Codificación y Programación en Python Capitulo 2 (Quices)

Carlos Daniel García Chaparro.

Unidad 10

Quiz 1

Q1.

Cree una primos_lista que tenga como elementos números primos entre 2~10.

A continuación, utilice la indexación de la lista al primer elemento de la lista e imprima como se muestra a continuación

Condicion para la ejecución	1er elemento de primos_lista 2. 3.
Tiempo	5 Minutos

```
In [1]: primos_lista = [2, 3, 5, 7]
        print(primos_lista[0])
```

```
2
```

Quiz 3

Q3.

Hay una lista con la cadena s_list = ['abc', 'bcd', 'bcdefg', 'abba', 'cddc', 'opq']. Implementa la siguiente función a esta lista.

Del problema de programación por pares anterior, la longitud de 'abc', 'bcd', 'opq' es igual a 3.

Asimismo, si las longitudes de las cadenas son iguales, escribe un programa que imprima las tres cadenas más cortas como sigue.

Utiliza la función sort(key=len) para ordenar las cadenas por su longitud y luego escribe un código.

***Ejemplo de salida:**

Las cadenas más cortas son:

'abc', 'bcd', 'opq'

```
cadenas_list = ['abc', 'bcd', 'bcdefg', 'abba', 'cddc', 'opq']

cadenas_list = sorted(cadenas_list, key=len)

palabras = ""
lista = []

for pal in cadenas_list:
    if palabras == "" or len(pal) <= len(palabras):
        palabras = pal
        lista.append(palabras)

print(lista)
```

```
['abc', 'bcd', 'opq']
```

Unidad 11

Quiz 1

```
frutas_dic = {
    "manzana" : 6000,
    "melon" : 3000,
    "banano" : 5000,
    "naranja" : 4000,
}
print(frutas_dic.keys())

if "manzana" in frutas_dic:
    print("manzana está en frutas_dic")
if "mango" not in frutas_dic:
    print("mango no está en frutas_dic")
```

```
dict_keys(['manzana', 'melon', 'banano', 'naranja'])
manzana está en frutas_dic
mango no está en frutas_dic
```

Unidad 12

Codificación en papel Quiz 2:

Q2.

La siguiente tupla registra las ventas diarias de una tienda durante 10 días. Escribe un código para imprimir cuántos días se redujeron las ventas en comparación con el día anterior. (Sugerencia: compara los valores iterando los elementos con la sentencia de iteración).

Condicion para la ejecución	Registro diario de ventas: (100, 121, 120, 130, 140, 120, 122, 123, 190, En los últimos 10 días, 3 días han reducido las ventas en comparación con el día anterior.
Tiempo	10 Minutos

```
reg_ventas= (100, 121, 120, 130, 140, 120, 122, 123, 190, 3)
contador = 0

for venta in range(1, len(reg_ventas)):
    if reg_ventas[venta] < reg_ventas[venta - 1]:
        contador += 1

print(f"En los últimos 11 días, {contador} días han reducido las ventas en comparación con el día anterior.")
```

```
En los últimos 11 días, 3 días han reducido las ventas en comparación con el día anterior.
```

Programación por parejas

Quiz 1

Q1.

Devuelve el elemento con el máximo número de ocurrencias. Cuando hay más de dos elementos frecuentes, imprime el número más alto.

Ejemplo de salida:

Tupla obtenida: (1, 2, 5, 4, 3, 2, 1, 9, 9, 3, 7, 3, 9)

El elemento mas frecuente: 9

```
tupla = (1, 2, 5, 4, 3, 2, 1, 9, 9, 3, 7, 3, 9,)
elemento_frecuente = None
for e in tupla:
    if (elemento_frecuente == None) or (e > elemento_frecuente):
        elemento_frecuente = e
print("El elemento mas frecuente:", elemento_frecuente)
```

```
El elemento mas frecuente: 9
```

Quiz 2

Q2.

En el ejemplo de salida de abajo, hay tuplas que contienen elementos, así como tuplas vacías, cadenas vacías y listas vacías que no tienen elementos. Escribe un código para eliminar estas tuplas vacías, cadenas vacías y listas vacías de la lista dada a continuación. (Sin embargo, no elimine la tupla (,) porque se considera que tiene una tupla vacía).

