prog_datasci_2_python_entrega

October 2, 2022

1 Programación para *Data Science*

1.1 PEC 2 - Introducción a Python

En este Notebook encontraréis el conjunto de actividades evaluables como PEC de la asignatura. Veréis que cada una de ellas tiene asociada una puntuación, que indica el peso que tiene la actividad sobre la nota final de la PEC. Adicionalmente, hay un ejercicio opcional, que no tiene puntuación dentro de la PEC, pero que se valora al final del semestre de cara a conceder las matrículas de honor y redondear las notas finales. Podréis sacar la máxima nota de la PEC sin necesidad de hacer este ejercicio. El objetivo de este ejercicio es que sirva como pequeño reto para los estudiantes que quieran profundizar en el contenido de la asignatura.

Veréis que todas las actividades de la PEC tienen una etiqueta, que indica los recursos necesarios para llevarla a cabo. Hay tres posibles etiquetas:

- NM **Sólo materiales**: las herramientas necesarias para realizar la actividad se pueden encontrar en los materiales de la asignatura.
- EG **Consulta externa guiada**: la actividad puede requerir hacer uso de herramientas que no se encuentran en los materiales de la asignatura, pero el enunciado contiene indicaciones de dónde o cómo encontrar la información adicional necesaria para resolver la actividad.
- EI **Consulta externa independiente**: la actividad puede requerir hacer uso de herramientas que no se encuentran en los materiales de la asignatura, y el enunciado puede no incluir la descripción de dónde o cómo encontrar esta información adicional. Será necesario que el estudiante busque esta información utilizando los recursos que se han explicado en la asignatura.

Es importante notar que estas etiquetas no indican el nivel de dificultad del ejercicio, sino únicamente la necesidad de consulta de documentación externa para su resolución. Además, recordad que las **etiquetas son informativas**, pero podréis consultar referencias externas en cualquier momento (aunque no se indique explícitamente) o puede ser que podáis hacer una actividad sin consultar ningún tipo de documentación. Por ejemplo, para resolver una actividad que sólo requiera los materiales de la asignatura, podéis consultar referencias externas si queréis, ya sea tanto para ayudaros en la resolución como para ampliar el conocimiento!

En cuanto a la consulta de documentación externa en la resolución de los ejercicios, recordad **citar siempre la bibliografía utilizada** para resolver cada actividad.

1.2 Ejercicios y preguntas teóricas.

1.2.1 Pregunta 1

En Python, hay un conjunto de palabras que no se pueden usar para declarar variables, puesto que tienen un significado especial y su nombre está reservado para ciertas tareas internas del lenguaje de programación. Estas palabras se conocen como **palabras reservadas**. ¿Cuáles de las siguientes palabras corresponden a palabras reservadas? Haz una breve descripción de las palabras que lo sean. **(0.5 puntos)** NM

a) while

b) return

c) inout

d) asis

e) False

f) beta

[1]:
```
# Respuesta
```

1.2.2 Ejercicio 1

Proporcionad un ejemplo para cada uno de los tipos de variables que se enumeran a continuación:

- a) Número entero diferente a 0
- b) Número decimal con dos decimales
- c) Cadena de caracteres con una longitud mínima de 5 caracteres
- d) Listado de 4 nombres de persona

A continuación, realizad las operaciones `/` y `//` entre el número definido en el apartado `a` y el número definido en el apartado `b`. Asignad cada resultado a una variable y mostradlas por pantalla. ¿Se obtienen resultados diferentes? ¿Por qué?. **(1 punto)** NM

[2]:
```
# Respuesta
```

1.2.3 Ejercicio 2

Definid la función `resta(x,y)` donde `x` y `y` son dos números enteros diferentes de 0. Esta función tiene que mostrar por pantalla el **valor absoluto** de la diferencia entre los valores `x` y `y`. **(1 punto)**

A continuación encontraréis un conjunto de ejemplos de respuestas esperadas (test cases). La primera columna contiene los valores de `x` y la segunda columna contiene los valores de `y`. Ejecutad y mostrad cada uno de los ejemplos por pantalla.**(0.5 puntos)**

x	y	Resultado
3	2	1
4	6	2
10	-10	20

Nota: Para resolver este ejercicio tenéis que utilizar la función `abs()` que podéis consultar en el seguiente link. **(1.5 puntos)** EG

[3]:
```
# Respuesta
```

1.2.4 Ejercicio 3

Cread el string **roxanneENNAXORanneRox** en una única expresión y utilizando solamente la variable m = `roxanne`.

