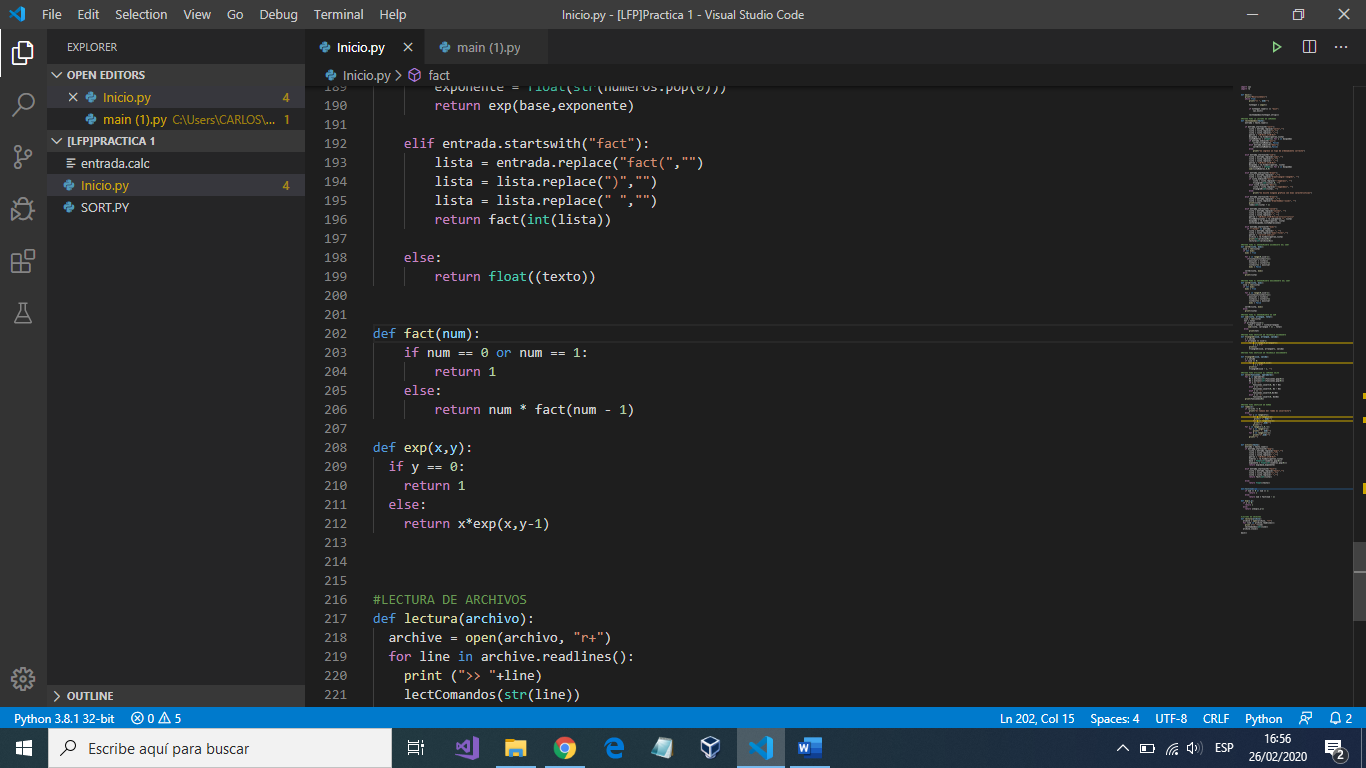
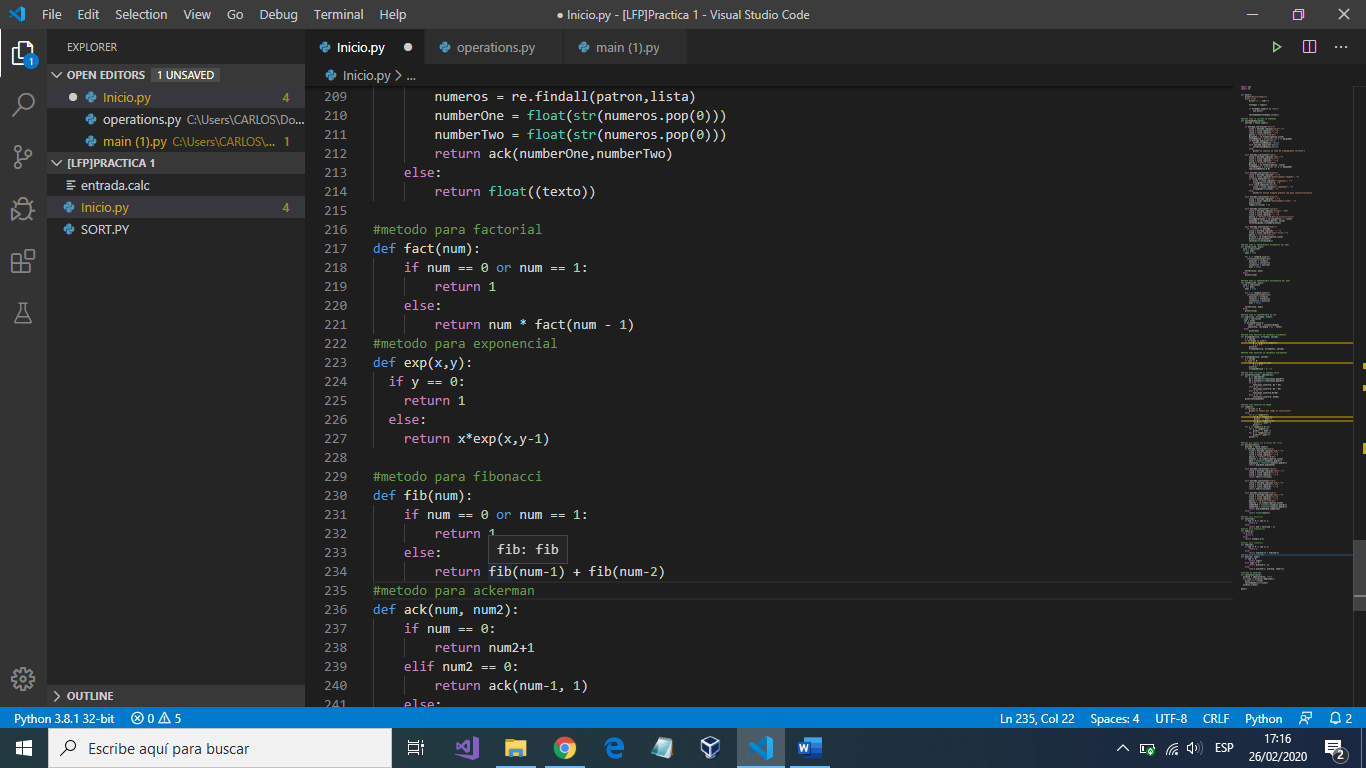
**OPERACIONES:**

