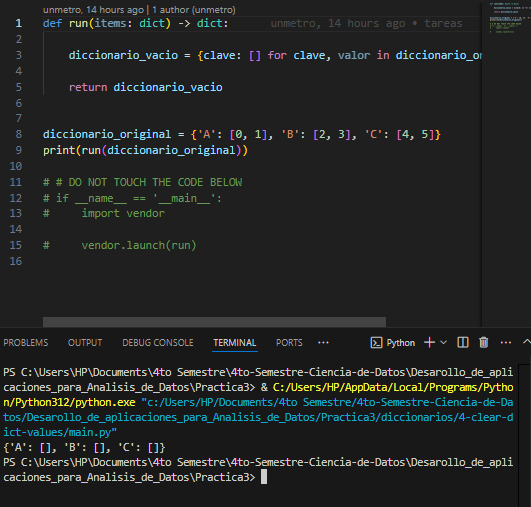
1. same-dict-values **El código verifica si todos los valores de un diccionario son iguales, devolviendo `True` si todos son iguales o `False` si hay diferencias.**



1. dict-from-list **El código convierte una lista de sublistas en un diccionario, usando el primer elemento de cada sublista como clave y el resto de los elementos como su valor.**



1. clear-dict-values **El código crea un nuevo diccionario con las mismas claves del diccionario original, pero con valores vacíos (listas vacías) para cada clave.**



1. fix-keys **El código limpia las claves de un diccionario eliminando los espacios en blanco de las claves originales y luego crea un nuevo diccionario con las claves modificadas y los valores originales. En el ejemplo, para el diccionario `{"key 1": "value 1", "key 2": "value 2"}`, el código produce un nuevo diccionario `{"key1": "value 1", "key2": "value 2"}`, eliminando los espacios de las claves.**

A screen shot of a computer program

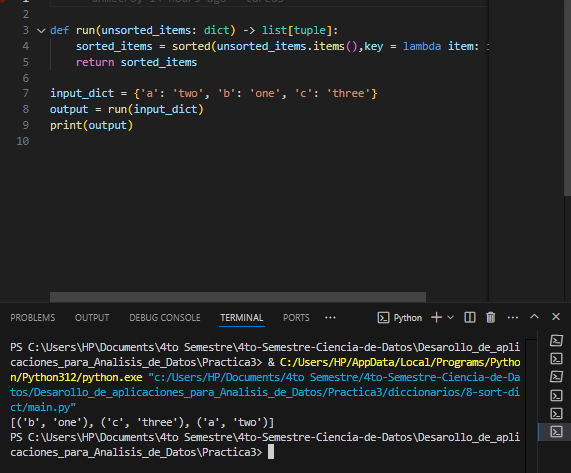
Description automatically generated

1. order-stock **El código verifica si hay suficiente stock de un artículo en un diccionario de inventario. Si el artículo existe y la cantidad requerida está disponible, devuelve `True`; de lo contrario, devuelve `False`. En este ejemplo, el diccionario de inventario es `{'pen': 20, 'cup': 11, 'keyring': 40}`, y se verifica si hay 9 unidades de "cup" (taza) disponibles. El resultado es `True` porque hay 11 unidades y se requieren solo 9.**

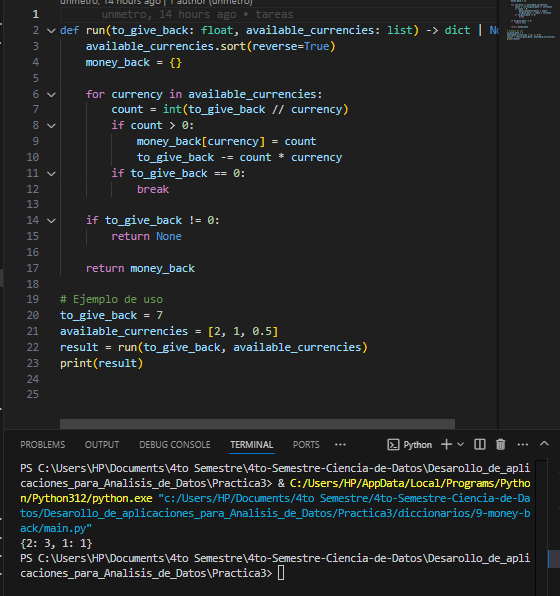
A screen shot of a computer

Description automatically generated

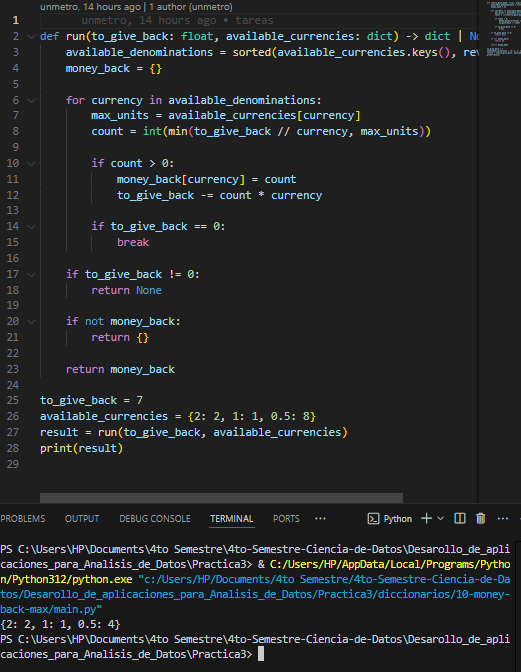
1. sort-dict **El código ordena los elementos de un diccionario según sus valores y devuelve una lista de tuplas con las claves y valores ordenados.**



