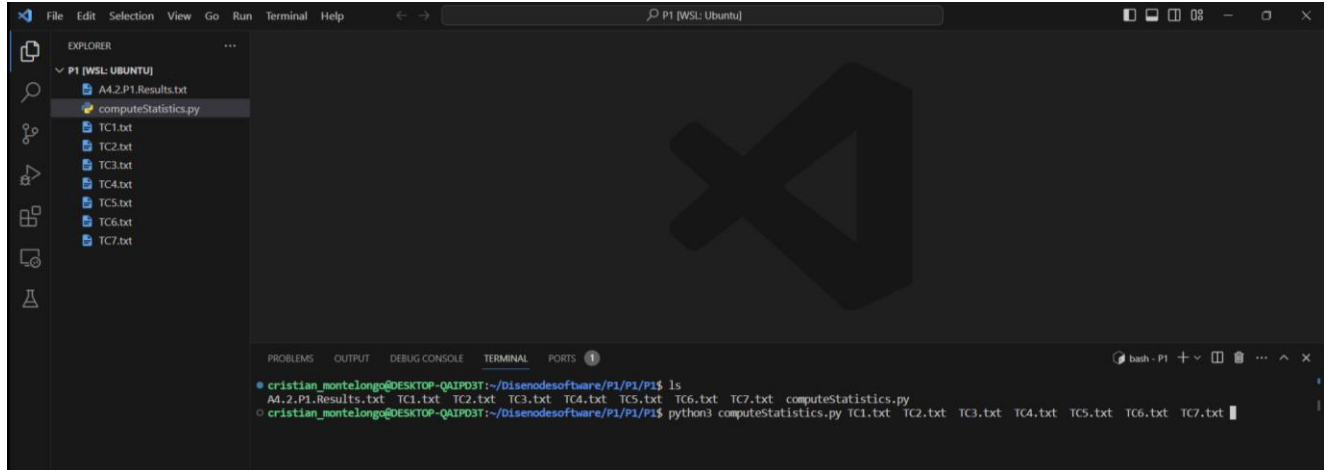


Actividad 4.2. Ejercicio de programación

Compute Statistics

Utilizando Visual Studio Code, desde la línea de comandos se manda a llamar el programa `computeStatistics.py` y los archivos(argumentos) que fueron proporcionados como recursos los cuales están ubicados en la misma dirección de mi programa.

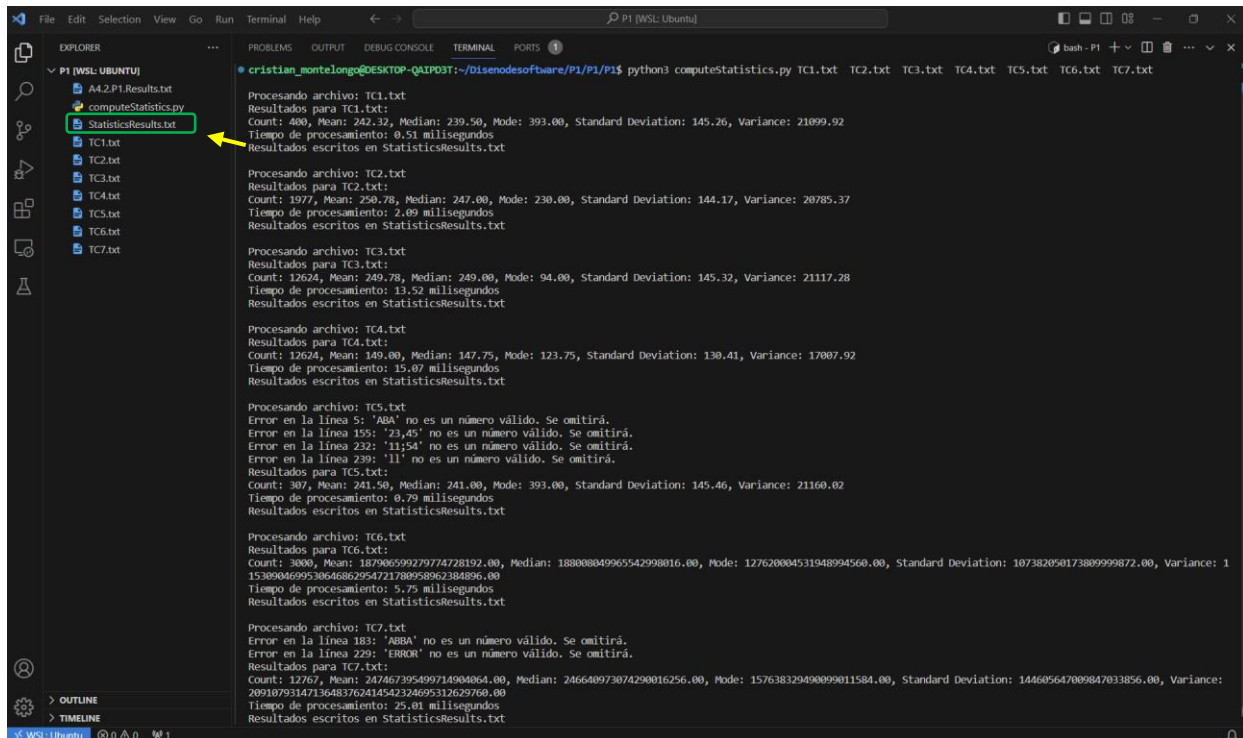


```
File Edit Selection View Go Run Terminal Help
P1 [WSL: Ubuntu]

EXPLORER
P1 [WSL: UBUNTU]
  A4.2.P1.Results.txt
  computeStatistics.py
  TC1.txt
  TC2.txt
  TC3.txt
  TC4.txt
  TC5.txt
  TC6.txt
  TC7.txt

TERMINAL
bash - P1
cristian_montelongo@DESKTOP-QAIPD3T:~/Disenodesoftware/P1/P1$ ls
A4.2.P1.Results.txt TC1.txt TC2.txt TC3.txt TC4.txt TC5.txt TC6.txt TC7.txt computeStatistics.py
cristian_montelongo@DESKTOP-QAIPD3T:~/Disenodesoftware/P1/P1$ python3 computeStatistics.py TC1.txt TC2.txt TC3.txt TC4.txt TC5.txt TC6.txt TC7.txt
```

