In [47]:
```
#MAESTRÍA EN INTELIGENCIA ARTIFICIAL APLICADA
#Pruebas de software y aseguramiento de la calidad
#Dr. Gerardo Padilla Zárate

#Actividad 4.2. Ejercicio de Programación 1
#CARLOS ENRIQUEZ GORGONIO
#A01793102
#20 de febrero de 2024
```

In [45]:
```
'''Ejercicio 3. Count Words
Detalles:
Req1. The program shall be invoked from a command line. The program shall receive a file as parameter. The file will contain a words (presumable between spaces).
Req 2. The program shall identify all distinct words and the frequency of them (how many times the word “X” appears in the file). The results shall be print on a screen and on a f
All computation MUST be calculated using the basic algorithms, not functions or libraries.
Req 3. The program shall include the mechanism to handle invalid data in the file. Errors should be displayed in the console and the execution must continue.
Req 4. The name of the program shall be wordCount.py
Req 5. The minimum format to invoke the program shall be as follows:python wordCount.py fileWithData.txt
Req 6. The program shall manage files having from hundreds of items to thousands of items.
Req 7. The program should include at the end of the execution the time elapsed for the execution and calculus of the data. This number shall be included in the results file and on
Req 8. Be compliant with PEP8.
'''
```

Out[45]:
```
'Ejercicio 1. Compute statistics\nDetalles:\nReq1. The program shall be invoked from a command line. The program shall receive a file as parameter. The file will contain a list of
items (presumable numbers).\nReq 2. The program shall compute all descriptive statistics from a file containing numbers. The results shall be print on a screen and on a file named
StatisticsResults.txt. All computation MUST \nbe calculated using the basic algorithms, not functions or libraries.The descriptive statistics are mean, median, mode, standard devi
ation, and variance.\nReq 3. The program shall include the mechanism to handle invalid data in the file. Errors should be displayed in the console and the execution must continue.
\nReq 4. The name of the program shall becomputeStatistics.py\nReq 5. The minimum format to invoke the program shall be as follows:python computeStatistics.py fileWithData.txt\nRe
q 6. The program shall manage files having from hundreds of items to thousands of items.\nReq 7. The program should include at the end of the execution the time elapsed for the ex
ecution and calculus of the data. This \nnumber shall be included in the results file and on the screen.\nReq 8. Be compliant with PEP8.\n'
```

In [14]:
```
!pip install pylint
!pip install pylint[spelling]
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pylint in c:\programdata\anaconda3\lib\site-packages (2.16.2)
Requirement already satisfied: platformdirs>=2.2.0 in c:\programdata\anaconda3\lib\site-packages (from pylint) (3.10.0)
Requirement already satisfied: astroid<=2.16.0-dev0,>=2.14.2 in c:\programdata\anaconda3\lib\site-packages (from pylint) (2.14.2)
Requirement already satisfied: isort<6,>=4.2.5 in c:\programdata\anaconda3\lib\site-packages (from pylint) (5.9.3)
Requirement already satisfied: mccabe<0.8,>=0.6 in c:\programdata\anaconda3\lib\site-packages (from pylint) (0.7.0)
Requirement already satisfied: tomlkit>=0.10.1 in c:\programdata\anaconda3\lib\site-packages (from pylint) (0.11.1)
Requirement already satisfied: dill>=0.3.6 in c:\programdata\anaconda3\lib\site-packages (from pylint) (0.3.6)
Requirement already satisfied: colorama>=0.4.5 in c:\programdata\anaconda3\lib\site-packages (from pylint) (0.4.6)
Requirement already satisfied: lazy-object-proxy>=1.4.0 in c:\programdata\anaconda3\lib\site-packages (from astroid<=2.16.0-dev0,>=2.14.2->pylint) (1.6.0)
Requirement already satisfied: wrapt<2,>=1.14 in c:\programdata\anaconda3\lib\site-packages (from astroid<=2.16.0-dev0,>=2.14.2->pylint) (1.14.1)
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pylint[spelling] in c:\programdata\anaconda3\lib\site-packages (2.16.2)
Requirement already satisfied: platformdirs>=2.2.0 in c:\programdata\anaconda3\lib\site-packages (from pylint[spelling]) (3.10.0)
Requirement already satisfied: astroid<=2.16.0-dev0,>=2.14.2 in c:\programdata\anaconda3\lib\site-packages (from pylint[spelling]) (2.14.2)
Requirement already satisfied: isort<6,>=4.2.5 in c:\programdata\anaconda3\lib\site-packages (from pylint[spelling]) (5.9.3)
Requirement already satisfied: mccabe<0.8,>=0.6 in c:\programdata\anaconda3\lib\site-packages (from pylint[spelling]) (0.7.0)
Requirement already satisfied: tomlkit>=0.10.1 in c:\programdata\anaconda3\lib\site-packages (from pylint[spelling]) (0.11.1)
Requirement already satisfied: dill>=0.3.6 in c:\programdata\anaconda3\lib\site-packages (from pylint[spelling]) (0.3.6)
Requirement already satisfied: colorama>=0.4.5 in c:\programdata\anaconda3\lib\site-packages (from pylint[spelling]) (0.4.6)
Requirement already satisfied: pyenchant~=3.2 in c:\users\traba\appdata\roaming\python\python311\site-packages (from pylint[spelling]) (3.2.2)
Requirement already satisfied: lazy-object-proxy>=1.4.0 in c:\programdata\anaconda3\lib\site-packages (from astroid<=2.16.0-dev0,>=2.14.2->pylint[spelling]) (1.6.0)
Requirement already satisfied: wrapt<2,>=1.14 in c:\programdata\anaconda3\lib\site-packages (from astroid<=2.16.0-dev0,>=2.14.2->pylint[spelling]) (1.14.1)
```

