Programación para *Data Science*

Unidad 3: Conceptos avanzados de Python

Ejercicios y preguntas teóricas

A continuación, encontraréis la parte que tenéis que completar en este modulo y las preguntas teóricas a contestar.

Ejercicio 1

Completad el código necesario para calcular el número de palabras y de espacios de un texto. El número de espacios no tiene que ser necesariamente función del número de palabras. **(1 punto)**

In [1]:
```
def contar_palabras_y_espacios(texto):
    # Cuenta las palabras contenidas en el string texto y también sus espacios.
    num_palabras = 0
    num_espacios = 0

    # Código que hay que completar.

    return num_palabras, num_espacios

texto = "Orbiting Earth in the spaceship, I saw how beautiful our planet is. \
            People, let us preserve and increase this beauty, not destroy it!"

num_palabras, num_espacios = contar_palabras_y_espacios(texto)
print "El número de palabras es de %d" % num_palabras
print "El número de espacios es de %d" % num_espacios
```

```
El número de palabras es de 0
El número de espacios es de 0
```

Ejercicio 2

Dada una molécula representada por un *string* del estilo C9-H8-O4 calculad su masa atómica. Por ejemplo, para una molécula C4-H3, la masa atómica sería de 4*12.01 + 3*1.007825.

Haced una solución general accediendo al diccionario mediante la clave, que en este caso será el tipo de átomo. Por ejemplo, para la molécula C5-H3 deberíamos seguir estos pasos:

- Separar la molecula por los guiones (podemos hacerlo con la función split, por ejemplo).
- Para cada una de las partes, C5 y H3, encontrar el tipo de átomo: C y H (necesitaremos un bucle de algún tipo aquí).
- Acceder al diccionario de masas y para la clave que se corresponde con el tipo de átomo, obtener la masa.
- Una vez encontrada la masa, multiplicarla por el número de átomos encontrados.

Pista: para un string del tipo a = 'C15', a[0] nos proporcionará el tipo de átomo, C. a[1:] nos proporciona el string restante: '15'. Tened en cuenta que ha de convertirse a número decimal para poder multiplicarse. **(1.5 puntos)**

In [2]:
```
# Masas atómicas
masas = {'H': 1.007825, 'C': 12.01, 'O': 15.9994, 'N': 14.0067, 'S': 31.972071, 'P': 30.973762}

def calcula_masa_atomica(molecula):
    """
    Calcula la masa atómica de una molécula
    """
    masa = 0.0

    # Código que hay que completar.

    return masa

print calcula_masa_atomica('C13-H18-O2')
print calcula_masa_atomica('C8-H10-N4-O2')
print calcula_masa_atomica('C20-H25-N3-O')
print calcula_masa_atomica('C20-H10-O2-P2-S')
```

```
0.0
0.0
0.0
0.0
```

Ejercicio 3

Completad las siguientes funciones matemáticas y documentad el código también de cada función. Por último, escribid algún ejemplo de uso de cada una de las funciones. **(1.5 puntos)**

In [ ]:
```
# Completad las siguientes funciones matemáticas
import math

def area_circulo(radio):
    # Código que hay que completar.
    return 0.


def longitud_circulo(radio):
    # Código que hay que completar.
    return 0.


def hipotenusa(cateto1, cateto2):
    # Código que hay que completar.
    return 0.


def area_cuadrado(lado):
    # Código que hay que completar.
    return 0.


def volumen_esfera(radio):
    # Código que hay que completar.
    return 0.


def volumen_cubo(lado):
    # Código que hay que completar.
    return 0.


# Escribid aquí algunos ejemplos utilizando estas funciones, por ejemplo:
# print 'El volumen del cubo de lado 3.5 es %f' % volumen_cubo(3.5)
```

Ejercicio 4

El siguiente ejercicio consiste en pasar un número en base 2 (binario, 0/1) a base 10 (decimal).

Dado un *string* que representa un número en binario, por ejemplo, 1011, devolved el número natural correspondiente, en este caso, 11. **(1.5 puntos)**

In [1]:
```
# Código que hay que completar: DEFINID UNA FUNCIÓN y escribid algunos casos de uso de esa función.
```

Ejercicio 5

Uno de los algoritmos más básicos en criptografía es el cifrado César (https://es.wikipedia.org/wiki/Cifrado_C%C3%A9sar), que fue utilizado por Julio César para comunicarse con sus generales, y que consiste en dado un texto, por cada una de las letras del texto, añadirle un desplazamiento para conseguir una nueva letra diferente de la original. Comprenderemos rápidamente su mecanismo mediante un ejemplo:

Si asignamos el número 1 a la primera letra del abecedario, A, 2 a la siguiente, B, etc., imaginad que tenemos el siguiente mensaje:
ABC
123

Si aplicamos un desplazamiento de 3, buscaremos cuál es la letra en el abecedario que se corresponde:
DEF
456

ABC se ha convertido en DEF porque hemos sumado un desplazamiento de 3. También podríamos aplicar otros tipos de desplazamiento como los negativos. Por ejemplo, para el desplazamiento -1 y el mensaje original ABC tendríamos un mensaje cifrado de: ZAB.

Escribid una función que dado un mensaje original y un desplazamiento, calcule y devuelva el mensaje cifrado. **(1.5 puntos)**

In [3]:
```
def cifrado_cesar(mensaje, desplazamiento=1):
    """
    Cifra el mensaje utilizando el metodo de Cesar dado un desplazamiento
    """
    mensaje_cifrado = ""

    # Código que hay que completar.

    return mensaje_cifrado

# Aquí podéis añadir mas ejemplos.
print cifrado_cesar("PROGRAMACION PARA DATA SCIENCE", 1)
```

Pregunta 1

Las funciones *range* y *xrange* pueden utilizarse con la misma finalidad, pero su funcionamiento es distinto. Explicad en qué se diferencian y comentad para qué sirve la palabra reservada *yield*. Poned un ejemplo de uso de *range*, *xrange* y otro de *yield*. **(1 punto)**

Respuesta:

Respuesta

Pregunta 2

El paradigma de programación orientada a objetos es ampliamente utilizado en gran parte de las librerías que se escriben en Python. Es una forma útil de encapsular información de la que se ocupará el propio objeto donde se ha definido esa información. Explicad qué es una clase, un objeto, un atributo, un método, un constructor, una superclase y una subclase. Poned un ejemplo de definición de una clase en código Python y un ejemplo de uso de esa misma clase. Podéis basaros en el material Object Oriented Program Design (http://life.bsc.es/pid/brian/python/#/7). **(1 punto)**

Respuesta:

Respuesta

Pregunta 3

Las excepciones son errores detectados en tiempo de ejecución. Pueden y deben ser manejadas por el programador para minimizar el riesgo de que un determinado programa falle de forma no controlada. Poned ejemplos de tipos de excepción en el lenguaje Python y de cómo se capturan. **(1 punto)**

Respuesta:

Respuesta