Una vez que el programa se ejecuta tomando los archivos para alimentarlo, se obtendrán las variables que cuentan la cantidad de numero dentro del archivo, así como la media, mediana, moda, desviación estándar y varianza los cuales se imprimen en la pantalla, así como se genera un nuevo archivo llamado `StatisticsResults.txt` como se muestra en la imagen. También se está agregando el tiempo de procesamiento de cada archivo y el manejo de errores los cuales se muestran en la pantalla.



```
File Edit Selection View Go Run Terminal Help
P1 [WSL: Ubuntu]

EXPLORER
P1 [WSL: UBUNTU]
  A4.2.P1.Results.txt
  computeStatistics.py
  StatisticsResults.txt
  TC1.txt
  TC2.txt
  TC3.txt
  TC4.txt
  TC5.txt
  TC6.txt
  TC7.txt

TERMINAL
bash - P1
cristian_montelongo@DESKTOP-QAIPD3T:~/Disenodesoftware/P1/P1$ python3 computeStatistics.py TC1.txt TC2.txt TC3.txt TC4.txt TC5.txt TC6.txt TC7.txt

Procesando archivo: TC1.txt
Resultados para TC1.txt:
Count: 400, Mean: 242.32, Median: 239.50, Mode: 393.00, Standard Deviation: 145.26, Variance: 21099.92
Tiempo de procesamiento: 0.51 milisegundos
Resultados escritos en StatisticsResults.txt

Procesando archivo: TC2.txt
Resultados para TC2.txt:
Count: 1977, Mean: 250.78, Median: 247.00, Mode: 230.00, Standard Deviation: 144.17, Variance: 20785.37
Tiempo de procesamiento: 2.09 milisegundos
Resultados escritos en StatisticsResults.txt

Procesando archivo: TC3.txt
Resultados para TC3.txt:
Count: 12624, Mean: 249.78, Median: 249.00, Mode: 94.00, Standard Deviation: 145.32, Variance: 21117.28
Tiempo de procesamiento: 13.52 milisegundos
Resultados escritos en StatisticsResults.txt

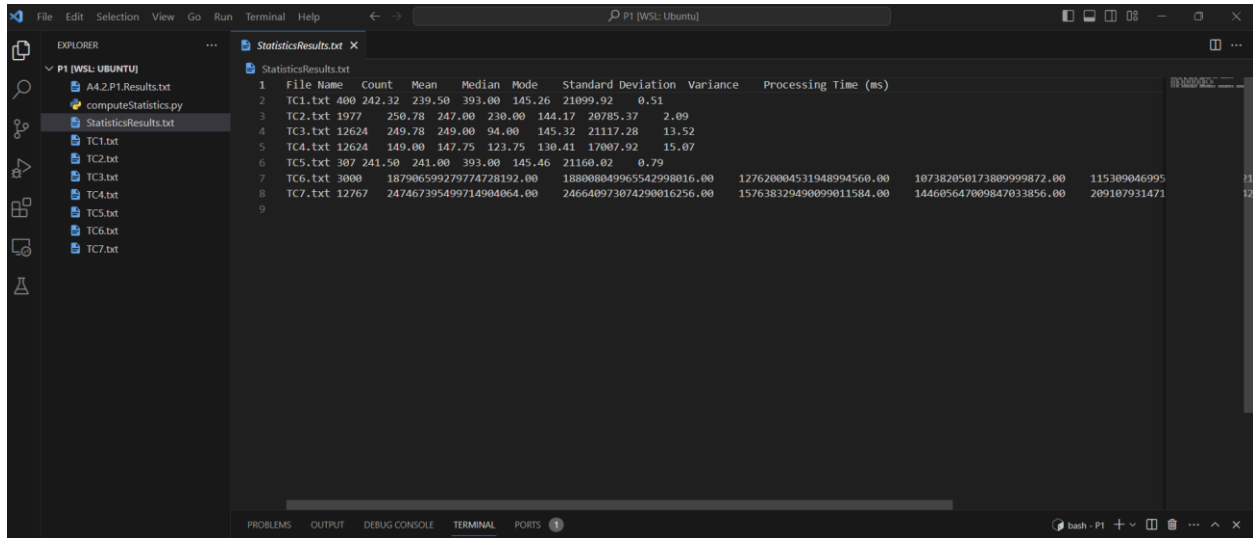
Procesando archivo: TC4.txt
Resultados para TC4.txt:
Count: 12624, Mean: 149.00, Median: 147.75, Mode: 123.75, Standard Deviation: 130.41, Variance: 17007.92
Tiempo de procesamiento: 15.07 milisegundos
Resultados escritos en StatisticsResults.txt

Procesando archivo: TC5.txt
Error en la línea 5: 'ABA' no es un número válido. Se omitirá.
Error en la línea 155: '23,45' no es un número válido. Se omitirá.
Error en la línea 232: '11:54' no es un número válido. Se omitirá.
Error en la línea 239: '11' no es un número válido. Se omitirá.
Resultados para TC5.txt:
Count: 307, Mean: 241.50, Median: 241.00, Mode: 393.00, Standard Deviation: 145.46, Variance: 21160.02
Tiempo de procesamiento: 0.79 milisegundos
Resultados escritos en StatisticsResults.txt

Procesando archivo: TC6.txt
Resultados para TC6.txt:
Count: 3000, Mean: 187906599279774728192.00, Median: 188008049965542998016.00, Mode: 127620004531948994560.00, Standard Deviation: 107382050173809999872.00, Variance: 1
153090469953064686295472178095896238486.00
Tiempo de procesamiento: 5.75 milisegundos
Resultados escritos en StatisticsResults.txt

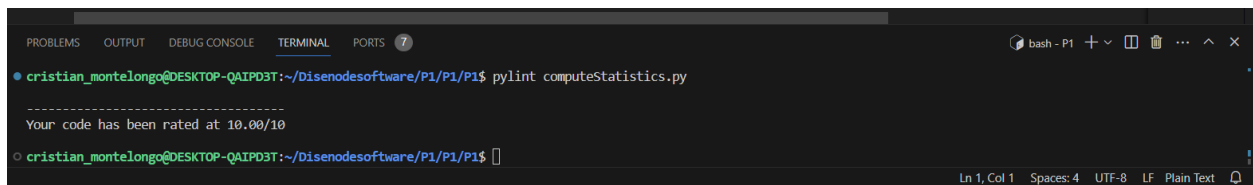
Procesando archivo: TC7.txt
Error en la línea 183: 'ABBA' no es un número válido. Se omitirá.
Error en la línea 229: 'ERROR' no es un número válido. Se omitirá.
Resultados para TC7.txt:
Count: 12767, Mean: 247467395499714904064.00, Median: 246640973074290016256.00, Mode: 15763832949009011584.00, Standard Deviation: 144605647009847033856.00, Variance: 1
20910793147136483762414542324695312629760.00
Tiempo de procesamiento: 25.01 milisegundos
Resultados escritos en StatisticsResults.txt
```

