ACTIVIDAD 4.2 EJERCICIOS DE PROGRAMACIÓN 1

PRUEBAS DE SOFTWARE Y ASEGURAMIENTO DE LA CALIDAD

Aarón Cortés García A01730451

Programa 1: Descriptive statistics from a file containing numbers.

EVIDENCIA FRAGMENTOS DEL CÓDIGO

computeStatistics.py

Programa1 > computeStatistics.py > main

```python
"""Program 1 of Activity 4.2 - Descriptive statistics from a file containing numbers.
   Aarón Cortés García - A01730451
"""

import sys
import time

def main():
    """Main function to execute the program."""

    if len(sys.argv) != 2:
        print("Usage: python computeStatistics.py <fileWithData.txt>")
        sys.exit(1)

    start_time = time.time()  # Start timing execution
    numbers = []

    with open(sys.argv[1], 'r', encoding='utf-8') as file:
        for line in file:
            try:
                numbers.append(float(line.strip()))  # Convert line to float and store it
            except ValueError:
```

computeStatistics.py

Programa1 > computeStatistics.py > main

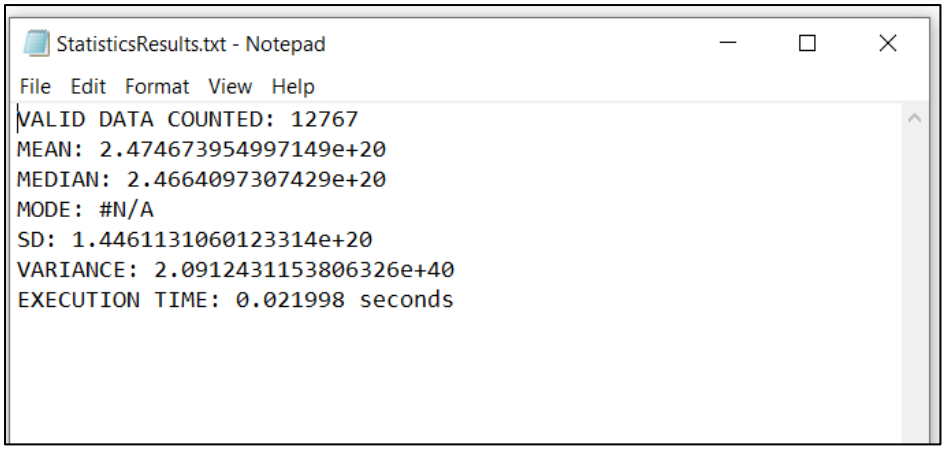
```python
def main():
    print(f"MODE: {mode}")
    print(f"SD: {std_dev}")
    print(f"VARIANCE: {variance}")
    print(f"EXECUTION TIME: {elapsed_time:.6f} seconds")

    # Save results to file
    with open("StatisticsResults.txt", "w", encoding="utf-8") as file:
        file.write(f"VALID DATA COUNTED: {count}")
        file.write(f"\nMEAN: {mean}")
        file.write(f"\nMEDIAN: {median}")
        file.write(f"\nMODE: {mode}")
        file.write(f"\nSD: {std_dev}")
        file.write(f"\nVARIANCE: {variance}")
        file.write(f"\nEXECUTION TIME: {elapsed_time:.6f} seconds\n")

if __name__ == "__main__":
    main()
```

EVIDENCIA DE EJECUCIÓN EN CONSOLA

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell - Programa1

```
(base) PS C:\Users\aaron\Documents\Maestría\Pruebas de software y aseguramiento de calidad\Act 4.2 Ejercicios Programación 1> cd Programa1
(base) PS C:\Users\aaron\Documents\Maestría\Pruebas de software y aseguramiento de calidad\Act 4.2 Ejercicios Programación 1\Programa1> python computeStatistics.py TC7.txt
Invalid data ignored: ABBA
Invalid data ignored: ERROR
VALID DATA COUNTED: 12767
MEAN: 2.474673954997149e+20
MEDIAN: 2.4664097307429e+20
MODE: #N/A
SD: 1.446113106012331 4e+20
VARIANCE: 2.0912431153806326e+40
EXECUTION TIME: 0.021998 seconds
(base) PS C:\Users\aaron\Documents\Maestría\Pruebas de software y aseguramiento de calidad\Act 4.2 Ejercicios Programación 1\Programa1>
```

EVIDENCIA DE ARCHIVO GENERADO



Programa 2: Convert decimal numbers to binary and hexadecimal base.