**EXPONENCIAL:**



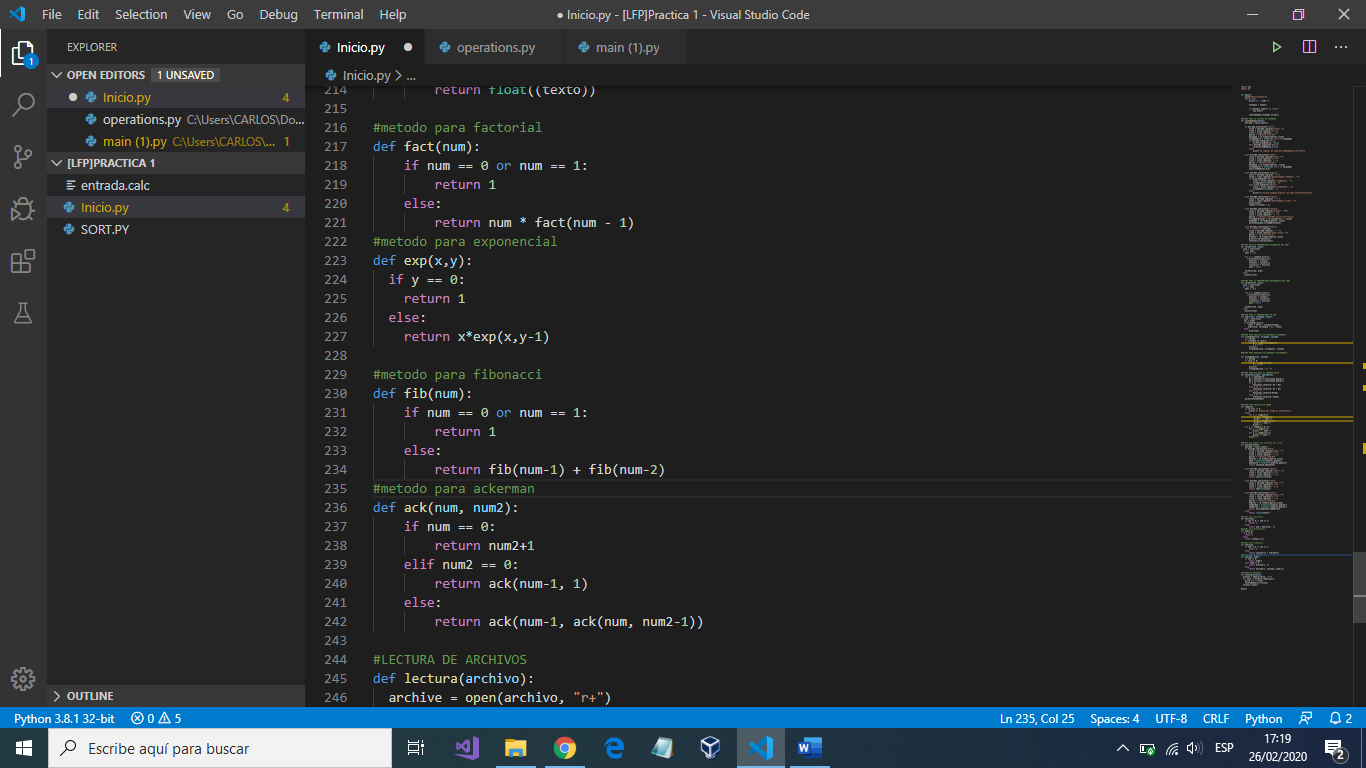
Este método sirve para hacer la operación exponencial, el cual utiliza dos parámetros en x,y, el x representa la base del numero y el y la potencia, donde se multiplica la x en si misma las veces que tiene el tamaño de la y utilizando la recursividad restando 1 a la potencia.