En esta imagen se observa la información que se escribió en el nuevo archivo StatisticsResults.txt lo cual contiene los resultados de las estadísticas mencionadas.



```
1 File Name Count Mean Median Mode Standard Deviation Variance Processing Time (ms)
2 TC1.txt 400 242.32 239.50 393.00 145.26 21099.92 0.51
3 TC2.txt 1977 250.78 247.00 230.00 144.17 20785.37 2.09
4 TC3.txt 12624 249.78 249.00 94.00 145.32 21117.28 13.52
5 TC4.txt 12624 149.00 147.75 121.75 130.41 17007.92 15.07
6 TC5.txt 307 241.50 241.00 393.00 145.46 21160.02 0.79
7 TC6.txt 3000 187906599279774728192.00 188008040965542998016.00 127620004531948994560.00 107382050173809999872.00 115309046995 11
8 TC7.txt 12767 247467395499714904064.00 246640973074290016256.00 157638329490099011584.00 144605647009847033856.00 209107931471 12
9
```

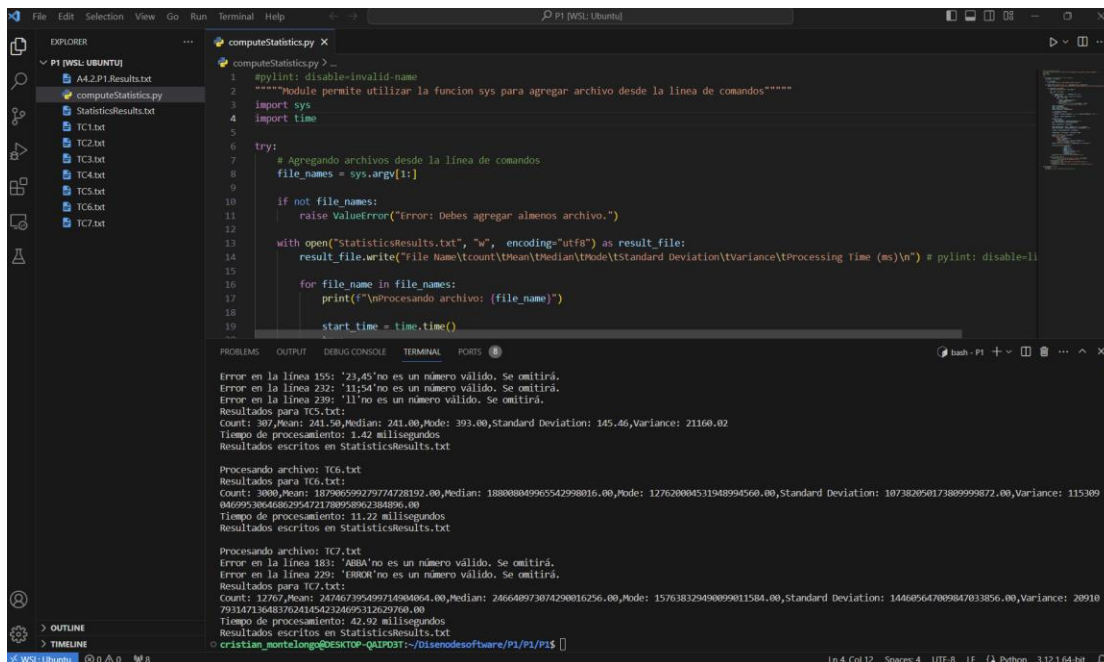
Se encontraron algunos detalles en el código como el código de “invalid-name” que se debe al, y con ayuda de pylint se corrigieron y se validó que todo estuviera funcionando correctamente y sin errores.



```
crístian_montelongo@DESKTOP-QAIPD3T:~/Disenodesoftware/P1/P1$ pylint computeStatistics.py
-----
Your code has been rated at 10.00/10

crístian_montelongo@DESKTOP-QAIPD3T:~/Disenodesoftware/P1/P1$
```

Se vuelve a correr el código para revisar la funcionalidad correcta.