EVIDENCIA FRAGMENTOS DEL CÓDIGO

convertNumbers.py

Programa2 > convertNumbers.py > dec_to_bin

```python
"""Program 2 of Activity 4.2 - Convert decimal numbers to binary and hexadecimal base.
   Aarón Cortés García - A01730451
"""

import sys
import time

def dec_to_bin(number):
    """Converts a decimal number to binary.
       Negative numbers use 10-bit two's complement,
       positive numbers are regular binary.
    """
    if number < 0:
        # Two's complement for negative numbers (10 bits)
        number = (1 << 10) + number  # Compute two's complement for negative numbers
        binary = ""
        for _ in range(10):  # Ensure 10-bit representation
            binary = str(number % 2) + binary
            number //= 2
        return binary

    # Regular binary conversion for positive numbers
```

convertNumbers.py

Programa2 > convertNumbers.py > dec_to_bin

```python
def main():
                except ValueError:
                    print(f"Invalid data ignored: {line.strip()}")  # Notify about invalid data
                results.append((item_number, number, binary, hexadecimal))  # Store results
                item_number += 1  # Increment item count

    elapsed_time = time.time() - start_time  # Compute elapsed time

    # Print results to console
    for item, number, binary, hexadecimal in results:
        print(f"ITEM: {item}   DEC: {number}   BIN: {binary}   HEXA: {hexadecimal}")
    print(f"EXECUTION TIME: {elapsed_time:.6f} seconds")

    # Save results to file
    with open("ConvertionResults.txt", "w", encoding="utf-8") as file:
        for item, number, binary, hexadecimal in results:
            file.write(f"ITEM: {item}   DEC: {number}   BIN: {binary}   HEXA: {hexadecimal}\n")
        file.write(f"EXECUTION TIME: {elapsed_time:.6f} seconds\n")

if __name__ == "__main__":
    main()
```

EVIDENCIA DE EJECUCIÓN EN CONSOLA

```
(base) PS C:\Users\aaron\Documents\Maestría\Pruebas de software y aseguramiento de calidad\Act 4.2 Ejercicios Programación 1\Programa2>
(base) PS C:\Users\aaron\Documents\Maestría\Pruebas de software y aseguramiento de calidad\Act 4.2 Ejercicios Programación 1\Programa2> python convertNumbers.py TC4.txt
Invalid data ignored: ABC
Invalid data ignored: ERR
Invalid data ignored: VAL
ITEM: 1   DEC: -39   BIN: 1111011001   HEXA: FFFFFFFD9
ITEM: 2   DEC: -36   BIN: 1111011100   HEXA: FFFFFFFDC
ITEM: 3   DEC: 8   BIN: 1000   HEXA: 8
ITEM: 4   DEC: 34   BIN: 100010   HEXA: 22
ITEM: 5   DEC: 17   BIN: 10001   HEXA: 11
ITEM: 6   DEC: 49   BIN: 110001   HEXA: 31
ITEM: 7   DEC: 5   BIN: 101   HEXA: 5
```

EVIDENCIA DE ARCHIVO GENERADO

ConvertionResults.txt - Notepad

File Edit Format View Help

```
ITEM: 1   DEC: -39   BIN: 1111011001   HEXA: FFFFFFFD9
ITEM: 2   DEC: -36   BIN: 1111011100   HEXA: FFFFFFFDC
ITEM: 3   DEC: 8   BIN: 1000   HEXA: 8
ITEM: 4   DEC: 34   BIN: 100010   HEXA: 22
ITEM: 5   DEC: 17   BIN: 10001   HEXA: 11
ITEM: 6   DEC: 49   BIN: 110001   HEXA: 31
ITEM: 7   DEC: 5   BIN: 101   HEXA: 5
ITEM: 8   DEC: #N/A   BIN: #N/A   HEXA: #N/A
ITEM: 9   DEC: 0   BIN: 0   HEXA: 0
ITEM: 10   DEC: 33   BIN: 100001   HEXA: 21
ITEM: 11   DEC: 12   BIN: 1100   HEXA: C
ITEM: 12   DEC: -6   BIN: 1111111010   HEXA: FFFFFFFFFA
ITEM: 13   DEC: 27   BIN: 11011   HEXA: 1B
ITEM: 14   DEC: -4   BIN: 1111111100   HEXA: FFFFFFFFFC
ITEM: 15   DEC: -38   BIN: 1111011010   HEXA: FFFFFFFFDA
```

