IES SAN JUAN DE LA RAMBLA
DEPARTAMENTO DE INFORMÁTICA Y COMUNICACIONES
CFGS DESARROLLO DE APLICACIONES MULTIPLATAFORMA
MÓDULO: PROGRAMACIÓN

San Juan de la Rambla a 17 de octubre de 2022

UT01 - Act4 Resolución de problemas en Python

Ejercicio 01 (4 puntos)

Escriba un programa en Python que lea dos palabras desde el teclado e imprima por pantalla las veces que aparece cada letra de la primera palabra en la segunda.

Tener en cuenta que NO se pueda preguntar por una letra ya revisada anteriormente.

Como sugerencia deberían pasar todos los caracteres a minúsculas o mayúsculas antes de empezar a realizar las búsquedas.

Un ejemplo de la ejecución del programa sería::

Introduzca una palabra: sandía
Introduzca una segunda palabra: zanahoria

> s no está en “zanahoria”
> a está en “zanahoria” 3 veces
> n está en “zanahoria”
> d no está en “zanahoria”
> í no está en “zanahoria”

Fíjense que ya no se da información de la última “a” de sandía, porque ya ha sido revisada anteriormente. En el caso de la “i”, en sandía está acentuada y en zanahoria no por eso no sale.

Existen métodos en Python para los string que pueden ayudar en la realización de este ejercicio.

`s.count(<sub>[, <start>[, <end>]])`

> Counts occurrences of a substring in the target string.

`s.count(<sub>)` returns the number of non-overlapping occurrences of substring `<sub>` in `s`:

`s.find(<sub>[, <start>[, <end>]])`

> Searches the target string for a given substring.

You can use `.find()` to see if a Python string contains a particular substring. `s.find(<sub>)` returns the lowest index in `s` where substring `<sub>` is found:

Python >>>

```
>>> 'foo bar foo baz foo qux'.find('foo')
0
```

This method returns –1 if the specified substring is not found:

Python >>>

```
>>> 'foo bar foo baz foo qux'.find('grault')
-1
```

The search is restricted to the substring indicated by `<start>` and `<end>`, if they are specified:

Python >>>

```
>>> 'foo bar foo baz foo qux'.find('foo', 4)
8
>>> 'foo bar foo baz foo qux'.find('foo', 4, 7)
-1
```

Ejercicio 02 (3 puntos)

Se quiere diseñar un programa en. Python que imprima por pantalla todas las fichas de un juego de dominó.

El dominó es un juego de 28 fichas, cada una de las cuales tiene dos valores. Los valores que puede tener son del blanco (0) hasta el 6.

El formato para imprimir las fichas será el siguiente

[nro , nro]

Fila 1: Las fichas con el 0 en orden
Fila 2: Las fichas con el 1 en orden (sin repetir las ya impresas)
Fila 3: Las fichas con el 2 en orden (sin repetir las ya impresas)
Fila 4: Las fichas con el 3 en orden (sin repetir las ya impresas)
Fila 5: Las fichas con el 4 en orden (sin repetir las ya impresas)
Fila 6: Las fichas con el 5 en orden (sin repetir las ya impresas)
Fila 7: Las fichas con el 6 en orden (sin repetir las ya impresas)

[nro , nro] [nro , nro] [nro , nro] [nro , nro]
[nro , nro] [nro , nro] [nro , nro]

En la impresión se debe colocar una ficha debajo de la otra, la separación entre las fichas es un espacio en blanco.

Con las herramientas que tenemos hasta ahora la mejor solución es poner todos los valores posibles que puede tomar las fichas dentro de un string, de forma análoga como se hizo con el abecedario en el ejercicio del código de César.

Ejercicio 03 (3 puntos)

Escribir un programa en Python que pida por teclado un número n e imprima los primeros n números triangulares, junto con su índice. El programa debe verificar la entrada de datos, es decir se debe comprobar que lo que introduzca el usuario es un número y que este es mayor que 0.

Los números triangulares se obtienen mediante la suma de los números naturales desde 1 hasta n.

Es decir, si se piden los primeros 5 números triangulares, el programa debe imprimir:

1 - 1
2 - 3
3 - 6
4 - 10
5 - 15

Forma de entrega: 3 ficheros .py con el número de ejercicio subidos a la plataforma campus.

Fecha tope de entrega: Sábado 22 de octubre de 2022 hasta las 23:59 horas.