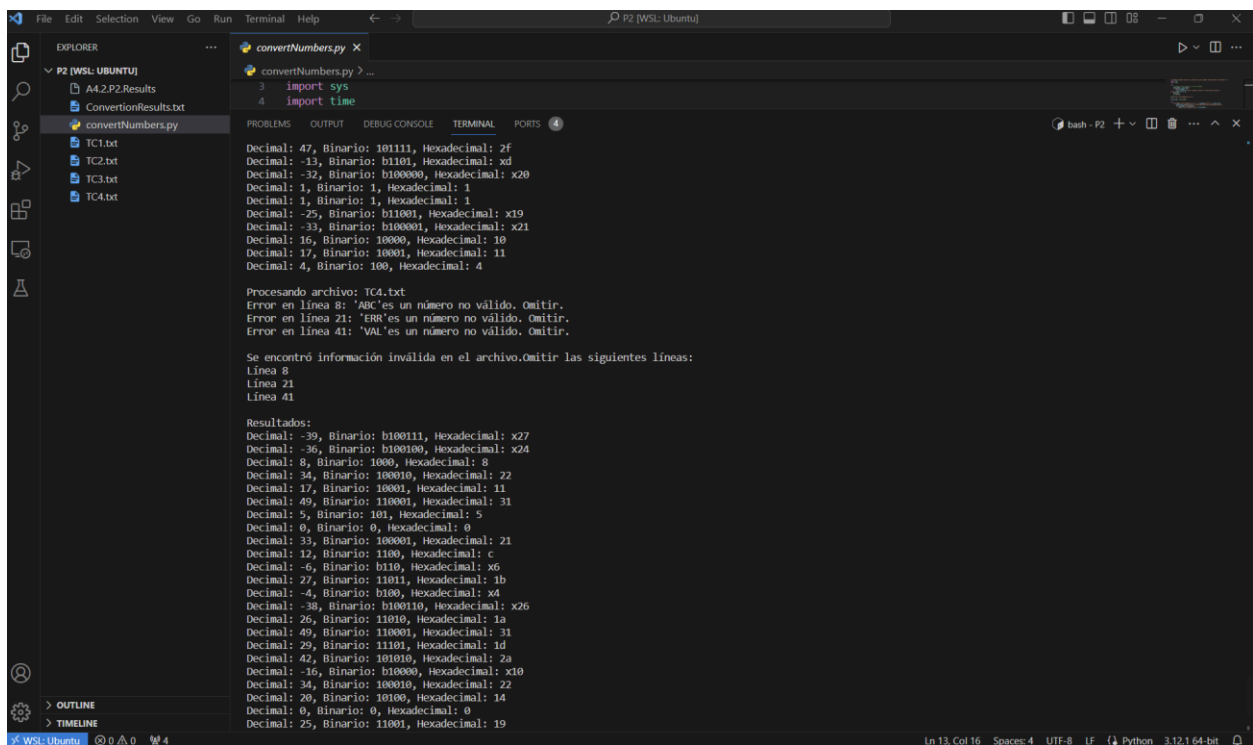
```
1 #pylint: disable=invalid-name
2 """Module permite utilizar la función sys para agregar archivo desde la línea de comandos"""
3 import sys
4 import time
5
6 try:
7     # Agregando archivos desde la línea de comandos
8     file_names = sys.argv[1:]
9
10    if not file_names:
11        raise ValueError("Error: Debes agregar al menos un archivo.")
12
13    with open("StatisticsResults.txt", "w", encoding="utf-8") as result_file:
14        result_file.write("File Name\tCount\tMean\tMedian\tMode\tStandard Deviation\tVariance\tProcessing Time (ms)\n")
15
16        for file_name in file_names:
17            print(f"Procesando archivo: {file_name}")
18
19            start_time = time.time()
20
21            # ... (rest of the code) ...
22
23            end_time = time.time()
24            processing_time = end_time - start_time
25
26            result_file.write(f"{file_name}\t{count}\t{mean}\t{median}\t{mode}\t{std_dev}\t{variance}\t{processing_time}\n")
27
28    result_file.close()
29
30    print("Resultados escritos en StatisticsResults.txt")
31
32    # Resultados para TC5.txt:
33    Count: 307, Mean: 241.50, Median: 241.00, Mode: 393.00, Standard Deviation: 145.46, Variance: 21160.02
34    Tiempo de procesamiento: 1.42 milisegundos
35
36    # Resultados para TC6.txt:
37    Count: 3000, Mean: 187906599279774728192.00, Median: 188008040965542998016.00, Mode: 127620004531948994560.00, Standard Deviation: 107382050173809999872.00, Variance: 115309046995
38    Tiempo de procesamiento: 11.22 milisegundos
39
40    # Resultados para TC7.txt:
41    Count: 12767, Mean: 247467395499714904064.00, Median: 246640973074290016256.00, Mode: 157638329490099011584.00, Standard Deviation: 144605647009847033856.00, Variance: 209107931471
42    Tiempo de procesamiento: 42.92 milisegundos
43
44    Resultados escritos en StatisticsResults.txt
```

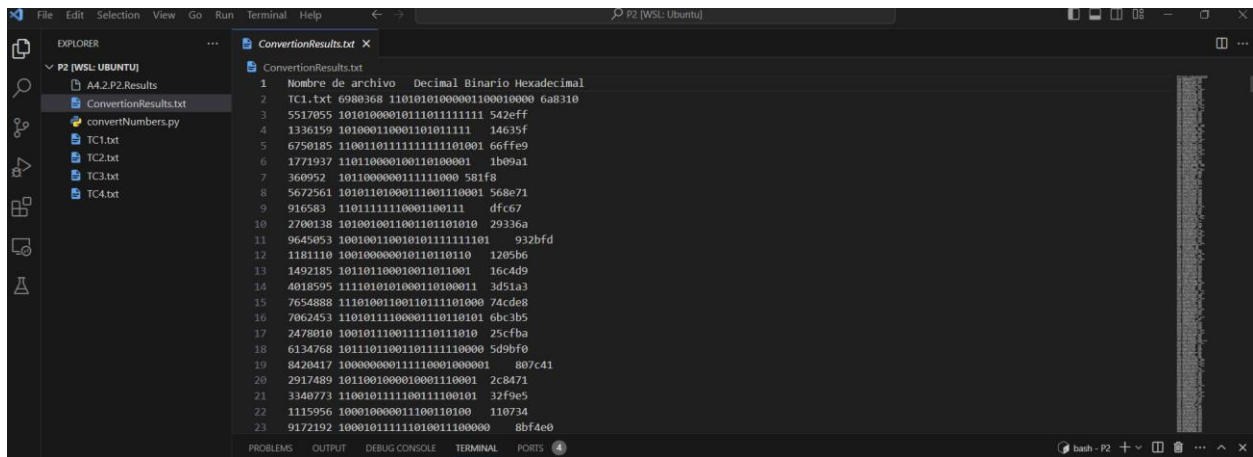
Converter

Volvemos a llamar desde la línea de comandos el segundo programa requerido el cual es Convert.Number.py



Al ejecutar se ejecuta correctamente el programa realizando lo que se pide en la actividad lo cual es convertir los números decimales a binario, hexadecimal y crear un archivo donde guardaran los datos con el nombre de ConversionResults.txt. (ver imagen).

