

Alumno: Carlos Daniel Cruz Pino

Matricula: 201617962

Materia: Programación Concurrente y Paralela

Tarea: Reporte de Clase archivo

Realizar un programa que contenga la clase Archivo, con los métodos:

- Cuenta vocales
- Cuenta consonantes
- Cuenta signos de puntuación
- Cuenta espacios
- Cuenta palabras
- Cuenta líneas
- Mayúsculas
- Minúsculas
- Copia archivo
- Convierte a mayúsculas
- Convierte a minúsculas
- Muestra en hexadecimal

Clase principal (Main)

La función primordial de esta clase es la de mandar a llamar a las funciones de la clase archivo para que se ejecuten, para ello lo primero que se hace es importar el archivo "archivo1.py" que es el que contiene a la clase Archivo.

```
from archivo1 import *
```

Otra de sus funciones es pedir el nombre o la dirección del archivo de texto que ocuparemos para ejecutar las funciones de la clase Archivo; en el caso de que solo se introduzca el nombre del archivo, este tiene que estar guardado en la misma carpeta que el programa principal (main.py) y el programa archivo (archivo1.py), si no lo está entonces aparecerá el siguiente letrero:

No se puede abrir el archivo

La otra opción es introducir la dirección en la que se encuentra guardado el archivo.txt.

Posteriormente se crea un objeto de tipo Archivo, el cual nos servirá para hacer uso de las funciones de la clase Archivo.

```
archivo=Archivo(nomb)
```

Teniendo esto, ya solo se ejecutan las funciones de la clase Archivo, algunas apoyándose de la función `print()` para imprimir los datos y poner letreros.

[illegible]

Clase Archivo

Esta clase contiene algunas funciones de manipulación de archivos, tales como contar vocales, consonantes, signos de puntuación, espacios, palabras, líneas, mayúsculas, minúsculas, copiar el archivo, convertir el texto a mayúsculas, minúsculas y a formato hexadecimal.

Para su funcionamiento importamos el módulo **shutil**, que incluye operaciones de archivos de alto nivel como copiar y archivar, y el modulo **sys** que provee acceso a funciones y objetos mantenidos por del intérprete.

```
import shutil
from sys import exit
```

Constructor

En esta parte se es definido el constructor de la clase, el cual recibe como parámetro el nombre o dirección del archivo de origen bajo el cual se trabajará, en él se encuentra también la función de abrir el archivo y un **try**, el cual funciona de tal manera que, si no encuentra el archivo especificado o no lo puede abrir, entonces imprime en pantalla un letrero que dice que no se puede abrir el archivo

```
class Archivo: #Aquí se define la clase archivo, así como su constructor
    def __init__(self,nombre):
        try:
            self.f=open(nombre,'r') #Abrimos el archivo con el nombre dado
            self.nombre=nombre
        except:
            print("No se puede abrir el archivo",nombre) #en caso de que no se encuentre
            exit()
```

Función muestra()

Esta función lo que hace es mostrar el contenido del archivo de texto dado, la impresión va acompañada de un contador, el cual indica el numero de línea.

```
def muestra(self):
    i=1
    for linea in self.f:
        print("{:3}:{:}".format(i,linea)) #imprime linea a linea el contenido del arch
ivo enumerando cada linea
        i+=1 #incrementa el contador segun el numero de lineas
    self.f.seek(0) #nos sirve para ir al byte 0 del fichero
```

Ejecución:

```
Nombre del archivo: p1.txt

>>>>>>>>TEXTO ORIGINAL<<<<<<<<<
1:Esto es un texto de prueba.
```

Función cuentaVocales()

Esta función retorna el numero de vocales que encuentra en el archivo de texto, para ello hace uso de un for para recorrer el archivo, dentro del for tambien se hace uso de la funcion len(), la cual nos devuelve el numero de elementos en el archivo.

```
def cuentaVocales(self):
    def vocales(s):
        contador=0
        for i in range(len(s)): #recorre el archivo
            if s[i] in set("aeiouáéíóúAEIOUÁÉÍÓÚ"): #verifica cada posicion del
                                                    #archivo en busca de vocales
                contador += 1 #incrementa el contador en caso de encontrar alguna
        return contador
    contador=0
    for linea in self.f:
        contador += vocales(linea)
    self.f.seek(0) #nos sirve para ir al byte 0 del fichero
    return contador
```

Función cuentaConsonantes()

