

UD4. MapReduce y Spark

"Idiomas"

NOMBRE Y APELLIDOS: MARIO REY BULLIDO

DNI: 39459575Q

1.- Crea un archivo mapper.py que recorra el “Whatsapp_español.txt” y para cada letra guarde en “salida_mapper.txt” una pareja de clave-valor donde la clave es la letra y el valor sea 1 indicando que es una aparición. Nuestro mapper no debería distinguir entre mayúsculas y minúsculas ni letras acentuadas. Por ejemplo, “U”, “ü”, “ú”,... son la misma letra. Los números, espacios en blanco, signos de puntuación, comillas y similares no deben ser tenidos en cuenta, solamente queremos las letras del alfabeto.

Como respuesta a esta pregunta muestra tu código comentado.

```
# mapper.py
import unicodedata
import sys
import os

def normalizar_letra(letra): # Normaliza la letra a mayúscula y elimina acentos
    letra = letra.upper() # Convertir a mayúscula
    if letra == 'Ñ': # Verifica si la letra es Ñ
        return 'Ñ' # No modificar la Ñ
    letra = unicodedata.normalize('NFD', letra) # Descomponer caracteres acentuados
    letra = ''.join([c for c in letra if unicodedata.category(c) != 'Mn']) # Eliminar acentos
    return letra

def es_letra(c):
    return c.isalpha() # Verifica si el carácter es una letra

def main():
    if len(sys.argv) != 2: # Verifica que se haya pasado el nombre del archivo como argumento
        print("Uso: python mapper.py <archivo_entrada.txt>")
        sys.exit(1)

    archivo_entrada = sys.argv[1] # Nombre del archivo de entrada

    if not os.path.isfile(archivo_entrada): # Verifica si el archivo existe
        print(f"Error: El archivo '{archivo_entrada}' no existe.")
        sys.exit(1)

    with open(archivo_entrada, "r", encoding="utf-8") as entrada,
        open("salida_mapper.txt", "w", encoding="utf-8") as salida: # Abre el archivo de entrada y crea el archivo de salida
        for linea in entrada:
            for caracter in linea:
                if es_letra(caracter): # Verifica si el carácter es una letra
                    letra_normalizada = normalizar_letra(caracter) # Normaliza la letra
                    salida.write(f"{letra_normalizada}\t1\n") # Escribe la letra normalizada y el conteo (1) en el archivo de salida

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

Este script mapper.py recibe como argumento el nombre del fichero (en este caso Whatsapp_castellano.txt) abre el archivo y recorre las líneas y por cada línea recorre cada caracter. Verifica si el caracter es una letras y si lo es la normaliza pasándola a mayúsculas y eliminando los acentos.

Después la escribe en el archivo salida_mapper.txt seguido de un 1 para indicar el conteo.

pia > Mapreduce_Idiomas > salida_mapper.txt > data

| | | |
|----|---|---|
| 1 | F | 1 |
| 2 | E | 1 |
| 3 | C | 1 |
| 4 | H | 1 |
| 5 | A | 1 |
| 6 | D | 1 |
| 7 | E | 1 |
| 8 | E | 1 |
| 9 | N | 1 |
| 10 | T | 1 |
| 11 | R | 1 |
| 12 | A | 1 |
| 13 | D | 1 |
| 14 | A | 1 |
| 15 | E | 1 |
| 16 | N | 1 |
| 17 | V | 1 |
| 18 | I | 1 |
| 19 | G | 1 |
| 20 | E | 1 |
| 21 | N | 1 |
| 22 | C | 1 |

2.- Aplica el archivo Python “ordenar.py” al archivo “salida_mapper.txt” para ordenar los resultados en un nuevo archivo “entrada_reducer.txt”.

Muestra la salida del comando “head -n 20 entrada_reducer.txt” para ver las 20 primeras líneas del resultado.

```
# ordenar.py
import sys
import locale

def clave_ordenación(linea):
    letra = linea.split('\t')[0]
    # Reglas de ordenamiento: la Ñ debe ir justo después de la N
    # Truco: para comparar, cambiamos "Ñ" por "N~", así se ordena bien
    if letra == 'Ñ':
        return 'N~'
    return letra

def main():
    try:
        with open("salida_mapper.txt", "r", encoding="utf-8") as entrada:
            líneas = entrada.readlines()