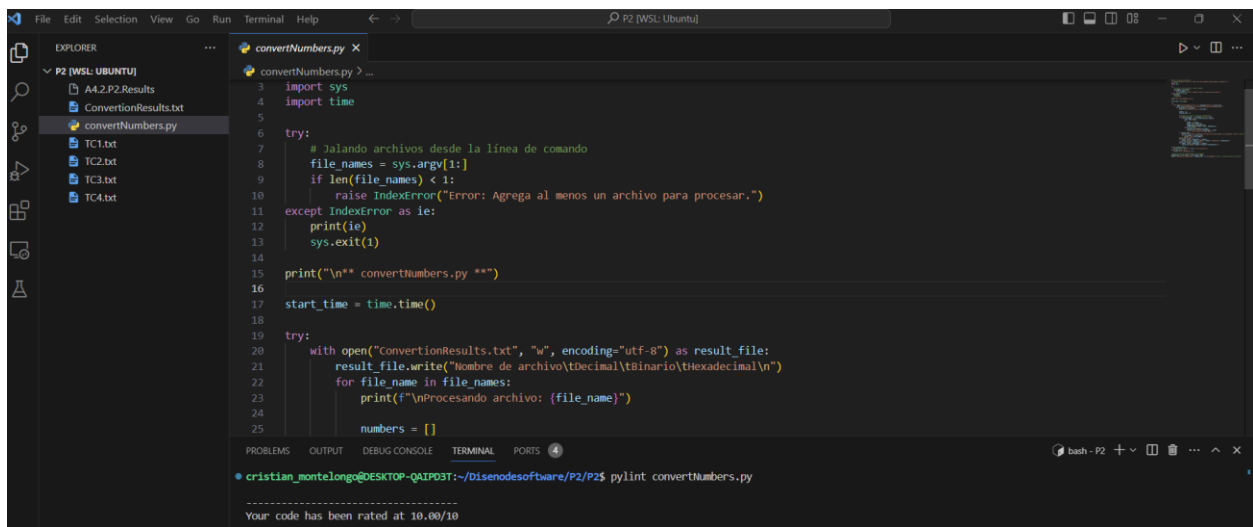




The screenshot shows a VS Code window with the Explorer sidebar on the left displaying a file tree for 'P2 [WSL: UBUNTU]'. The files include 'A4.2.P2.Results', 'ConversionResults.txt', 'convertNumbers.py', and four text files (TC1.txt to TC4.txt). The main editor displays 'ConversionResults.txt' with a table of conversion results. The table has four columns: 'Nombre de archivo', 'Decimal', 'Binario', and 'Hexadecimal'. It contains 23 rows of data, each representing a file from 'TC1.txt' to 'TC23.txt'.

Nombre de archivo	Decimal	Binario	Hexadecimal
TC1.txt	6980368	1101010100000100010000	6a8310
TC2.txt	5517055	1010100001011101111111	542eff
TC3.txt	1336159	1010001000101011111	14635f
TC4.txt	6750185	1100110111111111101001	66ffe9
TC5.txt	1771937	11011000010011010001	1b09a1
TC6.txt	360952	101100000111111000	581f8
TC7.txt	5672561	101010100011001110001	568e71
TC8.txt	916583	110111111000100111	dfc67
TC9.txt	2700138	101001001100110101010	29336a
TC10.txt	9645053	10010011001010111111101	932bfd
TC11.txt	1181110	1001000000101010110	1205b6
TC12.txt	1492185	10110110001001101001	16c4d9
TC13.txt	4018595	1111010101000110100011	3d51a3
TC14.txt	7654888	11101001100110111101000	74cd8
TC15.txt	7062453	11010111100001110110101	6bc3b5
TC16.txt	2478010	100101110011111011010	25cfba
TC17.txt	6134768	10111011001101111110000	5d9bf0
TC18.txt	8420417	100000000111110001000001	807c41
TC19.txt	2917489	101100100001000110001	2c8471
TC20.txt	3340773	110010111110011100101	32f9e5
TC21.txt	1115956	10001000001100110100	110734
TC22.txt	9172192	10001011111101001100000	8bf4e0
TC23.txt			

Se verifican los errores que arroja pylint y son corregidos, una vez realizado se valida con pylint que no cuenta con errores.



