Scripting in Python

Oscar Ortega Real 1st DAM

Índice

Exercici 1

Prepara amb un exemple on expliques com fer en Python 3:

- Clonar una llista.
    - ¿Quina és la diferència en Python entre "shallow copy" i "deep copy"?

- Afegir un element a una llista.
- Llevar un element a una llista.
- Crear una nova llista amb els 4 últims elements d'una llista.
- Convertir les paraules d'una cadena (separades per espai) a una llista.
- Comentaris amb una línia.
- Comentaris multilínia.

```
#Creo una lista
original_lista = [1, 2, 3, 4, 5]

#======== Formas de clonar una lista  ========
#   list()
nueva_lista = list(original_lista)
#    :
nueva_lista = original_lista[:]
#   copy()
nueva_lista = original_lista.copy()


# == MÓDULO COPY ==
import copy
#   copy()
nueva_lista = copy.copy(original_lista)
#   deepcopy()
# - Copia en profundidad de la estructura cuando tienes listas
anidadas. -
anidadas_lista = [[1, 2, 3],[4, 5]]
nueva_lista = copy.deepcopy(anidadas_lista)

#   [-x:] -> X últimos elementos de una lista
ultimos4_lista = original_lista[-4:]
```

```python
#========  Añadir un elemento a la lista  ========
#   append() -> Elemento al final de la lista
original_lista.append(4)
#   insert() -> Pide la posición donde guardar el elemento
original_lista.insert(1, 4)
#   extend() -> Agrega elementos de otra lista al final

#========  Borrar un elemento a la lista  ========
#   remove(x) -> Elimina el valor igual a x
original_lista.remove(3)
#   del example_list[x] -> Elimina la posición x
del original_lista[2]
#   pop() -> Almacena el elemento que elimina de la lista
elemento_eliminado = original_lista.pop(2)
ultimo_eliminado = original_lista.pop()
#   clear() -> Deja vacía la lista
original_lista.clear()



#========  Guardar cadenas de texto en listas ========
#   split() -> "X" : busca X separador / Vacío : espacios en
blanco.
cadena = "Esta es una cadena de ejemplo"
lista_resultante = cadena.split()
print(lista_resultante)
```

Resultado

```
oscarortega@lliurex-client19:~$ /usr/bin/python3 /home/oscarortega/Escritorio/CLASE/GestionEmp/actEval1.py
['Esta', 'es', 'una', 'cadena', 'de', 'ejemplo']
```

Exercici 2

Crea un exemple amb 3 funcions que:

● Reva 2 números i torne la suma.

● Reva una llista i modifique eixa mateixa llista (referència) doblant els valors de tots els elements. No ha de retornar res.

● Reva una llista i torne una còpia de la llista mateixa llista (referència) doblant els valors de

tots els elements. La llista original no hi ha de modificar-se.

```python
# Pide dos números los suma y devuelve el resultado =====
def suma(a, b):
    resultado = a + b
    return resultado

# - Uso de la función -
resultado_suma = suma(3, 4)
print(resultado_suma)

# Recibe una lista y duplica todos los valores =====
def multiTodo(lista):
    listaMulti = [elemento * 2 for elemento in lista]

# Devuelve una lista multiplicada sin modificar la original =====
def addListMulti(lista):
    import copy
    listaMulti = [elemento * 2 for elemento in lista]
    listaNueva = copy.copy(listaMulti)
    return listaNueva

# - Uso de las funciones -
list = [1,2,3]
multiTodo(list)
listaCopiada = addListMulti(list)
print(list, listaCopiada)
```

Resultado

```
oscarortega@lliurex-client19:~$ /usr/bin/python3
7
[1, 2, 3] [2, 4, 6]
```

Exercici 3

Volem emmagatzemar un usuari i la seua contrasenya. Fes un exemple que explica com es faria:

- Utilitzant una llista.
- Utilitzant un diccionari.

En omplir-se, les contrasenyes han de passar-se a un format Hash.