Esta función recorre el archivo con la ayuda de un for y hace comparaciones de cada posición con los elementos del conjunto de las consonantes, en caso de que se encuentre alguna consonante se incrementa el contador, esta función retorna el número de consonantes encontradas.

```
def cuentaConsonantes(self):
    def consonantes(s):
        contador=0
        for i in range(len(s)): #recorre el archivo
            if s[i] in set("bcd fghjklmnñpqrstvwxyzBCDFGHJKLMNÑPQRSTUVWXYZ"): #verifica
                                                    #cada posicion del archivo en busca de consonantes
                contador += 1 #incrementa el contador en caso de encontrar alguna
                                                    #consonante
        return contador
    contador=0
    for linea in self.f:
        contador += consonantes(linea)
    self.f.seek(0) #nos sirve para ir al byte 0 del fichero
    return contador
```

Función cuentaSignosP()

Su función es la de buscar en el archivo signos de puntuación para posteriormente retornar el número de estos. Para realizar su función se utiliza un for para recorrer el archivo y con la ayuda de una sentencia if se compara cada una de las posiciones del archivo con los elementos del conjunto de signos de puntuación, en caso de encontrar alguno, se incrementa la variable contador, para posteriormente retornarla al acabar de recorrer el archivo.

```
def cuentaSignosP(self):
    def signos(s):
        contador=0
        for i in range(len(s)): #recorre el archivo
            if s[i] in set(".,:;[]!()?"): #verifica cada posicion del archivo
                                                #en busca de algun elemento del conjunto
                contador += 1 #incrementa el contador en caso de encontrar algun
                                #elemento del conjunto anterior
        return contador
    contador=0
    for linea in self.f:
        contador += signos(linea)
    self.f.seek(0) #nos sirve para ir al byte 0 del fichero
    return contador
```

Función cuentaEspacios()

Esta función se encarga de recorrer el archivo y contar el numero de espacios existentes en el mismo, para ello se utiliza el mismo mecanismo que las funciones anteriores, con un for para recorrer el archivo y un if para comparar si la posición en que se encuentra es un espacio y al encontrar alguno se incrementa la variable contador, al final retorna la variable contador.

```
def cuentaEspacios(self):
    def espacios(s):
        contador=0
        for i in range(len(s)): #recorre el archivo
            if s[i] in set(" "): #verifica si en esa posicion del archivo hay un espacio
                contador += 1 #incrementa en caso de encontrar espacios
        return contador
    contador=0
    for linea in self.f:
        contador += espacios(linea)
    self.f.seek(0) #nos sirve para ir al byte 0 del fichero
    return contador
```

Función cuentaPalabras()

Esta función recorre el archivo y cuenta las palabras existentes en el texto de este. A diferencia de las funciones anteriores, esta es mas extensa, esto se debe a que se implementan varios if y else, esto debido a que la lógica de este algoritmo se basa en contar espacios en el texto y se contara como palabra siempre y cuando se cumplan todas las condiciones establecidas.

Primer if:

En esta sentencia preguntamos si en la posición actual del archivo hay un espacio, si es asi, revisa que no sea la primera posición en el archivo, si es correcto, verifica que la posición anterior en la que se está en el archivo no sea un espacio, al cumplir todas las condiciones se da por hecho de que es una palabra y se incrementa el contador.

```
if s[i] == ' ' and i > 0 and s[i - 1] != ' ':  
    contador += 1
```

Segundo if:

En caso de que no se haya cumplido el if anterior entonces se ejecuta este.

En este caso se compara que la posición actual del archivo sea igual a un salto de línea, si es asi, entonces es una palabra y se procede a incrementar el contador.

```
if s[i] == "\n":  
    contador += 1
```

tercer if:

dentro de este if existe un cuarto, el tercero lo que hace es comparar si el contador "i" es igual a la posición final en el archivo, si se cumple entonces se procede a la siguiente sentencia, en esta se verifica que la posición anterior a la actual en el archivo sea diferente de un salto de línea y que la posición anterior a la anterior de la posición actual sea diferente de un espacio, si se cumple lo anterior entonces es una palabra, por lo que incrementamos el contador.

```
if i == tot-1:  
    if s[tot-1] != "\n" and s[tot-2] != ' ':  
        contador += 1
```

código completo de cuentaPalabras():

```
def cuentaPalabras(self):  
    def palabras(s):  
        tot = len(s) #le asignamos a tot el numero total de posiciones en el archivo  
        contador = 0  
        for i in range(len(s)): #recorre el archive
```

```

#verifica que se cumplan las condiciones
    if s[i] == ' ' and i > 0 and s[i - 1] != ' ':
        contador += 1 #aumenta el contador si se cumplen las
                        #condiciones anteriores
    else:
        if s[i] == "\n": #si lo anterior fue falso, entonces se hace esta
                        #comparacion
            contador += 1 #incrementa en caso verdadero
        else:
#en otro caso verifica que seala ultima posicion del archivo
        if i == tot-1:
#verifica si se cumplen las condiciones
            if s[tot-1] != "\n" and s[tot-2] != ' ':
                contador += 1 #en caso de cumplirse, se umenta el contado

    return contador
contador = 0
for linea in self.f:
    contador += palabras(linea)
self.f.seek(0) #nos sirve para ir al byte 0 del fichero
return contador