Ejemplo de salida:

Tupla obtenida: [(), (1,), [], 'abc', (), (), (1,), ('a',), ('a', 'b'), ((),), '']

El elemento mas frecuente: [(1,), 'abc', (1,), ('a',), ('a', 'b'), ((),)]

```
t = [(), (1,), [], 'abc', (), (), (1,), ('a',), ('a', 'b'), ((),), '']
lista = []
# print(type(t))

for objeto in t:
    if len(objeto) != 0:
        lista.append(objeto)

print(lista)
```

```
[(1,), 'abc', (1,), ('a',), ('a', 'b'), ((),)]
```

Unidad 13

Quiz 1

Q1.

Escriba un programa que genere un arreglo multidimensional de tamaño n×n, basado en el número de entradas, al recibir dos o más n como entradas de los usuarios. En este caso, el contenido del arreglo debe mostrarse de manera que los valores de 0 y 1 se crucen en un patrón de cuadros.

Ejemplo de salida

10101

01010

10101

01010

10101

```
n = int(input("Ingrese la cantidad de filas: "))
# m = int(input("Ingrese la cantidad de columnas: "))
# print(rango_n)
# rango_m = range(m)
print(f"La matriz es {n} x {n}")
matriz = []
for i in range(n):
    lista = []
    for j in range(n):
        if i % 2 == 0 and j % 2 == 0:
            lista.append(1)
        elif i % 2 == 0 and j % 2 != 0:
            lista.append(0)
        elif i % 2 != 0 and j % 2 == 0:
            lista.append(0)
        elif i % 2 != 0 and j % 2 != 0:
            lista.append(1)
    matriz.append(lista)

for x in matriz:
    for y in x:
        print(y, end="")
    print("")
```

```
Ingrese la cantidad de filas: 4
La matriz es 4 x 4
1010
0101
1010
0101
```

Unidad 14

Codificación en papel Quiz 2

Q2

Escribamos un programa que realice la gestión del inventario en una tienda de conveniencia. Para ello, el inventario de los artículos que se venden en las tiendas de conveniencia se almacena en el diccionario de artículos, como se muestra en el siguiente ejemplo. Escriba un programa que reciba el nombre del artículo de los usuarios y devuelva el inventario del mismo. Suponga que se trata de una tienda de conveniencia muy pequeña y que los artículos tratados son los siguientes.

Ejemplo de programa resultados de la ejecución	Introduzca el nombre del artículo: Leche 1
Tiempo	5 minutos

Items

items = {"Café": 7, "Lápiz": 3, "Vaso de papel": 2, "Leche": 1, "Coca-Cola": 4, "Libro": 5}

```
items = {"Café": 7, "Lápiz": 3, "Vaso de papel": 2, "Leche": 1, "Coca-Cola": 4, "Libro": 5}

producto = input("Introduzca el nombre del articulo: ")

if producto in items:
    print(items[producto])

else:
    print("El producto no está en el inventario")
```

```
Introduzca el nombre del articulo: Leche
1
```

Programación por parejas Quiz 1

Q1

Actualicemos el programa para gestionar el inventario de las tiendas de conveniencia que resolvimos en la codificación en papel. En otras palabras, añadir código para aumentar o disminuir el inventario. También, hacer menús simples como la consulta de inventario, el almacenamiento y el envío.

items = {"Cafe": 7, "Puma": 3, "Vaso de papel": 2, "Leche": 1, "coca -cola": 4, "Libro": 5,}

```
items = {"Cafe": 7, "Puma": 3, "Vaso de papel": 2, "Leche": 1, "coca -cola": 4, "Libro": 5,}

while True:

    print("Selecione el menú")
    print("1) comprobar existencia")
    print("2) almacenamiento")
    print("3) liberación")
    print("4) salir")
    print()
    opcion = int(input("Seleccione la opciión del menú: "))
    print()
    if opcion == 1:
        articulo = input("Introduzca el articulo: ")
        if articulo in items:
            print(f"Existencias: {items[articulo]}")
            print()
        else:
            print("El articulo no se encuentra en el inventario")
            print()
    if opcion == 2:
        print(items)
        print()
    if opcion == 3:
        liberacion, cantidad = input("Ingrese el articulo y numero de unidades que desea liberar del inventario: ").split(" ")
        cantidad = int(cantidad)
        if liberacion in items:
            items[liberacion] = items[liberacion] - cantidad
        else:
            print("El articulo no se encuentra en el inventario")
            print()
    if opcion == 4:
        print("Programa terminado")
        break
```

```
Selecione el menú
1) comprobar existencia
2) almacenamiento
3) liberación
4) salir

Seleccione la opciión del menú: 3

Ingrese el articulo y numero de unidades que desea liberar del inventario: Cafe 2
Selecione el menú
1) comprobar existencia
2) almacenamiento
3) liberación
4) salir

Seleccione la opciión del menú: 2

{'Cafe': 5, 'Puma': 3, 'Vaso de papel': 2, 'Leche': 1, 'coca -cola': 4, 'Libro': 5}

Selecione el menú
1) comprobar existencia
2) almacenamiento
3) liberación
4) salir

Seleccione la opciión del menú:
```

Unidad 15

Codificación en papel Quiz 2

La lista estudiante_tupla con tuplas como elementos es la que se muestra a continuación. La tupla, que es el elemento de esta tupla consiste en un (número de identificación del estudiante, nombre, número de teléfono). Usando esto, haga un diccionario para (número de identificación del estudiante: nombre) e imprímalo.