L'exemple ha d'omplir la llista amb 5 usuaris/contrasenya i fer dues consultes.

```
# Almacenar usuarios y contraseñas con listas
usuarios_lista = ["usuario1", "usuario2", "usuario3", "usuario4",
"usuario5"]
contraseñas_lista = ["pass1", "pass2", "pass3", "pass4", "pass5"]

# Consulta 1: Verificar si un usuario y contraseña existen en las
listas
usuario_input = input("Ingrese el nombre de usuario: ")
contraseña_input = input("Ingrese la contraseña: ")

if usuario_input in usuarios_lista and contraseña_input in
contraseñas_lista:
    print("Acceso concedido")
else:
    print("Acceso denegado")

# Consulta 2: Mostrar todos los usuarios y contraseñas
almacenados
print("Usuarios y contraseñas almacenados:")
for usuario, contraseña in zip(usuarios_lista,
contraseñas_lista):
    print(f"Usuario: {usuario}, Contraseña: {contraseña}")
```

```python
# Almacenar usuarios y contraseñas con diccionario
usuarios_diccionario = {
    "usuario1": "pass1",
    "usuario2": "pass2",
    "usuario3": "pass3",
    "usuario4": "pass4",
    "usuario5": "pass5"
}

# Consulta 1: Verificar si un usuario y contraseña existen en el diccionario
usuario_input = input("Ingrese el nombre de usuario: ")
contraseña_input = input("Ingrese la contraseña: ")

if usuario_input in usuarios_diccionario and usuarios_diccionario[usuario_input] == contraseña_input:
    print("Acceso concedido")
else:
    print("Acceso denegado")

# Consulta 2: Mostrar todos los usuarios y contraseñas almacenados
print("Usuarios y contraseñas almacenados:")
for usuario, contraseña in usuarios_diccionario.items():
    print(f"Usuario: {usuario}, Contraseña: {contraseña}")
```

Resultado

```
Ingrese el nombre de usuario: usuario1
Ingrese la contraseña: pass1
Acceso concedido
Usuarios y contraseñas almacenados:
Usuario: usuario1, Contraseña: pass1
Usuario: usuario2, Contraseña: pass2
Usuario: usuario3, Contraseña: pass3
Usuario: usuario4, Contraseña: pass4
Usuario: usuario5, Contraseña: pass5
```

Exercici 4

Explica amb exemples com funcionen els operadors “is”, “not”, “in”

```python
# Operador "is" para verificar si dos variables son el mismo objeto
a = [1, 2, 3]
b = a
c = 4
print(a is b)  # True, ya que b apunta a la misma lista en memoria
que a
print(c is b)  # False

# Operador "not" para negar una condición o una expresión booleana
condicion = True

if not condicion:
   print("La condición es falsa.")
else:
   print("La condición es verdadera.")

# Operador "in" para verificar si un elemento está presente en una
secuencia
lista = [1, 2, 3, 4, 5]

if 3 in lista:
   print("El elemento 3 está en la lista.")
else:
   print("El elemento 3 no está en la lista.")
```

Resultado

```
True
False
La condición es verdadera.
El elemento 3 está en la lista.

Process finished with exit code 0
```

Exercici 5

· Posa un exemple de com passar diversos paràmetres desde consola a un programa
· Posa un exemple de com fer "sobrecàrrega de funcions" (funcions que poden rebre diversos números de paràmetres), incloent el cas de què el nombre de paràmetres no siga definit.

```python
# Solicitar al usuario que ingrese información
nombre = input("Ingrese su nombre: ")
# Solicitar al usuario que ingrese un numero
edad = int(input("Ingrese su edad: "))

# Imprimir la información ingresada
print("Nombre:", nombre)
print("Edad:", edad)
```

```python
def saludar(nombre, saludo="Hola"):
    """
    Función que saluda a una persona con un saludo opcional.

    Parameters:
    nombre (str): El nombre de la persona.
    saludo (str): Saludo opcional. Por defecto es "Hola".
    """
    print(f"{saludo}, {nombre}!")


# Ejemplos de uso
saludar("Juan")
saludar("Maria", "Hi")
```

Resultado

```
/usr/bin/python3.6 /home/oscar
Ingrese su nombre: Pedro
Ingrese su edad: 24
Nombre: Pedro
Edad: 24
```