Nota: Para resolver este ejercicio tenéis que utilizar las funciones para convertir un string en mayúsculas que podéis consultar en el seguiente link. **(1 punto)** EG

[1]:
```
# Respuesta
```

1.2.5 Ejercicio 4

Considerando las tres variables siguientes `z = 3`, `r = 9` y `x = 7`, escribid las seguientes operaciones lógicas y mostrad el resultado por pantalla.

1. `z es más pequeño o igual a x y r es diferente a x.`
2. `r es igual a z.`
3. `x es más pequeño que z o r es más grande o igual a x.`

(1 punto) NM

[5]:
```
# Respuesta
```

1.2.6 Ejercicio 5

Cread una lista de 5 elementos formada por números enteros diferentes. Mediante las funciones de la llibrería de Python `Numpy`, cread un código para responder cada una de las siguientes preguntas y mostrad el resultado por pantalla.

a) Calculad la suma de todos los elementos de la lista.**(0.5 puntos)**

b) Calculad el máximo y el mínimo de la lista.**(0.5 puntos)**

c) Calculad la desviación estándard con dos números decimales.**(0.5 puntos)**

d) Ordenad la lista de mayor a menor.**Opcional**

(1.5 punts) NM

[ ]:
```
# Respuesta
```

1.2.7 Ejercicio 6

A partir del código que se muestra en la celda seguiente:

a) Añadir comentarios para entender mejor cada paso. **(0.5 puntos)**

b) Añadir el elemento "FutureIslands" en la lista original. **(0.5 puntos)**

c) Escrivid un programa que a partir de la lista de los grupos de música muestre por pantalla todos los grupos de música separados entre ellos por un guión. **Opcional**

(1 punt) NM

[2]:
```
music = ["ArcadeFire","TheFoals","REM","ColdPlay","U2","TheNational"]

music_sorted=sorted(music, key=len)

print(music_sorted)
```

```
['U2', 'REM', 'TheFoals', 'ColdPlay', 'ArcadeFire', 'TheNational']
```

[ ]:
```
# Respuesta
```

1.2.8 Ejercicio 7

Queremos guardar los títulos de libros que os habéis leído con la valoración numérica del 0 al 10 para saber si os ha gustado poco o mucho. Para hacer esto, crearemos un **diccionario** donde guardaremos el título del libro, donde sustituiremos los espacios por _, y la valoración númerica con un decimal.

a) ¿Qué variable utilizaríais como llave para crear un diccionario y cuál como valor si queremos acceder a la información del diccionario mediante el títol del libro? Crea un diccionario con 5 elementos y muestralos por pantalla. **(0.5 puntos)**

b) Muestra solamente las llaves existentes. **(0.25 puntos)**

c) Ordena el diccionario según valoración numérica de mayor a menor **(0.25 puntos)**

(1 punto) NM

[ ]:
```
# Respuesta
```

1.2.9 Ejercicio 8

Cread un fragmento de código que muestre per pantalla `True` si la longitud de un determinado string es más grande a 10 o `False` si la longitud es más pequeña o igual a 10. El código tiene que permetir introducir un string por pantalla. **(1 punto)** EG

[ ]:
```
# Respuesta
```

1.2.10 Ejercicio 9

El código que se presenta a continuación no funciona. Corregid el error y explicad por qué sucede. **(0.5 puntos)** NM

[3]:
```
radius = 1700

satellite = "Moon"

print("The %d is Earth's only natural satellite. Its radius is approximatly %d␣
 ↪km" % (satellite, radius))
```

```
      ␣
↪---------------------------------------------------------------------------

       TypeError                                 Traceback (most recent call last)

       <ipython-input-3-f46dfc3f10c9> in <module>
         3 satellite = "Moon"
         4
   ----> 5 print("The %d is Earth's only natural satellite. Its radius is␣
↪approximatly %d km" % (satellite, radius))


       TypeError: %d format: a number is required, not str
```