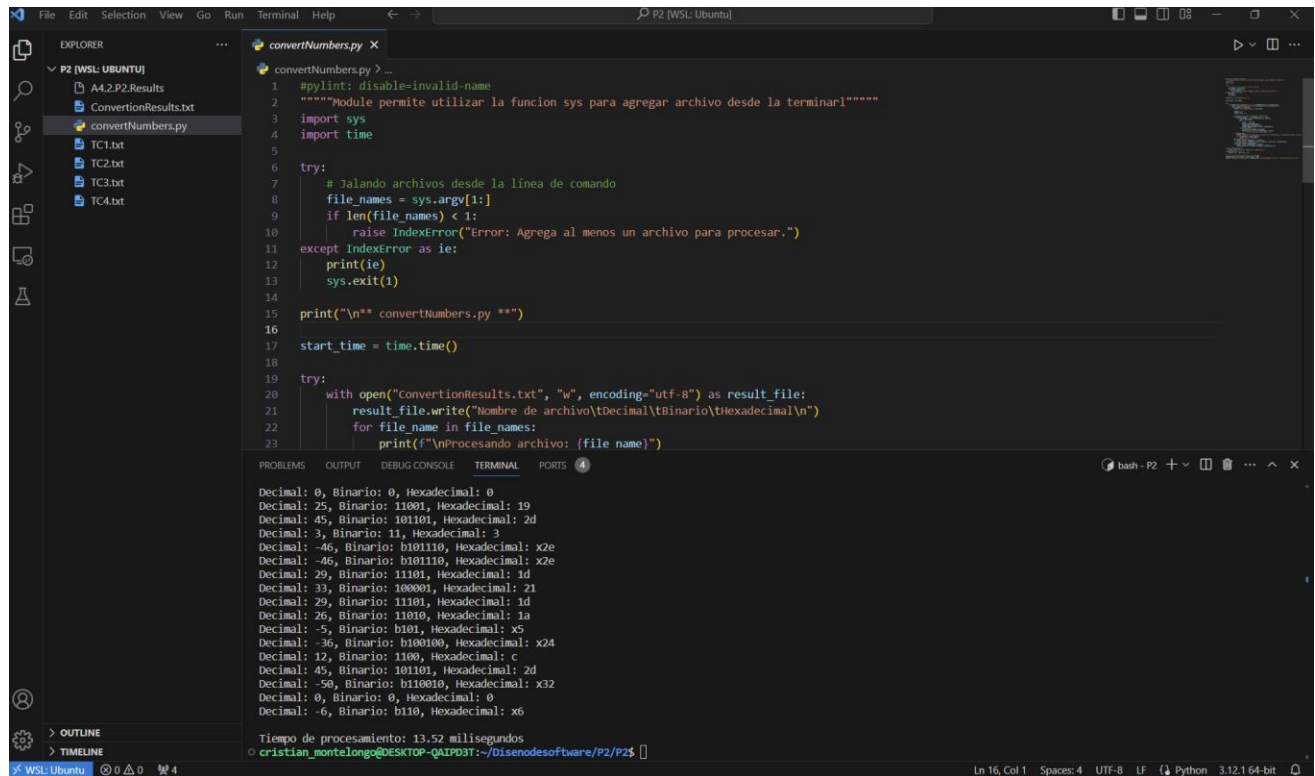
The screenshot shows the same VS Code window, but now the main editor displays the source code of 'convertNumbers.py'. The code is a Python script that takes command-line arguments, validates them, and writes conversion results to 'ConversionResults.txt'. The bottom terminal shows the command 'pylint convertNumbers.py' being executed, which returns a score of 10.00/10, indicating no errors were found.

```
3 import sys
4 import time
5
6 try:
7     # Jalando archivos desde la línea de comando
8     file_names = sys.argv[1:]
9     if len(file_names) < 1:
10         raise IndexError("Error: Agrega al menos un archivo para procesar.")
11 except IndexError as ie:
12     print(ie)
13     sys.exit(1)
14
15 print("\n** convertNumbers.py **")
16
17 start_time = time.time()
18
19 try:
20     with open("ConversionResults.txt", "w", encoding="utf-8") as result_file:
21         result_file.write("Nombre de archivo\tDecimal\tBinario\tHexadecimal\n")
22         for file_name in file_names:
23             print(f"Procesando archivo: {file_name}")
24             numbers = []
25
```

• cristian_montelongo@DESKTOP-QAIPD3T:~/Disenodesoftware/P2/P2\$ pylint convertNumbers.py

Your code has been rated at 10.00/10

Nuevamente se vuelve a correr el programa para revisar que todo esta funcionando correctamente .



The screenshot shows a Visual Studio Code editor window with a Python file named `convertNumbers.py` open. The file explorer on the left shows the project structure, including `convertNumbers.py` and several text files. The code in the editor is a Python script that takes command-line arguments and converts decimal numbers to binary and hexadecimal. The terminal at the bottom shows the output of the script, displaying a list of conversions for various decimal values.

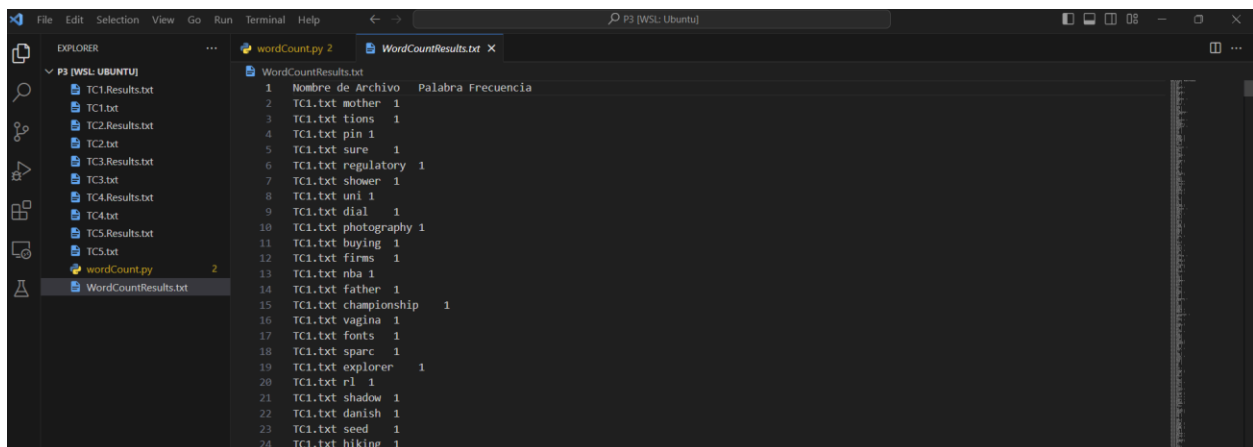
```
convertNumbers.py
1 #pylint: disable=invalid-name
2 """Module permite utilizar la funcion sys para agregar archivo desde la terminal"""
3 import sys
4 import time
5
6 try:
7     # Jalando archivos desde la línea de comando
8     file_names = sys.argv[1:]
9     if len(file_names) < 1:
10         raise IndexError("Error: Agrega al menos un archivo para procesar.")
11 except IndexError as ie:
12     print(ie)
13     sys.exit(1)
14
15 print("\n** convertNumbers.py **")
16
17 start_time = time.time()
18
19 try:
20     with open("ConversionResults.txt", "w", encoding="utf-8") as result_file:
21         result_file.write("Nombre de archivo\tDecimal\tBinario\tHexadecimal\n")
22         for file_name in file_names:
23             print(f"\nProcesando archivo: {file_name}")
24
25     # Output shown in the terminal
26     Decimal: 0, Binario: 0, Hexadecimal: 0
27     Decimal: 25, Binario: 11001, Hexadecimal: 19
28     Decimal: 45, Binario: 101101, Hexadecimal: 2d
29     Decimal: 3, Binario: 11, Hexadecimal: 3
30     Decimal: -46, Binario: b101110, Hexadecimal: x2e
31     Decimal: -46, Binario: b101110, Hexadecimal: x2e
32     Decimal: 29, Binario: 11101, Hexadecimal: 1d
33     Decimal: 33, Binario: 100001, Hexadecimal: 21
34     Decimal: 29, Binario: 11101, Hexadecimal: 1d
35     Decimal: 26, Binario: 11010, Hexadecimal: 1a
36     Decimal: -5, Binario: b101, Hexadecimal: x5
37     Decimal: -36, Binario: b100100, Hexadecimal: x24
38     Decimal: 12, Binario: 1100, Hexadecimal: c
39     Decimal: 45, Binario: 101101, Hexadecimal: 2d
40     Decimal: -50, Binario: b110010, Hexadecimal: x32
41     Decimal: 0, Binario: 0, Hexadecimal: 0
42     Decimal: -6, Binario: b110, Hexadecimal: x6
43
44     Tiempo de procesamiento: 13.52 milisegundos
45
46 cristian_montelongo@DESKTOP-QAIPD3T:~/Disenodesoftware/P2/P2$
```