**FIBONACCI:**



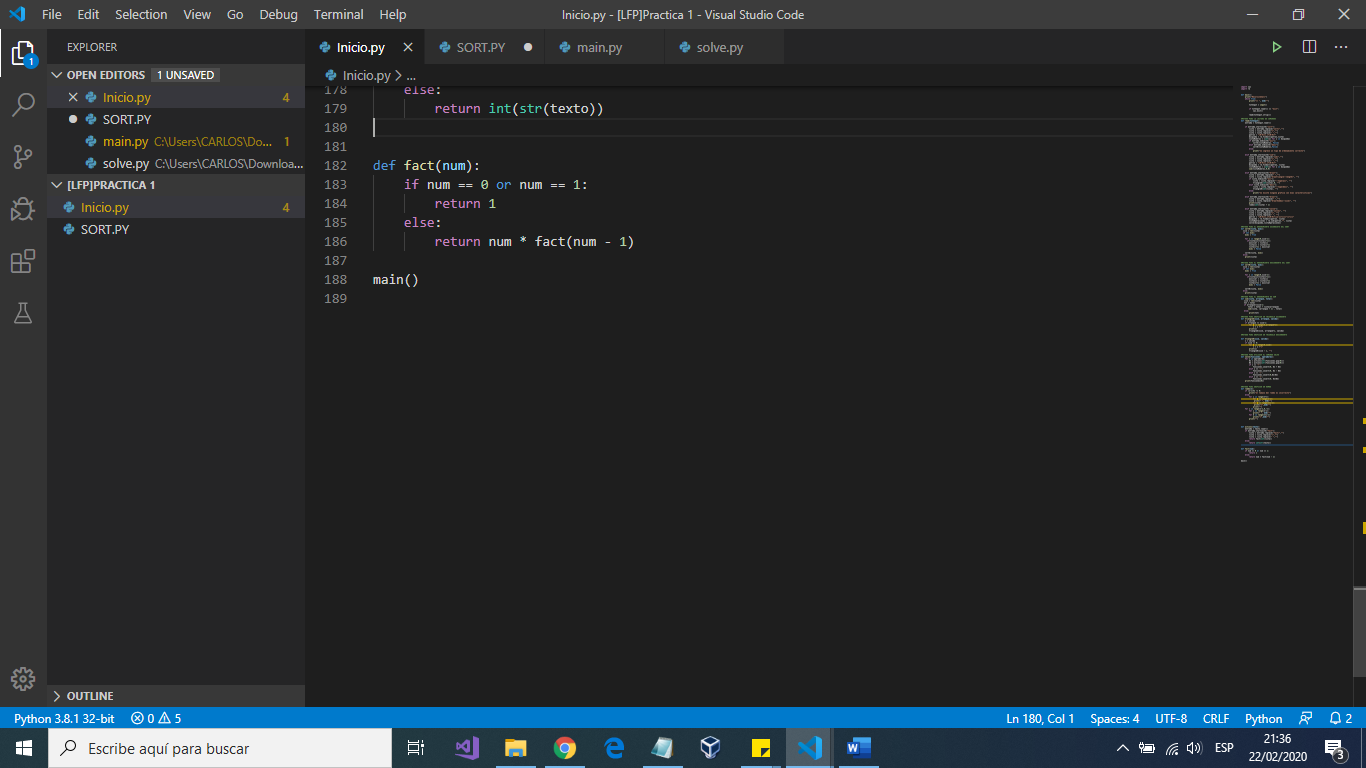
Este método sirve para hacer la operación Fibonacci, la cual utiliza solo un parámetro, esta da como resultado la suma de los dos números anterior al parámetro, y para el valor de 1 y cero se obtiene 1 por lo tanto en el método se pone un if donde si el parámetro es 1 o 0 se ponga como resultado 0, y para los demás se utiliza la recursividad sumando la función menos uno , más la función menos dos.