ConvertionResults.txt - Notepad

File Edit Format View Help

```
ITEM: 27   DEC: 3   BIN: 11   HEXA: 3
ITEM: 28   DEC: -46   BIN: 1111010010   HEXA: FFFFFFFFD2
ITEM: 29   DEC: -46   BIN: 1111010010   HEXA: FFFFFFFFD2
ITEM: 30   DEC: 29   BIN: 11101   HEXA: 1D
ITEM: 31   DEC: 33   BIN: 100001   HEXA: 21
ITEM: 32   DEC: 29   BIN: 11101   HEXA: 1D
ITEM: 33   DEC: 26   BIN: 11010   HEXA: 1A
ITEM: 34   DEC: -5   BIN: 1111111011   HEXA: FFFFFFFFFB
ITEM: 35   DEC: -36   BIN: 1111011100   HEXA: FFFFFFFDC
ITEM: 36   DEC: 12   BIN: 1100   HEXA: C
ITEM: 37   DEC: 45   BIN: 101101   HEXA: 2D
ITEM: 38   DEC: -50   BIN: 1111001110   HEXA: FFFFFFFFCE
ITEM: 39   DEC: 0   BIN: 0   HEXA: 0
ITEM: 40   DEC: -6   BIN: 1111111010   HEXA: FFFFFFFFFA
ITEM: 41   DEC: #N/A   BIN: #N/A   HEXA: #N/A
EXECUTION TIME: 0.000998 seconds
```

Programa 3: Identify all distinct words in a file and the frequency of them.

EVIDENCIA FRAGMENTOS DEL CÓDIGO

wordCount.py

```python
"""Program 3 of Activity 4.2 - Identify all distinct words in a file and the frequency of them.
   Aarón Cortés García - A01730451
"""

import sys
import time

def read_file(file_path):
    """Reads the content of a file and returns the words as a list."""
    words = []
    try:
        with open(file_path, 'r', encoding='utf-8') as file:
            for line in file:
                words.extend(process_line(line))  # Process each line and add words to the list
    except FileNotFoundError:
        print(f"Error: The file '{file_path}' was not found.")
        sys.exit(1)
    except (PermissionError, IsADirectoryError) as e:  # More specific exceptions
        print(f"Error reading file: {e}")
        sys.exit(1)
    return words
```

wordCount.py

```python
def write_results(word_count, word_order, execution_time):
    except (PermissionError, OSError) as e:  # More specific exceptions
        print(f"Error writing results: {e}")
        sys.exit(1)

def main():
    """Main function to execute the word count program."""
    if len(sys.argv) != 2:
        print("Usage: python wordCount.py fileWithData.txt")
        sys.exit(1)

    file_path = sys.argv[1]
    start_time = time.time()  # Start execution timer
    words = read_file(file_path)  # Read words from file
    word_count, word_order = count_word_frequency(words)  # Count word occurrences
    execution_time = time.time() - start_time  # Calculate execution time
    write_results(word_count, word_order, execution_time)  # Write and display results

if __name__ == "__main__":
    main()
```

EVIDENCIA DE EJECUCIÓN EN CONSOLA

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS        powershell - Programa3

(base) PS C:\Users\aaron\Documents\Maestría\Pruebas de software y aseguramiento de calidad\Act 4.2 Ejercicios Programación 1\Programa3> python wordCount.py TC3.txt
notice: 3
flood: 2
pottery: 2
charity: 2
suggestion: 2
pairs: 2
blues: 2
pipe: 2
thumb: 2
reveals: 2
copy: 2
hurt: 2
neighbors: 1
manual: 1
political: 1
mozambique: 1
old: 1
holding: 1
fc: 1
ford: 1
comparable: 1
```

EVIDENCIA DE ARCHIVO GENERADO