```

Función cuentaLineas()

Esta función recorre cada una de las líneas del archivo de texto incrementando el contador por cada una que encuentre, retorna el número total de líneas en el archivo.

```

def cuentaLineas(self):
    i=0
    for linea in self.f:
        i+=1 #si es una linea se aumenta el contador
        conta = "{:3}".format(i) #se le asigna a la variable "conta" el contenido de "i"
ya con formato de salida
        self.f.seek(0) #nos sirve para ir al byte 0 del fichero
    return conta #se retorna el numero de lineas en el archivo

```

Función cuentaMayusculas()

Esta función recorre todo el archivo comparando cada posición de este con los elementos del conjunto de mayúsculas incluidas en la sentencia if, en caso de encontrar alguna mayúscula se incrementa el contador, retorna la variable conta con el numero total de mayúsculas encontradas en el archivo de texto.

```
def cuentaMayusculas(self):
    def mayus(s):
        conta = 0
        for i in range(len(s)): #recorre el archivo
            if s[i] in set ("ABCDEFGHIJKLMNÑOPQRSTUVWXYZ"): #"verifica que en la
posicion s[i] contenga alguna letra mayuscula"
                conta += 1 #se incrementa el contador en caso de encontrar alguna
            return conta
        conta = 0
        for linea in self.f:
            conta += mayus(linea)
        self.f.seek(0) #nos sirve para ir al byte 0 del fichero
        return conta
```

Función cuentaMinusculas()

Su funcionamiento es prácticamente igual que la función cuentaMayusculas(), la diferencia es que en esta, las comparaciones se hacen sobre un conjunto de letras minúsculas y al final con la función print() se imprime en pantalla el resultado obtenido.

```
def cuentaMinusculas(self):
    def minusculas(s):
        contador = 0
        for i in range(len(s)): #recorre el archivo
            if s[i] in set ("abcdefghijklmnopqrstuvwxyzáéíóú"): #verifica que en la
posicion s[i] contenga alguna letra minuscula
                contador += 1 #se incrementa el contador en caso de encontrar alguna
            return contador
        contador = 0
        for linea in self.f:
            contador += minusculas(linea)
        self.f.seek(0) #nos sirve para ir al byte 0 del fichero
        print("Minusculas: ",contador)
```


Función copiarArchivo()

Como primer argumento en esta función se le pide al usuario teclear el nombre o la dirección del archivo de destino al que se quiere copiar el archivo origen, una vez obtenido la dirección de dicho archivo de destino se procede a hacer uso de la función “copy” del modulo shutil que importamos en el inicio del programa, la cual nos sirve para copiar el contenido de un archivo origen a un archivo destino.

[illegible]

Funcion convierteEnMayusculas() y convierteEnMinusculas()