        # Ordenamos usando la clave personalizada
        líneas.sort(key=clave_ordenación)

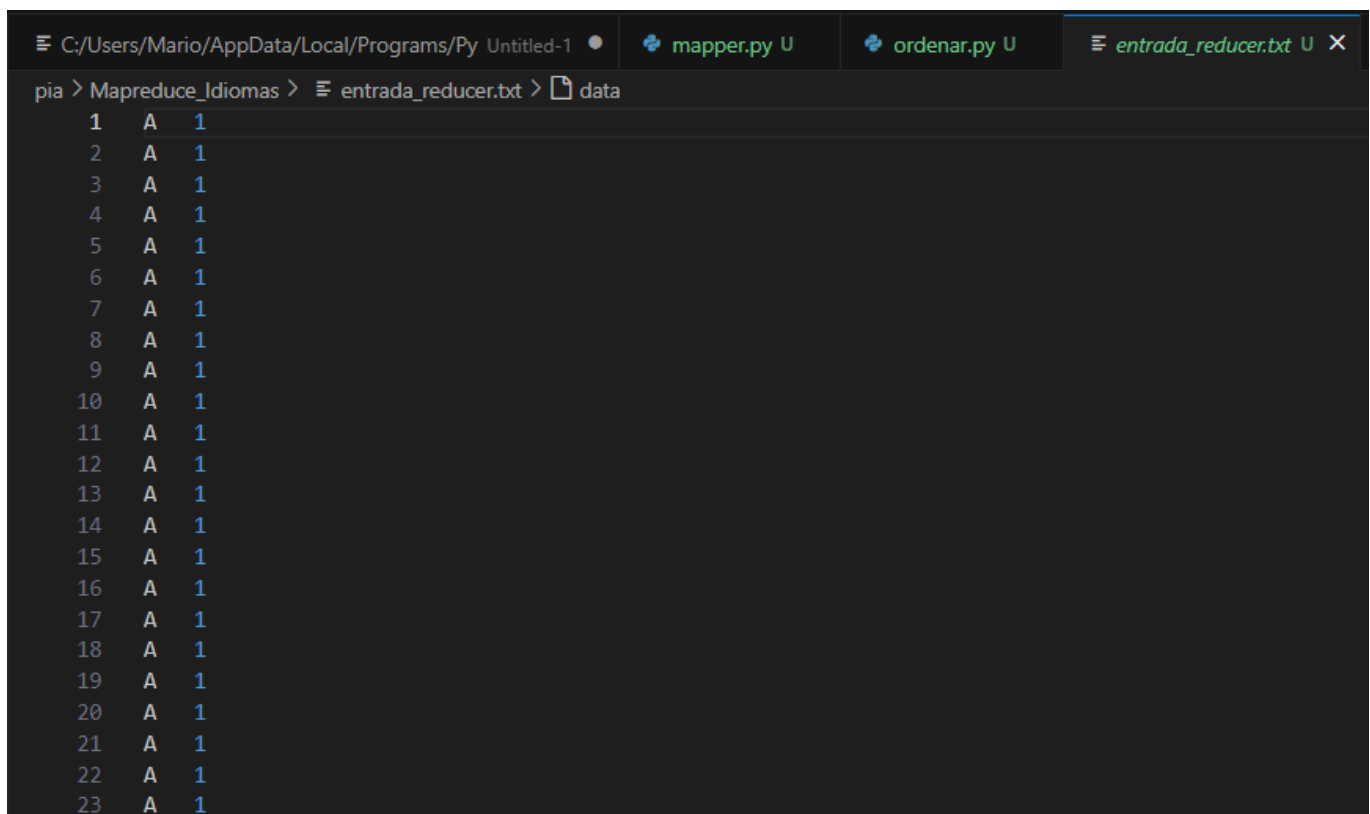
        with open("entrada_reducer.txt", "w", encoding="utf-8") as salida:
            salida.writelines(líneas)

        print("Archivo 'entrada_reducer.txt' creado correctamente.")

    except FileNotFoundError:
        print("Error: No se encontró el archivo 'salida_mapper.txt'. Asegúrate de ejecutar primero 'mapper.py'.")

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

Ordenamos teniendo en cuenta que en castellano la Ñ va después de la N indicando una clave de ordenación personalizada.



```
pia > Mapreduce_Idiomas > entrada_reducer.txt > data
1 A 1
2 A 1
3 A 1
4 A 1
5 A 1
6 A 1
7 A 1
8 A 1
9 A 1
10 A 1
11 A 1
12 A 1
13 A 1
14 A 1
15 A 1
16 A 1
17 A 1
18 A 1
19 A 1
20 A 1
21 A 1
22 A 1
23 A 1
```

3.- Crea un archivo llamado reducer.py que procese el “entrada_reducer.txt” y nos devuelva el número de apariciones de cada letra de nuestro texto en un archivo llamado “salida_reducer.txt”. Como respuesta muestra tu código comentado.

```
# reducer.py

def main():
    try:
        with open("entrada_reducer.txt", "r", encoding="utf-8") as entrada,
open("salida_reducer.txt", "w", encoding="utf-8") as salida: # Abrimos el archivo de
entrada y salida

            letra actual = None # Variable para almacenar la letra actual
            contador = 0 # Contador para sumar los valores de la misma letra

            for linea in entrada: # Iteramos sobre cada línea del archivo de entrada
                letra, valor = linea.strip().split('\t') # Separamos la letra y el
valor

                valor = int(valor)

                if letra == letra actual:
                    # Si es la misma letra que la anterior, sumamos
                    contador += valor
                else:
                    if letra_actual is not None:
                        # Guardamos la letra anterior y su contador
                        salida.write(f"{letra_actual}\t{contador}\n")
                        # Actualizamos la letra actual y su contador
                        letra actual = letra
                        contador = valor

                    # Guardamos la última letra después de terminar el bucle
                    if letra_actual is not None:
                        salida.write(f"{letra_actual}\t{contador}\n")

            print("Archivo 'salida_reducer.txt' creado correctamente.")

    except FileNotFoundError:
        print("Error: No se encontró el archivo 'entrada_reducer.txt'. Asegúrate de
ejecutar primero 'ordenar.py'.")

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

Abrimos el archivo entrada_reducer.txt y empezamos a recorrer cada línea separando las letras del contador. Si la letra leída coincide con la anterior sumamos el valor y cuando encontremos una nueva letra escribimos el valor y el contador de la letra anterior.

Al finalizar el bucle de líneas escribimos también la última letra con su contador.

