Status	Finished
Started	Wednesday, 27 November 2024, 6:38 PM
Completed	Wednesday, 27 November 2024, 6:43 PM
Duration	5 mins 8 secs
Marks	5.00/5.00
Grade	**10.00** out of 10.00 (**100**%)

Information

Leer un fichero en una única string

A veces, necesitamos leer el contenido de un archivo de entrada como una sola string:

```
f = open('poem.txt', 'r')
text = f.read()
print(len(text))           #  total number of characters
print(text)                #  contents of the file
f.close()
```

El método **read()** puede tener como parámetro el número de caracteres a leer:

```
f = open('poem.txt','r')
print(f.read(1))            # first character
print(f.read(5))            # next 5 characters
print(f.read())             # the rest of the file's contents
f.close()
```

Question 1

Complete

Mark 1.00 out of 1.00

El archivo de entrada *urls.txt* contiene las 3 líneas siguientes:

www.ukf.sk
www.bbc.com
www.google.sk

Analice el siguiente programa:

```
file = open('urls.txt', 'r')
print(file.read(3), end=']\n')          # línea 1
print(repr(file.read()), end=']\n')     # línea 2
print(file.readline(), end=']\n')       # línea 3
file.close()
```

Empareje números de líneas que contiene instrucciones `print`, con los resultados correctos.

línea 1 [www] ▾

linea 2 ['.ukf.sk\nwww.bbc.com\nwww.google.sk\n'] ▾

línea 3 [ ] ▾

Information

Escribir en un fichero de texto

Al abrir el fichero con `open()` especificaremos el modo de escritura mediante el carácter `'w'` como segundo parámetro. Si el archivo no existe, se creará vacío, y aun cuando existiera previamente, su contenido será sobreescrito.

Una vez abierto, podemos escribir usando el método **`write()`**, cuyo parámetro ha de ser una **string**. Al usarlo, deberemos incluir explícitamente todos los caracteres de fin de línea que queramos contenga el fichero de salida:

```
f = open('output.txt', 'w')
f.write('We are learning\n')
f.write('about using text files\n')
f.write('in Python\n')
f.close()>
```

Las mismas tres líneas de texto se podrían igualmente haber enviado al archivo llamando al método `write()` una sola vez:

```
f = open('output.txt', 'w')
f.write('We are learning\nabout using text files\nin Python\n')
f.close()
```

Hay que **acordarse de cerrar siempre los archivos abiertos**. En caso de no cerrarse, no se puede estar seguro de si lo que hemos mandado escribir realmente acaba guardado en el disco.

Question 2

Complete

Mark 1.00 out of 1.00

Elija la explicación correcta de la salida del programa para el valor de entrada n = 5:

```
import random
fw = open('random_numbers.txt', 'w')
n = int(input('n = '))
for i in range(n):
    fw.write(str(random.randint(1, 9)))
fw.close()
```

Select one:

- El fichero de salida contiene 5 líneas con un número aleatorio (random number) en cada una
- La función `str()` no puede invocarse como se hace en el programa
- El fichero de salida contiene una línea con 5 números aleatorios (random numbers)

Information

También se puede escribir en un fichero de texto usando la función **`print()`**. Podemos redireccionar la salida usando un parámetro opcional para escribir en un fichero, en vez de en la salida estándar:

```
f = open('primes.txt', 'w')
for number in (2, 3, 5, 7, 11, 13, 17, 19):
    print(number, end = ' ', file = f)
f.close()
```

El archivo primes.txt contendrá una sola línea conteniendo varios números separados por un espacio:

```
2 3 5 7 11 13 17 19
```

Si no se hubiese especificado el parámetro `end` cada número estaría en una línea diferente.

Obsérvese que mientras que `write()` requiere que el parámetro sea string, `print()` acepta **otros tipos** convirtiéndolos a string antes de escribir, ya sea en la terminal, como sabemos, o en el fichero de texto que se indique.

Question 3

Complete

Mark 1.00 out of 1.00

Seleccione la explicación correcta del resultado del siguiente programa:

```
fr = open('input.txt', 'r')
fw = open('output.txt', 'w')

for line in fr:
    if line != '\n':
        print(line, end='', file=fw)
    print(line, end='')

fr.close()
fw.close()
```

Select one:

- El archivo de salida contiene todas las líneas no vacías del archivo de entrada, que también se muestran en pantalla.
- Todas las líneas del archivo de entrada se copian en el archivo de salida y se muestran en pantalla.
- El archivo de salida contiene todas las líneas no vacías del archivo de entrada. Todas las líneas del fichero de entrada se muestran en pantalla

Information

La cláusula `with` con ficheros

Cuando se trabaja con archivos, se recomienda usar la cláusula **`with`**:

```
with open('poem.txt','r', encoding='utf-8') as f:
    for line in f:
        print(line, end='')
```

En el ejemplo anterior, el archivo abierto resulta referenciado con la variable `f`. Al salir del bloque `with` (ya sea porque se alcanza el final normalmente o cualquier otro motivo, como p.e. la ejecución en su interior de `break` o `return`) el archivo se **cierra automáticamente**.

Veamos cómo copiar con `with` el contenido de un archivo. El primer archivo debe abrirse para leer, el segundo para escribir:

```
with open('original.txt', 'r') as fr:
    with open('copy.txt', 'w') as fw:
        fw.write(fr.read())
```

También podríamos haber enumerado en un solo `with` los archivos que se manejan:

```
with open('original.txt', 'r') as fr, open('copy.txt', 'w') as fw:
    fw.write(fr.read())
```

Question 4

Complete

Mark 1.00 out of 1.00

Estudie el siguiente código:

```
with open('input.txt', 'r') as fr, open('output.txt', 'w') as fw:
    for line in fr:
        if line != '\n':
            print(line[::-1], end='', file=fw)
        print(line[::-1], end='')
    fw.close()
```

Marque todas las afirmaciones correctas.

Select one or more:

- [ ] Hemos olvidado cerrar el archivo fr
- [x] La instrucción `fw.close()` es redundante y debería quitarse
- [ ] El programa copia todas las líneas del fichero de entrada al de salida, dándoles la vuelta
- [x] El programa copia todas las líneas no vacías del fichero de entrada al de salida, dándoles la vuelta

Information

Adición de contenido a un archivo existente

Además de los modos de lectura y escritura, también es posible agregar contenido a un archivo.

Supongamos que las siguientes 3 líneas (todas terminadas en final de línea) constituyen el contenido de un cierto archivo llamado *names.txt*:

John
Paul
George

Abriremos este archivo en el modo ***append*** (añadir) usando `'a'` como segundo parámetro:

```
file = open('names.txt', 'a')
file.write('Ringo\n')
file.close()
```

Al abrir el archivo en modo añadir, la posición de escritura se coloca al final del mismo. En el ejemplo, a partir de ahí se le añade una línea, que será la 4ª del archivo; y finalmente se cierra.

El contenido del archivo *names.txt* resulta:

John
Paul
George
Ringo

Question 5

Complete

Mark 1.00 out of 1.00

Nos gustaría contar el número total de líneas en un archivo y agregar esa información a su final:

```
with open('names.txt', 'r') as f:
    text = f.read()

print(repr(text))   # 'John\nPaul\nRingo\nGeorge\n'

with open('names.txt', '  a  ') as f:
    n = text.count('\n')
    f.write(str(n))
```

Complete el programa anterior correctamente.