Cuando pregunte por el número de identificación del estudiante, asegúrese de que el número de identificación del estudiante, el nombre y el número de teléfono se impriman como se muestra a continuación.

estudiante_tupla = [('211101', 'David Doe', '010-123-1111'), ('211102', 'John Smith', '010-123-2222'),('211103', 'Jane Carter', '010-123-3333')]

Ejemplo de salida:

211101 : David Doe

211102 : John Smith

211103 : Jane Carter

Ingrese el numero de identificación del estudiante: 211103

211103 es el estudiante Jane Carter y su numero de telefono es 010-123-3333

Escriba el código completo y los resultados esperados en la nota

```
estudiante_tup = (
    ('211101', "David Doe", '010-1234-4500'),
    ('211102', ' John Smith', '010-2230-6540'),
     ('211103', ' Jane Carter', '010-3232-7788')
     )

identificacion = input("Ingrese el numero de identificación del estudiante: ")
diccionario_estudiantes = {}

for estudiante in estudiante_tup:
    diccionario_estudiantes[estudiante[0]] = [estudiante[1], estudiante[2]]


if identificacion in diccionario_estudiantes:
    print(f"{identificacion} es el estudiante {diccionario_estudiantes[identificacion][0]} y su numero de telefono es {diccionari

else:
    print("Identificacion no recocnocida")
```

```
Ingrese el numero de identificación del estudiante: 211103
211103 es el estudiante  Jane Carter y su numero de telefono es 010-3232-7788
```

Unidad 16

Codificación en papel Quiz 1

Intente comprender completamente el concepto básico antes de pasar al siguiente paso. La falta de comprensión de los conceptos básicos aumentará su carga en el aprendizaje de este curso, lo que puede hacer que no apruebe el curso. Puede ser difícil ahora, pero para completar con éxito este curso le sugerimos que entienda completamente el concepto y pase al siguiente paso.

Q1.

Utilice la función set para generar e imprimir el conjunto s1 de la siguiente lista lst

Condiciones variables del programa	lst = ['apple', 'mango', 'banana'] # A list of 3 fruit information
	s1 = {'apple', 'mango', 'banana'} # Set s1 generated from lst

Tiempo	7 Minutos

Ejemplo de salida:

s1 = {'banana', 'Manzana', 'Mango'}

.

Ejemplo de salida:

s1 = {'banana', 'Manzana', 'Mango'}

```
lst = ["manzana", "mango", "banana"]
S1 = set(lst)

print(S1)
```

```
{'manzana', 'banana', 'mango'}
```

Quiz 2

Q2.

Escriba los resultados computacionales de los dos conjuntos siguientes. Encuentra los resultados de 1) a 7).

	s1 = {10, 20, 30, 40} s2 = {30, 40, 50, 60, 70}
Condición para la operación	1) s1 | s2 2) s1 & s2 3) s1 – s2 4) s1 ^ s2 5) s1.issubset(s2) 6) s1.issuperset(s2) 7) s1.isdisjoint(s2)
Tiempo	5 Minutos

```
s1 = {10, 20, 30, 40}
s2 = {30, 40, 50, 60, 70}

# 1) s1 | s2 Unión: {10, 20, 30, 40, 50, 60, 70}
# 2) s1 & s2 intersección: {30, 40}
# 3) s1 - s2 diferenica: {10, 20}
# 4) s1 ^ s2 diferencia simetrica: {10, 20, 50, 60, 70}
# 5) s1.issubset(s2): False
# 6) s1.issuperset(s2): False
# 7) s1.isdisjoint(s2): False

print(s1 | s2)
print(s1 & s2)
print(s1 - s2)
print(s1 ^ s2)
print(s1.issubset(s2))
print(s1.issuperset(s2))
print(s1.isdisjoint(s2))
```

```
{70, 40, 10, 50, 20, 60, 30}
{40, 30}
{10, 20}
{50, 20, 70, 10, 60}
False
False
False
```