IES SAN JUAN DE LA RAMBLA
DPTO. INFORMÁTICA Y COMUNICACIONES
CFGS DESARROLLO DE APLICACIONES MULTIPLATAFORMA
PROGRAMACIÓN 1º

San Juan de la Rambla a 16 de diciembre de 2022

Unidad 03 - Funciones — Prueba de desarrollo

Esta actividad es la parte de desarrollo de la prueba de esta unidad. Su valor es de 8 puntos. Recuerden que la parte de conocimientos vale 2 puntos.

El peso de esta actividad en el total de la unidad es del 60%.

Este actividad consiste en la dar solución a tres problemas que se deben resolver mediante un algoritmo escrito en Python.

La forma de entrega es mediante 3 ficheros con extensión .py y con nombre “ejercicio0n.py”, cambiando la “n” por el número 1, 2 o 3 según sea el caso

La entrega se realiza a través de la plataforma campus. La prueba comenzará a las 9:50 y terminará a las 12:10 que se cerrará la plataforma.

Mucha suerte!!!!

Ejercicio 01 (2,0 puntos)

EN NINGUN CASO SE ACEPTARÁ FUNCIONES QUE TRABAJEN CON STRING,

LA FUNCIÓN SOLAMENTE DEBE TRABAJAR CON ENTEROS.

Un número entero es capicúa, cuando tiene el mismo valor numérico de derecha a izquierda que de izquierda a derecha.

Por ejemplo, el número 1234321 si lo invertimos da 1234321 que tiene el mismo valor numérico, o sea, que son iguales.

Se pide una función en Python que recibe como parámetro un número y retorna True si el número es capicúa y False en caso contrario.

Help:

Una forma de plantear de forma recursiva este ejercicio es la siguiente:

El número 1234321 podría dividirse de la siguiente forma

1	23432	1
primer_digito	centro	último dígito

Entonces el número será capicúa, si el primer dígito coincide con el último y la zona central es capicúa.

El caso base será.....cuando el número tiene un sólo dígito.

Ejercicio 02 (3,0 puntos)

El servicio de inteligencia de Pythonia ha estado monitoreando los mensajes de texto que recibe el líder del país de Javapolis, y ha detectado un patrón en éstos, pero no ha podido descifrar qué dicen los mensajes, por lo que han acudido a ti para que los apoye en esta importante misión.

Ellos detectaron que el líder recibe muchas palabras, pero sólo las que contienen una clave en alguna parte, son consideradas en el mensaje. Además, notaron que cuando quieren dar por finalizado un mensaje, usan la palabra out. Notar que en Javapolis, sólo trabajan con letras minúsculas.

El servicio de inteligencia logró crear la función obtenerPosicion (palabra, clave), la cual devuelve la posición en la cual comienza la clave dentro de la palabra y en caso contrario retorna -1. Notar que las posiciones comienzan desde 0.

A continuación, se presentan ejemplos de cómo funciona la función obtenerPosicion().

```
>>>obtenerPosicion('foniwida', 'iwi')      >>>obtenerPosicion('primavera', 'iwi')
> 3                                        > -1
```

En base a lo descrito anteriormente se le pide:

1. (1,0 punto) Escribir la función **obtenerPalabra(palabra, clave)** con las siguientes características:

- La función recibe dos parámetros
    1. 'palabra' es la palabra que se ha interceptado y que puede contener o no la clave en su interior
    2. 'clave', es la palabra clave que utiliza Javapolis para encriptar los mensajes.
- La función puede retornar:
    1. Si dentro del parámetro palabra está la clave se debe retornar la palabra sin la clave.
    2. Si dentro del parámetro palabra NO está la clave se debe retornar -1.
- **La función debe utilizar a la función obtenerPosicion()**
- Se debe asumir que la clave sólo va a estar como máximo 1 vez dentro de palabra.

2. (2,0 puntos) Escribir una función de nombre **descifraMensaje(mensaje, clave)**, donde mensaje es el mensaje a desencriptar y clave la clave que se ha usado para encriptarlo y retorna un string con el mensaje desencriptado.

La función debe usar la función obtenerPalabra().

Como ejemplo el mensaje es:

i wana be feliiwices programar public python iwilos class primavera cuatroiwi cueca i win .iwi out

El mensaje desencriptado sería:

felices los cuatro.

Ejercicio 03 (3,0 puntos)

Una empresa para guardar los datos de su personal mantiene el siguiente diccionario.

personal = { clave_1:valor_1, clave_2:valor_2, ... , clave_n:valor_n }

Donde:
clave: es el dni de la persona
valor: [nombre, apellido, edad, sexo] una lista de valores donde:

nombre, apellido y sexo son string y edad es un número

Se piden las siguientes funciones:

1. (1,0 punto) mayoresEdad(diccionario, edad)

Esta función retorna una lista de tuplas, donde en cada tupla se almacena el nombre y apellido de las personas del diccionario que sean mayores que la **edad** que se ha pasado como parámetro.

2. (1,0 punto) contarEmpleados(diccionario)

Esta función cuenta el número total de personas, el número de mujeres y el de hombres del diccionario y lo retorna como una tupla de la siguiente forma:

(nro_mujeres, nro_hombres, total_personas)

3. (1,0 punto) dividirEmpleados(diccionario)

Que crea dos diccionarios a partir del diccionario pasado como parámetro.
Uno de ellos guarda los datos de los trabajadores y el otro guarda los datos de las trabajadoras.

Los diccionarios son retornados de la siguiente forma:

(dicc_mujeres, dicc_hombres)