ClaseArchivo

February 10, 2020

1 Práctica: Clase archivo

Alumno: Axel Suárez Polo
Matrícula: 201744436

En este práctica el objetivo es realizar un programa que calcula múltiples métricas de un archivo, para esto se realiza una clase para que se pueda reutilizar la lógica de manipulación.

En cada método de la clase se encuentra la documentación de cada método.

1.1 Estrategias de diseño

Puesto que el código para contar cuántas vocales, consonantes, signos de puntuación, espacios, mayúsculas y minúsculas es básicamente el mismo, esto se abstrae en el método más genérico llamado `Archivo.cuentaSi` que lo que hace es contar cuantos de los caracteres del archivo cumplen con la condición dada, esta condición es pasada con la construcción funcional del lenguaje `lambda`; con lo que se logra reducir considerablemente la cantidad de líneas de código.

También resulta que el código para convertir a minúsculas, mayúsculas y hexadecimal es casi idéntico; por lo que lo que se realizó fue un método más genérico llamado `Archivo.convertir` que realiza la conversión solamente con una función para transformar cada caracter.

[14]:
```
# Importar la función exit
from sys import exit

# Definición de la clase Archivo
class Archivo:
    def __init__(self, nombre):
        """Constructor de la clase Archivo, requiere un nombre
           de archivo existente para poder ser instanciada"""

        try:
            # Intentar abrir el archivo
            self.f = open(nombre, 'r')
            self.nombre = nombre
        except:
            # Si no se puede abrir el archivo, entonces se termina el programa
            print('No se puede abrir el archivo', nombre)
```

```
            exit()

    def muestra(self):
        """Muestra el archivo línea por línea, con su número
           de línea correspondiente"""

        # Mostrar línea por línea
        i = 1
        for linea in self.f:
            print(f'{i:3} {linea}', end='')
            i += 1

        # Regresar el apuntador del archivo al inicio
        self.f.seek(0)

    def cuentaSi(self, condicion):
        """Cuenta cuántos de los caracteres en el archivo cumplen con la␣
↪condición dada.
           Realiza una conversión a minúsculas antes de verificar la condición.
↪"""

        # Función interna para contar los caracteres que están en
        # el conjunto en una línea
        def contar(s):
            contador = 0

            # Verificación de cada caracter de la cadena
            for i in range(len(s)):
                if condicion(s[i]):
                    contador += 1

            return contador

        # Realizar el conteo en todo el archivo
        contador = 0
        for linea in self.f:
            contador += contar(linea)

        # Devolver el apuntador del archivo al inicio
        self.f.seek(0)

        return contador

    def cuentaVocales(self):
        """Cuenta cuántas vocales hay en el archivo"""
        return self.cuentaSi(lambda c: c.lower() in set('aeiouáéíóú'))
```

```
    def cuentaConsonantes(self):
        """Cuenta cuántas consonantes hay en el archivo"""
        return self.cuentaSi(lambda c: c.lower() in␣
↪set('bcdfghjklmnpqrstvwxyz'))

    def cuentaPuntuacion(self):
        """Cuenta cuántos signos de puntuación hay en el archivo"""
        return self.cuentaSi(lambda c: c.lower() in set('¿?¡!"\',.:;-_'))

    def cuentaEspacios(self):
        """Cuenta cuántos espacios hay en el archivo, incluyendo tabulaciones.
↪"""
        return self.cuentaSi(lambda c: c.lower() in set(' \t'))

    def cuentaMayusculas(self):
        """Cuenta cuántas letras mayúsculas hay en el archivo."""
        return self.cuentaSi(str.isupper)

    def cuentaMinusculas(self):
        """Cuenta cuántas letras minúsculas hay en el archivo."""
        return self.cuentaSi(str.islower)

    def cuentaPalabras(self):
        """Cuenta cuántas palabras hay en el archivo"""
        contador = 0

        # Por cada línea
        for linea in self.f:
            # Inicialmente no se está dentro de una palabra
            dentroPalabra = False

            # Por cada caracter de la línea
            for c in linea:
                # Si encontramos una letra y no estamos dentro de una palabra
                # Entonces ahora sí estamos en una palabra y es una nueva␣
↪palabra
                if c.isalpha() and not dentroPalabra:
                    dentroPalabra = True
                    contador += 1

                # Si encontramos algo que no es una letra y estabamos en una␣
↪palabra
                # Entonces ya terminó la palabra
                if not c.isalpha() and dentroPalabra:
                    dentroPalabra = False

        # Devolver el apuntador del archivo al inicio
```

```
        self.f.seek(0)

        return contador

    def cuentaLineas(self):
        """Cuenta cuántas líneas tiene el archivo"""
        contador = 0

        # Incrementar el contador de archivo por cada línea
        for _ in self.f:
            contador += 1

        # Devolver el apuntador del archivo al inicio
        self.f.seek(0)

        return contador

    def copiar(self, destino):
        """Copia el archivo actual al archivo con el nombre especificado"""
        # Abrir el archivo y copiar linea por linea
        with open(destino, 'w') as destino:
            for l in self.f:
                # Se usa print para incluír automaticamente el salto de línea
                print(l, file=destino)

        # Devolver el apuntador del archivo al inicio
        self.f.seek(0)

    def convertir(self, fun, sep=''):
        """Imprime cada uno de los carácteres del archivo después de aplicarles␣
↪la
           función dada, con el separador dado"""
        for linea in self.f:
            for c in linea:
                # Imprime cada caracter después de aplicarle la función
                print(fun(c), end=sep)

            # Imprimir cada salto de línea
            print()

        # Devolver el apuntador del archivo al inicio
        self.f.seek(0)

    def convertirMayusculas(self):
        """Muestra todo el archivo en mayúsculas"""
        self.convertir(str.upper)
```

```
    def convertirMinusculas(self):
        """Muestra todo el archivo en minúsculas"""
        self.convertir(str.lower)

    def convertirHexadecimal(self):
        """Muestra todo el archivo en hexadecimal"""
        self.convertir(lambda c: hex(ord(c))[2:], sep=' ')
```