In [15]:
```
import sys
import time
```

In [16]:
```python
#Para definir la función de conteo de frecuencia de las palbras, separaremos los renglones con cadenas de texto, en palabras
def frecuencia(renglon):

    #Creamos un diccionario vacio
    frecuencia_individual = {}

    #Dividimos cada renglon en palabras, separadas por comas
    for cadena in renglon:
        palabras = cadena.split()

        for palabra in palabras:
            # Eliminamos los signos de puntuación, y homologamos las palabras en minuscula
            palabra = palabra.strip('.,!?').lower()

            # aumentamos nuestro indicador de frecuencias
            frecuencia_individual[palabra] = frecuencia_individual.get(palabra, 0) + 1

    return frecuencia_individual
```

In [20]:
```python
#Definimos nuestra función que mostrara resultados y generará el archivo
def impresora(ruta):

    try:
        #Iniciamos nuestro timer
        inicio = time.time()

        #Abrimos el archivo que se analizará
        with open(ruta, 'r', encoding="utf-8") as archivo:
            renglones = []
            for index, renglon in enumerate(archivo):
                try:
                    renglones.append(renglon.strip())
                except ValueError:
                    print(f"Error: El archivo tiene valores que no se pueden analizar {index+1}")

            integracion = "Row Labels   Count\n"
            contador = frecuencia(renglones)

            for indice, value in contador.items():
                integracion += f"{indice}: {value}\n"

            sumatoria = sum(contador.values())
            integracion += f"Total Global {sumatoria}"

            fin = time.time()
            tiempo_ejecucion = (fin - inicio) * 1000

            #imprimimos nuestros resultados, para visualizarlos
            print(integracion)
            print("\n")
            tiempo_total = f"Tiempo de ejecución es de: { tiempo_ejecucion:.6f} milisegundos"
            print(tiempo_total)

            #Creamos nuestro archivo con los elementos visualizados
            with open("WordCountResults.txt", "w", encoding="utf-8") as file:
                print(integracion, file=file)
                print("\n", file=file)
                print(tiempo_total, file=file)

    except FileNotFoundError:
        print(f"Error: No se encuentra el archivo '{file_path}'")
```

In [21]:
```python
#Para fines de observar resultados invocamos el archivo desde una ruta local, posteriormente queda la opción de invocarlo desde consola
impresora("C:/Users/traba/Downloads/TC1.txt")
#impresora("C:\Users\traba\Downloads\TC1.txt")
```

Row Labels Count
mother: 1
tions: 1
pin: 1
sure: 1
regulatory: 1
shower: 1
uni: 1
dial: 1
photography: 1
buying: 1
firms: 1
nba: 1
father: 1
championship: 1
vagina: 1
fonts: 1
sparc: 1
explorer: 1
rl: 1
shadow: 1
danish: 1
seed: 1
hiking: 1
instrumentation: 1
introduces: 1
kinda: 1
nor: 1
newer: 1
peter: 1
contamination: 1
matters: 1
bedding: 1
achievement: 1
password: 1
conservative: 2
webcast: 1
locks: 1
cove: 1
taxes: 1
could: 1
pct: 1
adequate: 1
nightmare: 1
marathon: 1
permission: 1
cartridge: 1
clear: 1
drum: 1
trained: 1
p: 1
manufacturer: 1
leisure: 1
media: 1
journey: 1
anal: 1
teaches: 1
customized: 1
oakland: 1
louis: 1
tab: 1
consistent: 1
enhanced: 1
liable: 1
ebony: 1
wan: 1

```
pubmed: 1
math: 1
tea: 1
craps: 1
gothic: 1
permissions: 1
recorded: 1
cgi: 1
confirm: 1
hyundai: 1
exhaust: 1
malpractice: 1
pens: 1
potentially: 1
glenn: 1
scoring: 1
andrews: 1
assessed: 1
adventures: 1
meals: 1
mortality: 1
club: 1
mon: 1
comm: 1
blues: 1
collect: 1
lies: 1
seats: 1
worse: 1
guestbook: 1
influences: 1
kodak: 1
significance: 1
coastal: 1
Total Global 100


Tiempo de ejecución es de: 0.000000 milisegundos
```

In [ ]:

```python
if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("Introduce la ruta como se muestra: python compute_statistics.py P1/TC2.txt")
        sys.exit(1)
    ruta = sys.argv[1]
    impresora(ruta)
```