Count Words

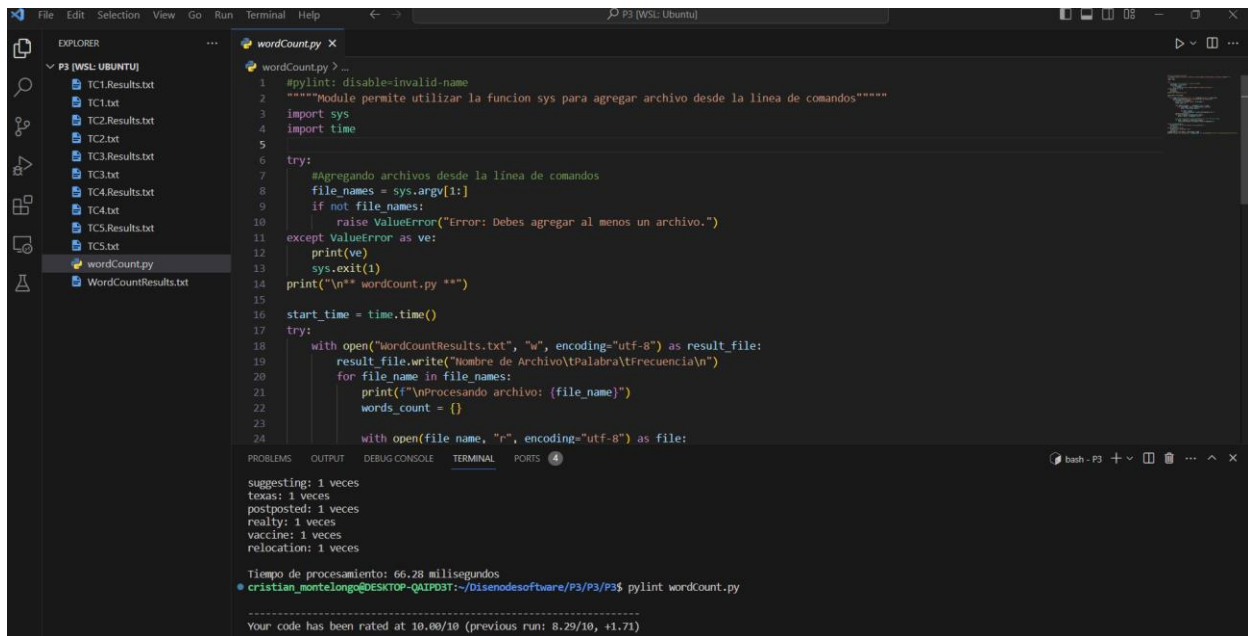
Volvemos a llamar desde la línea de comandos el tercer programa requerido el cual es wordCount.py



Al ejecutar se ejecuta correctamente el programa realizando lo que se pide en la actividad lo cual es contar la cantidad de veces que se repite una palabra en forma de frecuencia y crear un archivo donde guardaran los datos con el nombre de WordCountResults.txt. (ver imagen).



Se verifican los errores que arroja pylint y son corregidos, una vez realizado se valida con pylint que no cuenta con errores.



The screenshot shows the VS Code editor with the file explorer on the left displaying a project structure under 'P3 [WSL: UBUNTU]'. The files include TC1.Results.txt, TC1.txt, TC2.Results.txt, TC2.txt, TC3.Results.txt, TC3.txt, TC4.Results.txt, TC4.txt, TC5.Results.txt, TC5.txt, wordCount.py, and WordCountResults.txt. The main editor window shows the code for wordCount.py, which is a Python script designed to process multiple text files. The script includes comments in Spanish and uses sys.argv to handle command-line arguments. It features a try-except block for file processing and a final print statement for the execution time. The terminal at the bottom shows the output of running the script, listing word frequencies for various files and the total processing time of 66.28 milliseconds. A pylint message is also visible in the terminal output.