**ACKERMAN:**



Este método sirve para hacer la operación ackerman, donde si el numero uno es 0 el resultado será el numero 2 más uno que es lo que se presenta en el primer if, si el segundo numero es cero se utiliza la recursividad para restarle uno al primer número y se utiliza el numero uno en vez del cero en el segundo para determinar la salida, si ninguno de los dos es cero se utiliza recursividad y se le dan como parámetros el primer numero menos uno, y como el segundo se vuelve a utilizar recursividad con el primer número igual y el segundo menos uno.

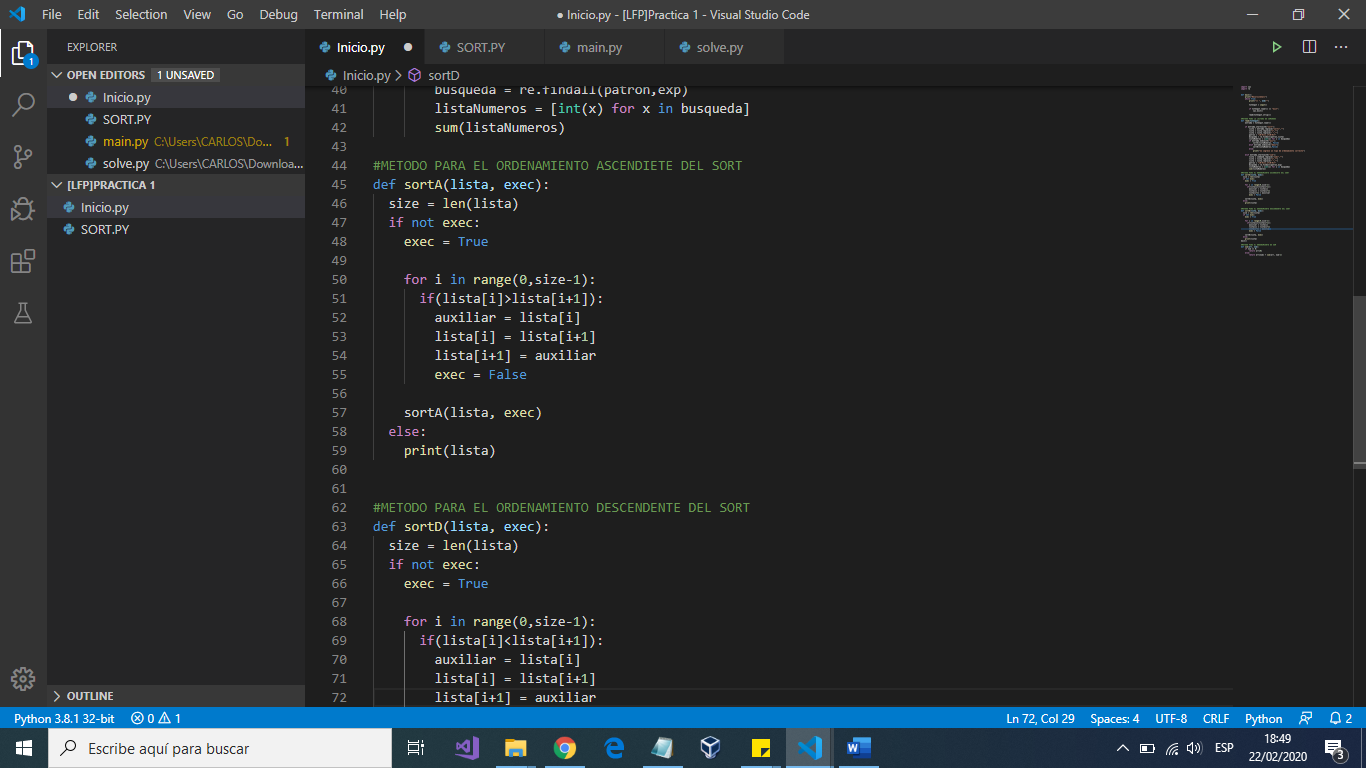
**FACTORIAL:**



Este método sirve para hacer la operación factorial , en el cual si el numero ingresado es 0 o 1 se retornara el valor de 1 pues es ese el resultado de ser mayor que uno se multiplica ese número por la función en si mima con el numero menos uno se detendrá cuando el número sea 1