Estas funciones la he puesto juntas en este apartado, ya que su código es casi el mismo, a diferencia de que para convertir el texto en mayúsculas utilizamos una función propia de Python, la cual es `upper()` y para convertir a minúsculas se utiliza la función `lower()`, para la conversión se recorre línea a línea el archivo aplicando la función anterior e imprimiendo el resultado en pantalla

```
#Convierte a mayusculas
def convierteEnMayusculas(self):
    for linea in self.f:
        print(linea.upper()) #esta funcion convierte el texto a mayusculas y lo imprim
me
        self.f.seek(0) #nos sirve para ir al byte 0 del fichero

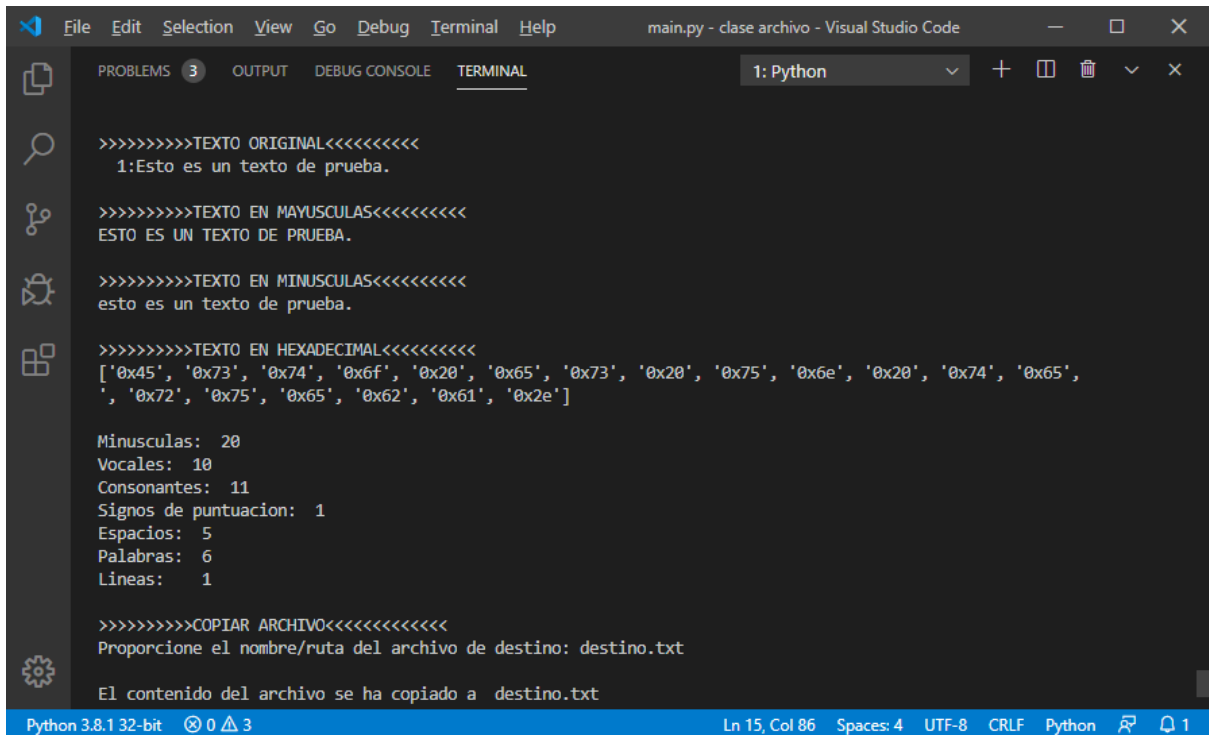
#Convierte en minusculas
def convierteEnMinusculas(self):
    for linea in self.f:
        print(linea.lower())#esta funcion convierte el texto a mayusculas y lo imprim
e
        self.f.seek(0) #nos sirve para ir al byte 0 del fichero
```

Función Hexadecimal()

Esta función se encarga de mostrar en pantalla el texto del archivo leído en formato hexadecimal.

```
def Hexadecimal(self):
    h = []
    def hexa(s):
        for i in range(len(s)): #recorre el archivo
            h.append(hex(ord(s[i]))) #convierte a formato hexadecimal
    for linea in self.f:
        hexa(linea)
    return h
    self.f.seek(0) #nos sirve para ir al byte 0 del fichero
```

Ejecución de todas las funciones de la clase Archivo:



```
File Edit Selection View Go Debug Terminal Help main.py - clase archivo - Visual Studio Code
PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL 1: Python
>>>>>>>TEXTO ORIGINAL<<<<<<<<<<<<
1:Esto es un texto de prueba.

>>>>>>>TEXTO EN MAYUSCULAS<<<<<<<<<<<<
ESTO ES UN TEXTO DE PRUEBA.

>>>>>>>TEXTO EN MINUSCULAS<<<<<<<<<<<<
esto es un texto de prueba.

>>>>>>>TEXTO EN HEXADECIMAL<<<<<<<<<<<<
['0x45', '0x73', '0x74', '0x6f', '0x20', '0x65', '0x73', '0x20', '0x75', '0x6e', '0x20', '0x74', '0x65',
', '0x72', '0x75', '0x65', '0x62', '0x61', '0x2e']

Minusculas: 20
Vocales: 10
Consonantes: 11
Signos de puntuacion: 1
Espacios: 5
Palabras: 6
Lineas: 1

>>>>>>>COPIAR ARCHIVO<<<<<<<<<<<<
Proporcione el nombre/ruta del archivo de destino: destino.txt

El contenido del archivo se ha copiado a destino.txt

Python 3.8.1 32-bit 0 3 Ln 15, Col 86 Spaces: 4 UTF-8 CRLF Python 1
```

Código completo:

<https://github.com/danielcruzp/Clase-Archivo>