Exercici 6

Crea un llistat en el qual cada element d'eixa llista siga una llista amb dos valors: mida i pes.
Utilitzant sorted i las "key functions", fer que aquesta llista s'ordene per major altura i en cas d'igualtat, per menor pes.

```
# Crear una lista de listas con medida y peso
lista_datos = [
    [1.75, 68],
    [1.60, 55],
    [1.80, 75],
    [1.65, 60],
]
# Imprimir la lista de datos
print("Lista de datos:")
for datos in lista_datos:
    print(f"Medida: {datos[0]} m, Peso: {datos[1]} kg")
```

```
# Ordenar la lista por mayor altura y, en caso de igualdad, por menor
peso
# key function -> Usadas para proporcionar una función que devuelve
un valor clave y con este ordenar elementos en lista

lista_ordenada = sorted(lista_datos, key=lambda x: (x[0], -x[1]))

# Imprimir la lista ordenada
print("Lista ordenada:")
for datos in lista_ordenada:
    print(f"Medida: {datos[0]} m, Peso: {datos[1]} kg")
```

Resultado

```
Lista de datos:
Medida: 1.75 m, Peso: 68 kg
Medida: 1.6 m, Peso: 55 kg
Medida: 1.8 m, Peso: 75 kg
Medida: 1.65 m, Peso: 60 kg
Lista ordenada:
Medida: 1.6 m, Peso: 55 kg
Medida: 1.65 m, Peso: 60 kg
Medida: 1.75 m, Peso: 68 kg
Medida: 1.8 m, Peso: 75 kg

Process finished with exit code 0
```

Exercici 7

Defineix la classe Car en Python 3. La classe tindrà com atributs “matrícula” (numèrica) i “color”.
Crea un mètode imprimir, i a més dos mètodes que vulgues.
En segon lloc, fes que el programa demane un número “n” per teclat i es creen “n” instàncies de la classe, on cada instància:

- Cada “matricula” tindrà un número consecutiu des d’1 fins a “n”.
- El “color” serà per a cada instància un color aleatori obtingut d’aquest llistat [“red”, “white”, “black”, “pink”, “blue”]

Finalment, el programa haurà d’imprimir els valors de les 10 primeres instàncies. En cas que “n” siga menor que 10, només imprimirà “n” instàncies.

```
import random
class Car:
    def __init__(self, matricula, color):
        """
        Constructor de la clase Car.

        Parameters:
        matricula (int): Número de matrícula del coche.
        color (str): Color del coche.
        """
        self.matricula = matricula
        self.color = color
        self.encendido = False

    def imprimir(self):
        """Método que imprime información sobre el coche."""
        estado_motor = "encendido" if self.encendido else
"apagado"
        print(f"Coche: Matrícula {self.matricula}, Color
{self.color}, Estado del motor: {estado_motor}")









    def encender(self):
```

```
        """Método para encender el motor del coche."""
        if not self.encendido:
            print("Motor encendido.")
            self.encendido = True
        else:
            print("El motor ya está encendido.")

    def apagar(self):
        """Método para apagar el motor del coche."""
        if self.encendido:
            print("Motor apagado.")
            self.encendido = False
        else:
            print("El motor ya está apagado.")

# Solicitar al usuario un número "n"
n = int(input("Ingrese un número entero para crear instancias de
la clase Car: "))

# Lista de colores posibles
colores_posibles = ["red", "white", "black", "pink", "blue"]

# Crear "n" instancias de la clase Car
coches = [Car(matricula=i+1,
color=random.choice(colores_posibles)) for i in range(n)]

# Imprimir información sobre las 10 primeras instancias o las "n"
instancias si son menos de 10
for coche in coches[:10]:
    coche.imprimir()
```

Resultado

```
Ingrese un número entero para crear instancias de la clase Car: 50
Coche: Matrícula 1, Color black, Estado del motor: apagado
Coche: Matrícula 2, Color white, Estado del motor: apagado
Coche: Matrícula 3, Color white, Estado del motor: apagado
Coche: Matrícula 4, Color pink, Estado del motor: apagado
Coche: Matrícula 5, Color pink, Estado del motor: apagado
Coche: Matrícula 6, Color white, Estado del motor: apagado
Coche: Matrícula 7, Color blue, Estado del motor: apagado
Coche: Matrícula 8, Color red, Estado del motor: apagado
Coche: Matrícula 9, Color pink, Estado del motor: apagado
Coche: Matrícula 10, Color blue, Estado del motor: apagado

Process finished with exit code 0
```