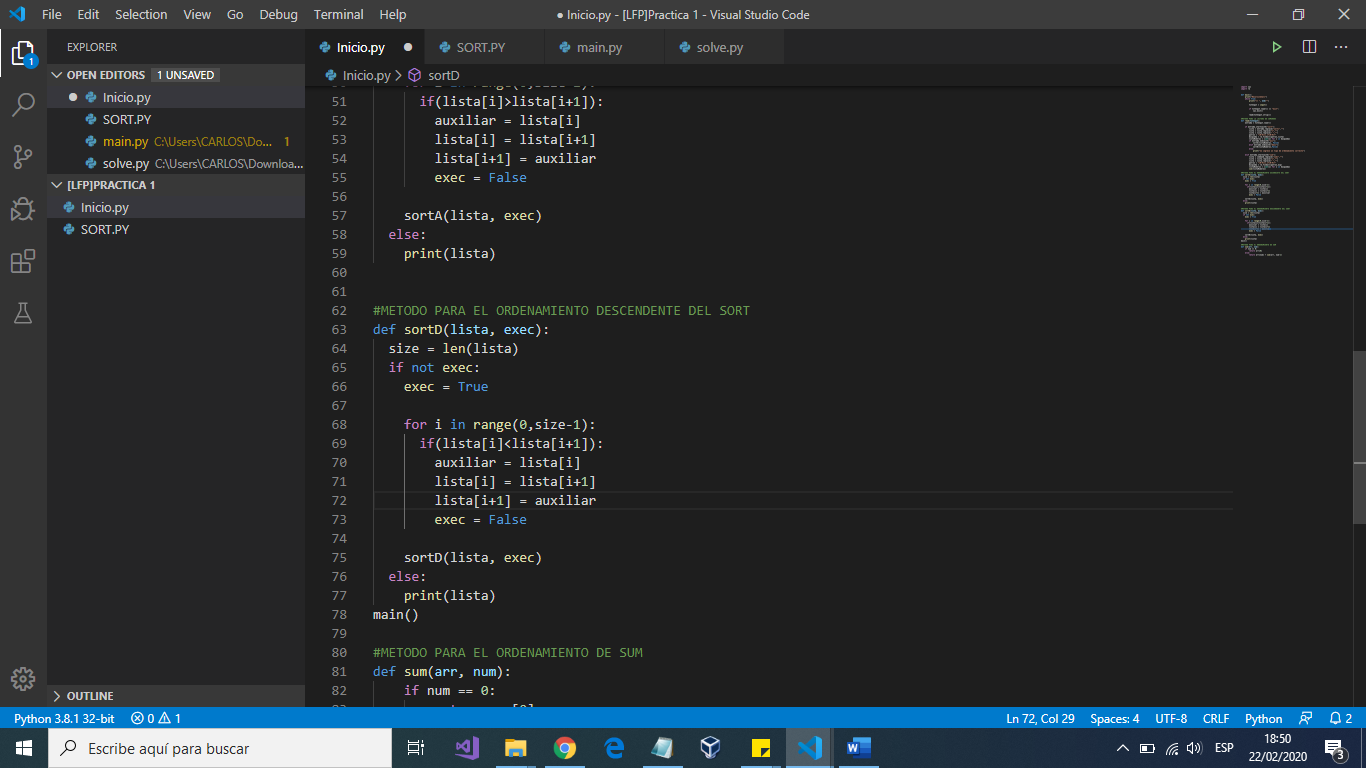
**MÉTODOS DE ORDENAMIENTO:**

**SORT ASC:**



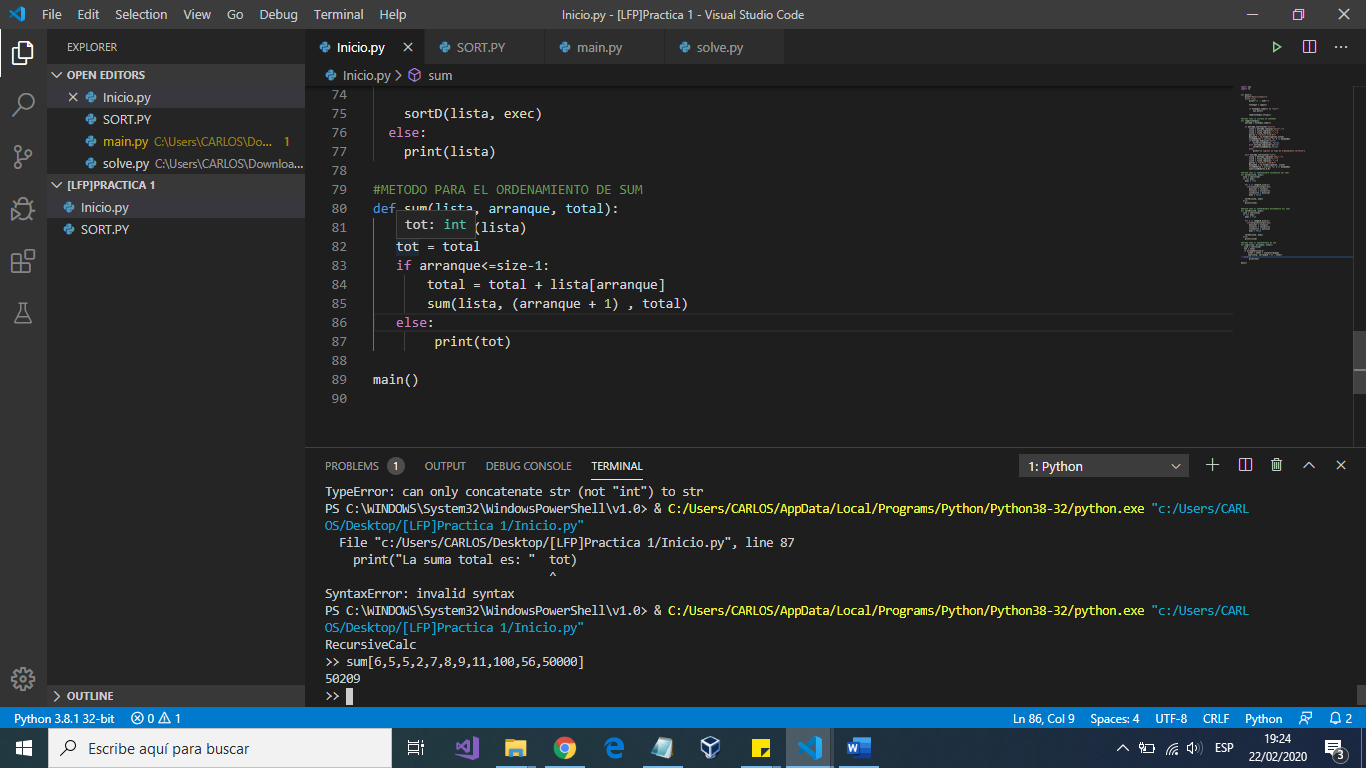
Este método sirve para ordenar una lista de numero de forma descendente, en donde va comparando desde el primer valor hasta el ultimo que estos sean uno menor que el otro de no ser así los cambia de lugar para ordenarlos de forma descendente.

**SORT DESC:**



Este método sirve para ordenar una lista de numero de forma ascendente, en donde va comparando desde el primer valor hasta el último que estos sean uno mayor que el otro de no ser así los cambia de lugar para ordenarlos de forma ascendente. En teoría es el mismo que se utilizo para el sort descendente únicamente cambiando el signo en el condicional del if utilizado.