1. money-back **El código calcula la cantidad de monedas necesarias para devolver un monto específico usando una lista de denominaciones disponibles y devuelve un diccionario con el número de cada denominación utilizada.**



1. money-back-max **El código calcula el cambio necesario utilizando un número máximo permitido de cada denominación disponible y devuelve un diccionario con la cantidad de cada denominación utilizada para alcanzar el monto deseado.**



1. first-ntimes
2. fix-id
3. dict-pull

## SETS

1. is-binary **checa si el numero dado es un numero binario o no**

A screenshot of a computer program

Description automatically generated

## FICHEROS

1. wc **El código abre un archivo de texto, cuenta el número de líneas, palabras y bytes, y muestra estos valores, manejando un error si el archivo no se encuentra.**

A computer screen shot of a program code

Description automatically generated

1. read-csv **El código lee un archivo CSV, convierte los valores de cada fila a su tipo correspondiente (enteros, booleanos o `None`), y devuelve una lista de diccionarios donde cada fila del CSV es un diccionario con claves y valores convertidos según su tipo. Luego imprime cada fila convertida.**

A computer screen shot of a program

Description automatically generated

1. txt2md **El código convierte un archivo de texto con guiones y tabulaciones en formato de encabezados de Markdown. Los guiones o tabulaciones indican el nivel del encabezado en Markdown. El código lee el archivo de entrada, transforma las líneas según el formato de encabezado de Markdown, y luego escribe el resultado en un archivo de salida.**

**- Los niveles de encabezado se determinan por la cantidad de tabulaciones o espacios al inicio de cada línea.**

**- La función añade el símbolo `#` seguido del título correspondiente según el nivel de indentación detectado.**

**El archivo `prueba.txt` es leído, transformado y luego guardado como `guion.txt`.**

A screen shot of a computer program

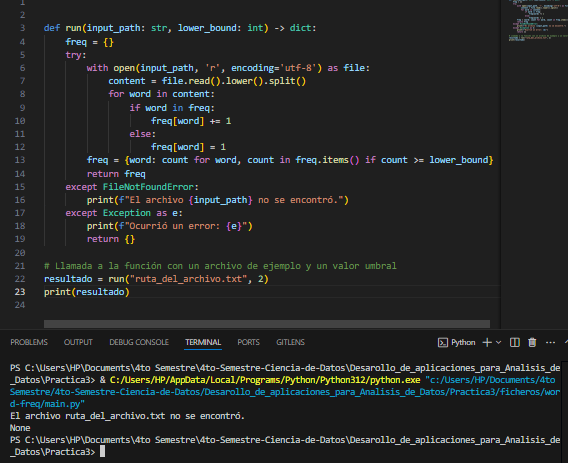
Description automatically generated

1. avg-temps **El código lee un archivo de temperaturas mensuales, calcula el promedio de cada mes y guarda los resultados en un archivo de salida.**

A screen shot of a computer program

Description automatically generated

1. find-words
2. sum-matrix
3. longest-word
4. word-freq **El código cuenta la frecuencia de cada palabra en un archivo de texto y devuelve un diccionario con las palabras que aparecen más veces que un umbral especificado (lower\_bound).**

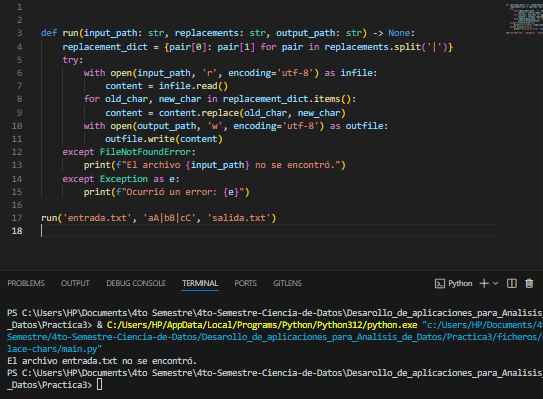


1. get-line **El código lee una línea específica de un archivo de texto y devuelve su contenido o None si la línea no existe o el archivo no se encuentra.**

**A screen shot of a computer program

Description automatically generated**

1. replace-chars **El código reemplaza caracteres específicos en el contenido de un archivo de texto y guarda el resultado en otro archivo de salida, manejando posibles errores si el archivo de entrada no se encuentra.**



1. histogram-txt
2. submarine
3. common-words

# CONCLUSIONES

Durante el desarrollo de aproximadamente 46 programas, se trabajó principalmente con métodos básicos como append para listas y lower y strip para manipulación de cadenas de texto. Además, se utilizaron métodos como istitle para verificar la capitalización de las cadenas y path.exists para verificar rutas de archivos. También se implementaron funciones relacionadas con el manejo de rutas (path.abspath, path.join) y entorno (environ.get), así como funciones de visualización (rich.print) y manejo de módulos (runpy.run\_path). Estos ejercicios no introdujeron conceptos nuevos, sino que se enfocaron en reforzar el conocimiento existente de estos métodos comunes. Fue un excelente repaso después de las vacaciones, permitiendo retomar la práctica de la programación tras un periodo de inactividad.