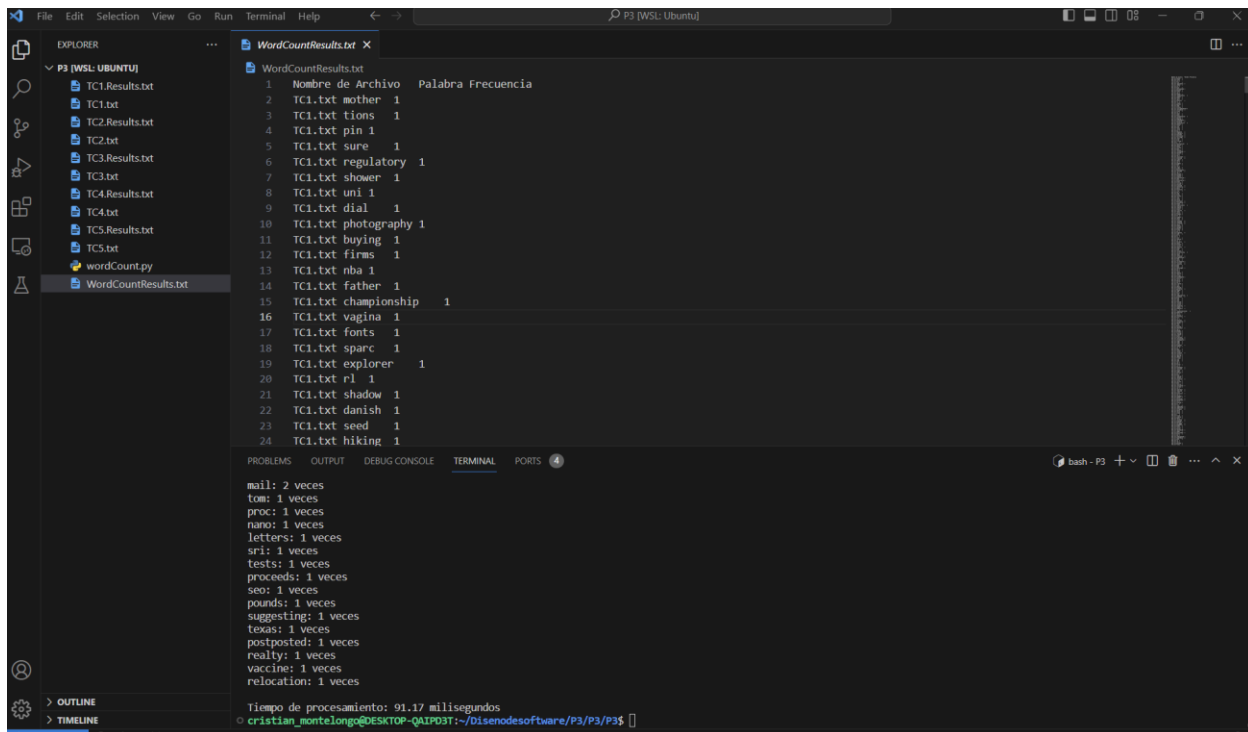
```
1 #pylint: disable=invalid-name
2 """Module permite utilizar la funcion sys para agregar archivo desde la linea de comandos"""
3 import sys
4 import time
5
6 try:
7     #Agregando archivos desde la linea de comandos
8     file_names = sys.argv[1:]
9     if not file_names:
10         raise ValueError("Error: Debes agregar al menos un archivo.")
11 except ValueError as ve:
12     print(ve)
13     sys.exit(1)
14 print("\n** wordCount.py **")
15
16 start_time = time.time()
17 try:
18     with open("WordCountResults.txt", "w", encoding="utf-8") as result_file:
19         result_file.write("Nombre de Archivo\tPalabra\tFrecuencia\n")
20         for file_name in file_names:
21             print(f"\nProcesando archivo: {file_name}")
22             words_count = {}
23
24             with open(file_name, "r", encoding="utf-8") as file:
```

suggesting: 1 veces
texas: 1 veces
postposted: 1 veces
really: 1 veces
vaccine: 1 veces
relocation: 1 veces

Tiempo de procesamiento: 66.28 milisegundos
cristian_montelongo@DESKTOP-QAIPD3T:~/Disenodesoftware/P3/P3\$ pylint wordCount.py

Your code has been rated at 10.00/10 (previous run: 8.29/10, +1.71)

Ahora se valida que una vez corregidos los errores con pylint se vuelve ejecutar el codigo para comprobar que esta funcionando correctamente.



The screenshot shows the VS Code editor with the file explorer on the left. The main editor window displays the contents of WordCountResults.txt, which is a tab-separated file listing the word frequency for each input file. The terminal at the bottom shows the output of running the script, listing word frequencies for various files and the total processing time of 91.17 milliseconds. A pylint message is also visible in the terminal output.

```
1 Nombre de Archivo Palabra Frecuencia
2 TC1.txt mother 1
3 TC1.txt tions 1
4 TC1.txt pin 1
5 TC1.txt sure 1
6 TC1.txt regulatory 1
7 TC1.txt shower 1
8 TC1.txt uni 1
9 TC1.txt dial 1
10 TC1.txt photography 1
11 TC1.txt buying 1
12 TC1.txt firms 1
13 TC1.txt nba 1
14 TC1.txt father 1
15 TC1.txt championship 1
16 TC1.txt vagina 1
17 TC1.txt fonts 1
18 TC1.txt sparc 1
19 TC1.txt explorer 1
20 TC1.txt rl 1
21 TC1.txt shadow 1
22 TC1.txt danish 1
23 TC1.txt seed 1
24 TC1.txt hiking 1
```

mail: 2 veces
tom: 1 veces
proc: 1 veces
nano: 1 veces
letters: 1 veces
sri: 1 veces
tests: 1 veces
proceeds: 1 veces
seo: 1 veces
pounds: 1 veces
suggesting: 1 veces
texas: 1 veces
postposted: 1 veces
really: 1 veces
vaccine: 1 veces
relocation: 1 veces

Tiempo de procesamiento: 91.17 milisegundos
cristian_montelongo@DESKTOP-QAIPD3T:~/Disenodesoftware/P3/P3\$