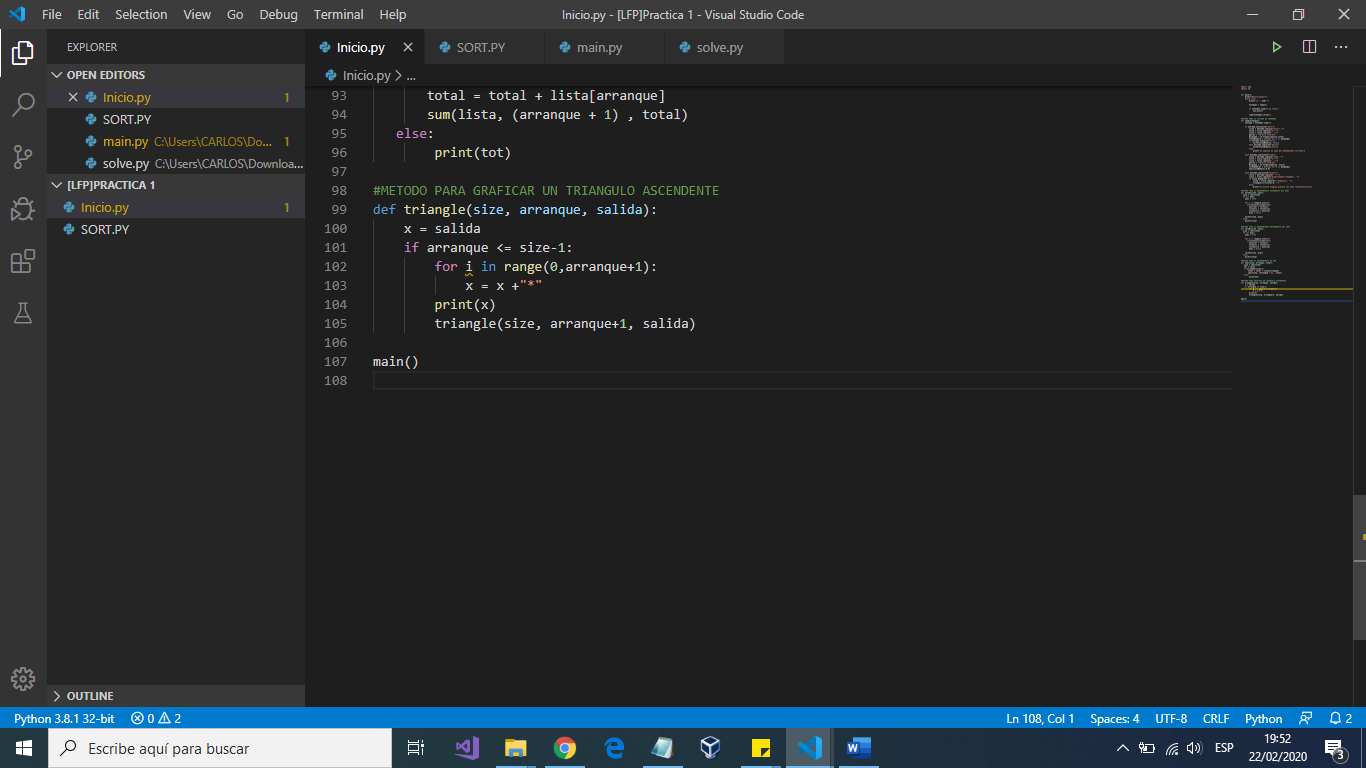
**SUM:**



Este método sirve para leer una lista de números dentro del comando de este y luego sumar uno por uno para luego dar como salida la suma de todos estos, utilizando recursividad tomamos una variable que sea el tamaño de la lista, un total en el que guardar el total obtenido suma por suma, se condiciona que la iteración de suma no sea mayo al numero de datos de la lista y se suma posición por posición hasta llegar a la ultima e imprimir el resultado.

