FUNDAMENTOS DE PROGRAMACIÓN EN PYTHON

FPP-PE-P6. Estructura de repetición for

ALUMNO: Alejandro Bermúdez Gorozabel

1) Muestra por pantalla los números del 1 al 100.

- Muestra por pantalla los números del 1 al 100 pares.
- Muestra por pantalla la suma de los números del 1 al 100.

Números del 1 al 100

```
for i in range(1, 101):

    print(i);
```

Números del 1 al 100 pares

```
for i in range(1, 101):

    if i%2 == 0:

        print(i);
```

Suma de los números del 1 al 100

```
suma = 0

for i in range(1,101):

    suma += i

print(suma)
```

2) Crea un script que pida un número al usuario del 1 al 10 y muestre la tabla de multiplicar de ese número.

```
numero = int(input("Introduce un número del 1 al 10: "))

print("Tabla del ", numero, ":")



if 1 <= numero <=10:

    for i in range(1, 11):

        print(numero, " * ", i, " = ", numero*i)

else:

    print("El número debe estar entre 1 y 10")
```

3) Crea un script que pida un texto al usuario y cuente el número de vocales que tiene el texto.

- *Pista: utiliza un string con las vocales y comprueba si cada letra está en dicho string*

```
texto = str(input("Introduce un texto: "))

vocales = "aeiou"

contador = 0

for i in texto.lower():

    if i in vocales:

        contador += 1

print("Número de vocales: ", contador)
```

4) Crea un script que pida un número al usuario e imprima por pantalla un triángulo rectángulo con asteriscos con altura igual a dicho número.

```
*
**
***
****
*****
```

```python
altura = int(input("Introduce la altura del triángulo: "))

for asterisco in range (1, altura+1):

    print("*"*asterisco)
```

5) Crea un script que pida un número al usuario e imprima por pantalla un triángulo equilátero con asteriscos con altura igual a dicho número.

```
    *
   ***
  *****
 *******
*********
```

```python
altura = int(input("Introduce la altura del triángulo: "))

for i in range (1, altura+1):

    print(" "*(altura - i), "*"*(2*i - 1))
```

6) Crea un script que utilice un bucle anidado y muestre para cada mes (`enero`, `febrero`, `marzo`, `abril`, etc.) los días del `1` al `30` de dicho mes.

- Modifica el script para que se muestren únicamente los días válidos para cada mes.

```
# EJEMPLO DE SALIDA
ene 1
ene 2
```

```
ene 3
...
ene 30
feb 1
feb 2
...
```

```python
meses = ["enero", "febrero", "marzo", "abril", "mayo", "junio",

        "julio", "agosto", "septiembre", "octubre", "noviembre",
"diciembre"]



for mes in meses:

    if mes == "febrero":

        for dia in range(1, 29):

            print(mes, dia)



    elif mes in ["abril", "junio", "septiembre", "noviembre"]:

        for dia in range(1, 31):

            print(mes, dia)

    else:

        for dia in range(1, 32):

            print(mes, dia)
```

7) Crea un script que utilice un bucle anidado y muestre las tablas de multiplicar del `1` al `10`.

```
# EJEMPLO DE SALIDA
TABLA DEL 1
1*1=1
```

```
1*2=2
1*3=3
1*4=4
1*5=5
...
TABLA DEL 2
2*1=2
2*2=4
...
```

```python
for i in range (1, 11):

    print()

    print("TABLA DEL ", i)

    for factor in range(1,11):

        producto = i*factor

        print(i, " * ", factor, " = ", producto)
```

8) Has descargado este archivo de una página de *dudosa procedencia*.

- El archivo resulta ser un fichero comprimido en **7zip** y tiene **contraseña**, pero la contraseña únicamente se proporciona a usuarios registrados y no hay forma de conseguirla.
- En un foro has leído que la contraseña es `mac-torrent-download.net_yh1`, pero que **cambia** para cada usuario y fichero.
- **Apáñatelas para descomprimirlo por fuerza bruta mediante un script en Python**.

```python
# DESCOMPRIMIR 7ZIP
import subprocess

ruta_7z = "ASO_Capure One ESSENTIALS Style Pack mac-torrent-download.net.7z"
ruta_destino = "./"
password = "mac-torrent-download.net_"
exec = subprocess.run(["7z", "x", f"-p{password}", ruta_7z, f"-o{ruta_destino}", "-y"], stdout=subprocess.DEVNULL, stderr=subprocess.DEVNULL)
```

```
exec.returncode      # 0 si se descomprimió con éxito
```

9) Crea un script que utilice:

- Una diccionario (`{}`) con los valores de las direcciones MAC de los equipos del aula indexados por el número de equipo.
- Una variable con una cadena con la disposición de los equipos del aula.

El script realizará las siguientes acciones:

- Realizará un barrido de las direcciones MAC-IP encontradas en la red local utilizando el comando `arp-scan` de Linux.
- A continuación recorrerá el mapa del aula por filas y columnas y para cada número de máquina:
  Obtendrá su mac del array de macs.
  Buscará si su mac se encuentra en la salida del comando arp-scan.
  En caso afirmativo obtendrá su IP y la guardará en una lista de IPs indexada por número de máquina.
  Mostrará el número de máquina según la disposición del mapa en color verde si estaba encendida o roja si estaba apagada.

```
# Direcciones MAC del aula
11 = 70:85:c2:63:6a:5b
12 = 70:85:c2:76:a9:1b
13 = 70:85:c2:63:69:0f
14 = 70:85:c2:63:6c:49
21 = 70:85:c2:76:a9:1d
22 = 70:85:c2:76:a7:97
23 = 70:85:c2:8b:b9:b9
24 = a8:a1:59:13:d4:22
25 = d8:bb:c1:42:b9:db
26 = 70:85:c2:8b:b9:b7
31 = 70:85:c2:76:a8:fb
32 = 70:85:c2:76:a6:3e
33 = 70:85:c2:76:a6:36
34 = 70:85:c2:76:a9:23
35 = 70:85:c2:76:aa:04
36 = 70:85:c2:76:a5:80
41 = 70:85:c2:76:a4:a7
42 = 70:85:c2:76:a8:df
43 = 70:85:c2:93:45:3f
44 = 70:85:c2:76:a6:5b
45 = 70:85:c2:76:a9:f8
46 = 70:85:c2:8b:b9:bd
51 = 70:85:c2:76:a9:c4
```

```
52 = d8:50:e6:48:f9:e1
53 = d8:50:e6:48:f9:cc
54 = d8:50:e6:48:f9:c9
55 = 70:85:c2:93:3a:9f
56 = 70:85:c2:93:55:f6


# Mapa del aula
56 66 __ 54 53 52 51
46 45 __ 44 43 32 31
36 35 __ 34 33 32 31
26 25 __ 24 23 22 21
__ __ __ 14 13 12 11


# COMO LLAMAR AL COMANDO ARP-SCAN
# - ¡Ejecutar el script como root con sudo!
# - En una máquina virtual configurar el adaptador de red como
Adaptador puente para ver las MACs del aula
import subprocess
arp_output=subprocess.run(["arp-scan", "-l"], capture_output=True,
text=True)

# COMO ESCRIBIR SIN SALTO DE LÍNEA
print("hola", end="")

# COMO ESCRIBIR CON COLOR VERDE
print("\033[38;5;82mHOLA\033[0m")

# COMO ESCRIBIR CON COLOR ROJO
print("\033[38;5;160mHOLA\033[0m")
```