UNIVERSITAT JAUME I

1.º Ingeniería Informática
1.º Matemática Computacional
1.º Inteligencia Robótica
EI1003/MT1003/IR2103 Programación I
Práctica 6

En los ejercicios de este boletín vas a abrir para lectura y para escritura algunos ficheros de texto. Recuerda que Python 3 utiliza UTF-8 como codificación de texto por omisión. Si usas un sistema operativo distinto a Linux o macOS, deberás abrir los ficheros con `open(nombre_fichero, modo, encoding='utf-8')` para asegurarte de que abres el fichero con la codificación UTF-8.

Ejercicio 1

Escribe un programa que pida el nombre de un fichero de texto, lea ese fichero y escriba por pantalla el número de dígitos (caracteres del `0` al `9`) y el número de letras del alfabeto inglés (caracteres de la `A` a la `Z` en minúscula o mayúscula) que contiene. No olvides (ni en este ejercicio ni en ningúno otro) cerrar todos los ficheros que abras.

He aquí un ejemplo de ejecución del programa:

```
Introduce el nombre de un fichero: test.txt
Dígitos decimales: 800
Letras del inglés: 962
```

Ejercicio 2

Echa un vistazo al fichero `test.txt`, que contiene información relativa a un examen de tipo test llevado a cabo por una centena de alumnos. La primera línea del fichero contiene la plantilla de corrección del test y cada una de las líneas restantes contiene el número de DNI de un alumno, un símbolo almohadilla (`#`) como separador y las respuestas de ese alumno[1]. Llamaremos *fichero de test* a cualquier fichero de texto que siga una estructura análoga a la de `test.txt`, con cero o más alumnos.

Escribe un programa que pida el nombre de un fichero de test, lea ese fichero y escriba por pantalla para cada alumno una frase sobre su DNI y sus respuestas según el formato que se observa en el siguiente ejemplo de ejecución:

```
Introduce el nombre de un fichero de test: t1.txt
El alumno con DNI 17634552U ha respondido cbababcd**
El alumno con DNI 54433747O ha respondido a*a*a*aaaa
El alumno con DNI 78476844H ha respondido cbbaabcdcc
El alumno con DNI 99550296L ha respondido *bbbbbbbbb
El alumno con DNI 14758126V ha respondido cbba*bcdcc
Un alumno perfecto habría respondido cbbaabcdcc
```

En el ejemplo también se observa una frase final que el programa debe escribir haciendo uso de la plantilla de corrección del test.

Pista: Recuerda que `cadena.split('#')` devuelve una lista con las subcadenas de `cadena` usando el carácter `#` como separador.

Ejercicio 3

Escribe un programa que pida el nombre de un fichero de test, lea ese fichero y escriba por pantalla la nota numérica obtenida por cada alumno, en principio según esta fórmula:

```
nota = 10 * (aciertos - fallos) / número_total_de_preguntas
```

Sin embargo, si el resultado de la fórmula es negativo, la nota finalmente asignada al alumno debe ser cero.

El programa debe utilizar la función `evaluación_test` que escribiste en el ejercicio 17 de la práctica 4, *sin modificarla*. Recuerda que esa función recibía como parámetros la plantilla y las respuestas de un alumno y, en

[1] Como en prácticas anteriores, cada respuesta se representa mediante un único carácter y las preguntas no contestadas se representan mediante asteriscos.

condiciones normales, devolvía una lista con el número de *aciertos*, el de *fallos* y el de *respuestas en blanco*. Puedes suponer que al programa siempre se le proporcionarán ficheros de test correctos y, por lo tanto, `evaluación_test` nunca le devolverá `None`. Además, el programa debe llamar *una única vez* a `evaluación_test` por cada alumno del fichero.

Finalmente, los mensajes del programa deben respetar estrictamente los formatos que ilustra el siguiente ejemplo de ejecución, incluyendo los decimales de las notas:

```
Introduce el nombre de un fichero de test: t1.txt
El alumno con DNI 17634552U ha obtenido una nota de  4.00 puntos
El alumno con DNI 544337470 ha obtenido una nota de  0.00 puntos
El alumno con DNI 78476844H ha obtenido una nota de 10.00 puntos
El alumno con DNI 99550296L ha obtenido una nota de  0.00 puntos
El alumno con DNI 14758126V ha obtenido una nota de  9.00 puntos
```

Ejercicio 4

Modifica una copia del programa anterior para cambiar el formato con el que se muestran las notas de los alumnos. Ahora habrá que escribir `NO APTO` para las notas menores a 5 puntos y `APTO` seguido de la correspondiente nota numérica entre paréntesis para las restantes. He aquí un ejemplo de ejecución del programa:

```
Introduce el nombre de un fichero de test: t1.txt
DNI: 17634552U NOTA: NO APTO
DNI: 544337470 NOTA: NO APTO
DNI: 78476844H NOTA: APTO (10.00)
DNI: 99550296L NOTA: NO APTO
DNI: 14758126V NOTA: APTO ( 9.00)
```

Ejercicio 5

Modifica una copia del programa anterior para que, en lugar de mostrar las notas por pantalla, las escriba en el fichero `notas.txt`. Así pues, la salida por pantalla será realmente breve, como ilustra este ejemplo de ejecución:

```
Introduce el nombre de un fichero de test: t1.txt
```

Todo lo demás deberá escribirse en el fichero `notas.txt`.

Ejercicio 6

Escribe un programa que pida el nombre de un fichero de test, lea ese fichero, construya una lista con los DNI de todos los alumnos, la ordene utilizando *sin modificar* el procedimiento `ordenar_lista` que escribiste en el ejercicio 11 de la práctica 4 y muestre por pantalla un listado con todos los DNI ordenados, uno por línea. He aquí un ejemplo de ejecución:

```
Introduce el nombre de un fichero de test: t1.txt
14758126V
17634552U
544337470
78476844H
99550296L
```

Nota: Aunque el procedimiento `ordenar_lista` lo escribieras inicialmente para ordenar listas de enteros, es harto improbable que lo hayas escrito de forma que no ordene igual de bien cadenas, que es lo que son los DNI.

Ejercicio 7

Modifica una copia del programa anterior para que, en lugar de mostrar los DNI por pantalla, los escriba en el fichero `dni.txt`. La salida por pantalla será tan breve como ilustra este ejemplo de ejecución:

```
Introduce el nombre de un fichero de test: t1.txt
```

Ejercicio 8

Escribe un programa que pida el nombre de un fichero de texto, lea ese fichero y muestre por pantalla un mensaje indicando la secuencia de caracteres sin espacios en blanco más larga encontrada en él. En caso de empate, el programa debe preferir la primera de las cadenas empatadas. Recuerda que `cadena.split()` devuelve una lista con todas las subcadenas de `cadena` que no contienen espacios en blanco.

En el siguiente ejemplo de ejecución se observa el formato deseado para los mensajes:

```
Introduce el nombre de un fichero: quijote.txt
Su secuencia de caracteres sin espacios más larga es "bienintencionadamente".
```

Ejercicio 9

Modifica una copia del programa anterior para que, de todas las subcadenas resultantes de trocear con el método `split()` cada línea del fichero de texto, en lugar de quedarse con una de las más largas, muestre por pantalla todas las cadenas formadas íntegramente por dígitos. Para ello, debes hacer uso, *sin modificarla*, de la función `todos_dígitos` que escribiste en el ejercicio 17 de la práctica 3.

He aquí un ejemplo de ejecución:

```
Introduce el nombre de un fichero: quijote.txt
1604
17
23
16
```