**MÉTODOS GRÁFICOS:**

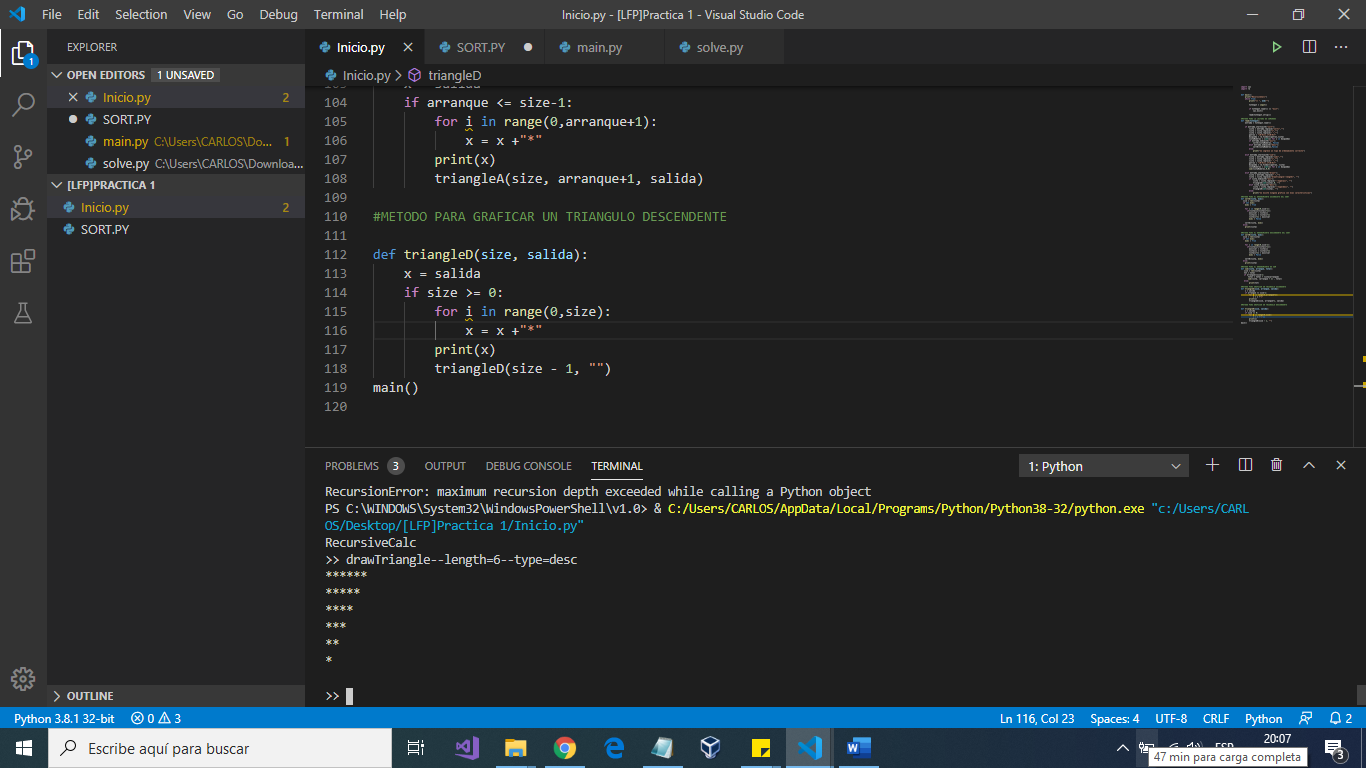
**DRAWTRIANGLE--ASC:**



Este método fue utilizado para graficar un triángulo rectángulo dependiendo del número de niveles que se desee esta ira incrementando un asterisco y línea, tiene como parámetro el tamaño del triángulo ósea el número de líneas, el arranque para darle el numero de asteriscos a empezar este caso 0 y salida que guarda el tamaño de asterisco por línea.