```
C:\Users\Mario\AppData\Local\Programs\Py\ Untitled-1 • mapper.py U ordenar.py U reducer.py U salida_reducer.txt U X
pia > Mapreduce_Idiomas > salida_reducer.txt > data
1 A 3332
2 B 303
3 C 1792
4 D 1660
5 E 3819
6 F 216
7 G 239
8 H 158
9 I 2691
10 J 124
11 K 3
12 L 1235
13 M 687
14 N 2251
15 Ñ 16
16 O 3194
17 P 986
18 Q 214
19 R 2343
20 S 2980
21 T 1660
22 U 1370
23 V 344
24 W 74
25 X 66
26 Y 209
27 Z 85
28
```

4.- Prueba tu mapper y reducer (aplicando ordenar.py como fase intermedia) con el resto de los archivos de prueba y elabora una tabla con las apariciones de cada letra en los distintos idiomas.

Devuelve el resultado de cada trabajo mapreduce donde se vea la cantidad de veces que aparece cada letra del alfabeto y una la tabla con los resultados finales de los 4 idiomas.

Castellano:

```
salida_reducer_castellano.txt U X C:\Users\Mario\AppData\Local\Programs\Py\ U
pia > Mapreduce_Idiomas > castellano > salida_reducer_castellano.txt > data
1 A 3332
2 B 303
3 C 1792
4 D 1660
5 E 3819
6 F 216
7 G 239
8 H 158
9 I 2691
10 J 124
11 K 3
12 L 1235
13 M 687
14 N 2251
15 Ñ 16
16 O 3194
17 P 986
18 Q 214
19 R 2343
20 S 2980
21 T 1660
22 U 1370
23 V 344
24 W 74
25 X 66
26 Y 209
27 Z 85
28
```

Francés:

```
salida_reducer_francés.txt U X C:/Users/Mario/AppData/Local/Program
pia > Mapreduce_Idiomas > francés > salida_reducer_francés.txt > data
1 A 2289
2 B 206
3 C 1270
4 D 1289
5 E 5150
6 F 302
7 G 307
8 H 144
9 I 2713
10 J 33
11 K 6
12 L 1320
13 M 676
14 N 2503
15 O 2520
16 P 1057
17 Q 222
18 R 2257
19 S 3172
20 T 2700
21 U 1963
22 V 658
23 W 78
24 X 147
25 Y 82
26 Z 109
27 æ 3
28
```

Inglés:

```
salida_reducer_ingles.txt U X C:/Users/Mario/AppData/Local/Programs/Py
pia > Mapreduce_Idiomas > inglés > salida_reducer_ingles.txt > data
1 A 2106
2 B 372
3 C 1023
4 D 892
5 E 2931
6 F 516
7 G 415
8 H 759
9 I 2102
10 J 26
11 K 64
12 L 921
13 M 541
14 N 1712
15 O 2337
16 P 770
17 Q 14
18 R 2194
19 S 1838
20 T 2388
21 U 1149
22 V 376
23 W 403
24 X 67
25 Y 635
26 Z 16
27
```

Tabla comparativa con las distintas ocurrencias:

| Letra | Inglés | Castellano | Francés |
|-------|--------|------------|---------|
| A | 2106 | 3332 | 2289 |
| B | 372 | 303 | 206 |
| C | 1023 | 1792 | 1270 |
| D | 892 | 1660 | 1289 |
| E | 2931 | 3819 | 5150 |
| F | 516 | 216 | 302 |

| | | | |
|---|------|------|------|
| G | 415 | 239 | 307 |
| H | 759 | 158 | 144 |
| I | 2102 | 2691 | 2713 |
| J | 26 | 124 | 33 |
| K | 64 | 3 | 6 |
| L | 921 | 1235 | 1320 |
| M | 541 | 687 | 676 |
| N | 1712 | 2251 | 2503 |
| Ñ | – | 16 | – |
| O | 2337 | 3194 | 2520 |
| P | 770 | 986 | 1057 |
| Q | 14 | 214 | 222 |
| R | 2194 | 2343 | 2257 |
| S | 1838 | 2980 | 3172 |
| T | 2388 | 1660 | 2700 |
| U | 1149 | 1370 | 1963 |
| V | 376 | 344 | 658 |
| W | 403 | 74 | 78 |
| X | 67 | 66 | 147 |
| Y | 635 | 209 | 82 |
| Z | 16 | 85 | 109 |
| Ɛ | – | – | 3 |