Tecnológico de Monterrey

Instituto Tecnológico y de Estudios Superiores de Monterrey

José Antonio Hernández Hernández

A01381334

Ejercicio de programación 1

Doctor Gerardo Padilla Zárate

Maestra María Mylen Treviño

Actividad 4.2

TE 4017

Domingo 2 de Febrero de 2025

Ejercicio de programación

Introducción

Esta documentación corresponde a la actividad 4.2, la cual incluye el desarrollo de tres programas en Python: computeStatistics.py, convertNumbers.py, y wordCount.py. Los programas fueron implementados siguiendo las prácticas de codificación, cumpliendo con el estándar PEP-8 y verificando su correcto funcionamiento mediante pruebas.

Desarrollo

1. computeStatistics.py

Funcionalidad: Calcula estadísticas descriptivas (media, mediana, moda, varianza y desviación estándar) a partir de un archivo de datos numéricos.

Requisitos cumplidos:

- Lectura de archivos con manejo de errores.
- Cálculo de estadísticas usando algoritmos básicos.
- Resultados mostrados en consola y guardados en StatisticsResults.txt.
- Medición del tiempo de ejecución.
- Cumplimiento con PEP-8.**Resultados de Pylint:**
- Cero errores de estilo.

Casos de Prueba: Archivo de entrada (data_numbers.txt):

10
20
30
40
50
60
70
80
90
100
abc
-10

Resultados esperados:

- Media: 50.00
- Mediana: 55.00
- Moda: 10.00
- Varianza: 916.67
- Desviación estándar: 30.28

Resultados de Pruebas:

- Todos los casos de prueba pasan correctamente.

2. convertNumbers.py

Funcionalidad: Convierte números de un archivo de entrada a su representación en binario y hexadecimal.

Requisitos cumplidos:

- Conversión implementada con algoritmos básicos.
- Manejo de errores para datos inválidos.
- Resultados mostrados en consola y guardados en ConvertionResults.txt.
- Medición del tiempo de ejecución.
- Cumplimiento con PEP-8.

Casos de Prueba: Archivo de entrada (data_numbers.txt):

```
5
15
255
1024
abc
-3
```

Resultados esperados:

- 5 -> Binary: 101, Hex: 5
- 15 -> Binary: 1111, Hex: F
- 255 -> Binary: 11111111, Hex: FF
- 1024 -> Binary: 10000000000, Hex: 400

Resultados de Pylint:

- Cero errores de estilo.

Resultados de Pruebas:

- Todos los casos de prueba pasan correctamente.

3. wordCount.py

Funcionalidad: Cuenta la frecuencia de palabras en un archivo de texto.

Requisitos cumplidos:

- Conteo de palabras implementado usando defaultdict.
- Manejo de errores para archivos inexistentes.
- Resultados mostrados en consola y guardados en WordCountResults.txt.
- Medición del tiempo de ejecución.
- Cumplimiento con PEP-8.

Casos de Prueba: Archivo de entrada (data_words.txt):

> Hello world
> hello Python
> Python is fun
> hello world hello

Resultados esperados:

- fun: 1
- hello: 4
- is: 1
- python: 2
- world: 2

Resultados de Pylint:

- Cero errores de estilo.

Resultados de Pruebas:

- Todos los casos de prueba pasan correctamente.

Liga de repositorio Github

https://github.com/pepehdezx2/A01381334_N-mero-de-actividadA4.2

Conclusión

Se cumplieron todos los requisitos de la actividad, asegurando un código limpio, funcional y conforme con los estándares de la industria. Los programas han sido verificados mediante pylint y pruebas de ejecución, obteniendo resultados satisfactorios.

Referencias

- Python Software Foundation. (2024). *PEP 8 – Style Guide for Python Code*. Recuperado de: https://peps.python.org/pep-0008/
- Python Software Foundation. (2024). *The Python Tutorial*. Recuperado de: https://docs.python.org/3/tutorial/index.html
- PyLint. (2024). *PyLint Documentation*. Recuperado de: https://pypi.org/project/pylint/
- Tutorial de PyLint (2024). *Video Tutorial*. Recuperado de: https://www.youtube.com/watch?v=fFY5103p5-c