El for se utiliza para concatenar los asteriscos y el método recursivo para las líneas.

**DRAWTRIANGLE--DESC:**

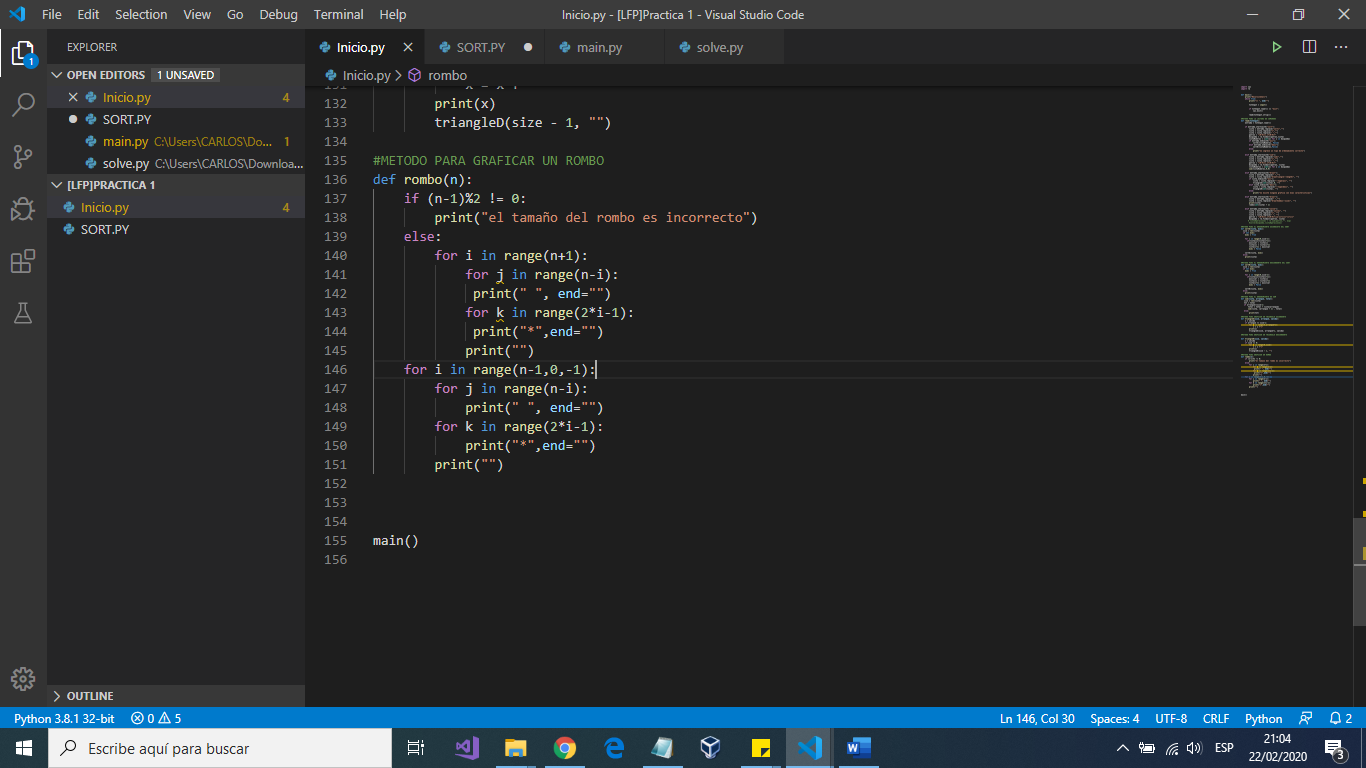


Este método fue utilizado para graficar un triángulo rectángulo dependiendo del número de niveles que se desee esta ira disminuyendo un asterisco y aumentando una línea, tiene como parámetro el tamaño del triángulo ósea el número de líneas, salida que guarda el tamaño de asterisco por línea.

El for se utiliza para concatenar los asteriscos y el método recursivo para las líneas.

En cada vez que se pasa por el método se le resta un uno al tamaño para tener el numero correctos de asterisco y terminar la condición

**DRAWRHOMBLE:**



Este método fue utilizado para graficar un rombo, en el cual se ingresaba un numero el cual seria la altura y longitud de este, se creo en haciendo la primer mitad de arriba con 3 for que recorren espacios en blanco según el tamaño de este y luego creo 2\*n – 1 asteriscos desde el centro del rombo, luego la diagonal y la parte de abajo juntas haciendo de manera inversa empezando por la mayor cantidad de puntos hasta llegar a uno contando los espacios vacíos a ir dejando