[15]:
```
# Abrir y mostrar el archivo con sus números de línea
nombre = input('Nombre del archivo:')
archivo = Archivo(nombre)

archivo.muestra()
```

```
Nombre del archivo:ejemplo.txt
  1 Este es un archivo de prueba para la clase archivo.
  2 Es evidente el número de líneas, pero no es tan evidente la cantidad de:
  3     Vocales, Minúsculas, Mayúsculas, etc.
  4
  5 ¿O sí lo es?
```

[16]:
```
# Imprimir estadísticas del archivo
print('Vocales:', archivo.cuentaVocales())
print('Consonantes:', archivo.cuentaConsonantes())
print('Signos de puntuación:', archivo.cuentaPuntuacion())
print('Espacios:', archivo.cuentaEspacios())
print('Palabras:', archivo.cuentaPalabras())
print('Líneas:', archivo.cuentaLineas())
print('Mayúsculas:', archivo.cuentaMayusculas())
print('Minúsculas:', archivo.cuentaMinusculas())
```

```
Vocales: 62
Consonantes: 73
Signos de puntuación: 9
Espacios: 32
Palabras: 32
Líneas: 5
Mayúsculas: 6
Minúsculas: 129
```

[18]:
```
# Copiar el archivo
destino = input('Nombre de la copia: ')
archivo.copiar(destino)

# Cargar la copia y mostrarla
copia = Archivo(destino)
print('Contenido de la copia:')
```

```
copia.muestra()
```

```
Nombre de la copia: copia.txt
Contenido de la copia:
  1 Este es un archivo de prueba para la clase archivo.
  2
  3 Es evidente el número de líneas, pero no es tan evidente la cantidad de:
  4
  5     Vocales, Minúsculas, Mayúsculas, etc.
  6
  7
  8
  9 ¿O sí lo es?
```

[22]:
```
# Muestra el contenido de todo el archivo en hexadecimal
archivo.convertirHexadecimal()
```

```
45 73 74 65 20 65 73 20 75 6e 20 61 72 63 68 69 76 6f 20 64 65 20 70 72 75 65 62
61 20 70 61 72 61 20 6c 61 20 63 6c 61 73 65 20 61 72 63 68 69 76 6f 2e a
45 73 20 65 76 69 64 65 6e 74 65 20 65 6c 20 6e fa 6d 65 72 6f 20 64 65 20 6c ed
6e 65 61 73 2c 20 70 65 72 6f 20 6e 6f 20 65 73 20 74 61 6e 20 65 76 69 64 65 6e
74 65 20 6c 61 20 63 61 6e 74 69 64 61 64 20 64 65 3a a
20 20 20 20 56 6f 63 61 6c 65 73 2c 20 4d 69 6e fa 73 63 75 6c 61 73 2c 20 4d 61
79 fa 73 63 75 6c 61 73 2c 20 65 74 63 2e a
a
bf 4f 20 73 ed 20 6c 6f 20 65 73 3f
```

[19]:
```
# Muestra todo el contenido del archivo en minúsculas
archivo.convertirMinusculas()
```

```
este es un archivo de prueba para la clase archivo.

es evidente el número de líneas, pero no es tan evidente la cantidad de:

    vocales, minúsculas, mayúsculas, etc.



¿o sí lo es?
```

[20]:
```
# Muestra todo el contenido del archivo en mayúsculas
archivo.convertirMayusculas()
```

```
ESTE ES UN ARCHIVO DE PRUEBA PARA LA CLASE ARCHIVO.

ES EVIDENTE EL NÚMERO DE LÍNEAS, PERO NO ES TAN EVIDENTE LA CANTIDAD DE:

    VOCALES, MINÚSCULAS, MAYÚSCULAS, ETC.
```

¿O SÍ LO ES?