EJERCICIO 1

Transformar el siguiente código de manera que no sea necesario usar la estructura **if**:

```
opcion = input("¿Que tipo de comida quieres?");

if opcion.lower()=="fruta":
    res = "Manzana, platano o pera";
elif opcion.lower()=="verdura":
    res = "Tomate, lechuga o zanahoria";
elif opcion.lower()=="carne":
    res = "Cerdo, ternera o pollo";
elif opcion.lower()=="pescado":
    res = "Sardinas, caballa ó salmonete";
else:
    res = "No tenemos de ese tipo de comida";

print(res)
```

EJERCICIO 2

Crear un código Python que transforme de forma equivalente el código que aparece en la captura eliminando la estructura condicional sustituyéndolo por llamadas a funciones:

```
opcion=input("¿Quieres mostrar,modificar o longitud?")
lista=[4,8,14,22]

if opcion=="mostrar":
    for i in range(0,len(lista)):
        print("Posicion",i,":",lista[i])
elif opcion=="modificar":
    for i in range(0,len(lista)):
        lista[i]*=2
    print(lista)
elif opcion=="longitud":
    print(len(lista))
else:
    print("Error")
```

EJERCICIO 3

Programar los siguientes mini-juegos de dados (random). No es necesario usar listas y debéis controlar los errores asociados a valores incorrectos (numero de jugadores negativo, un dado es de 6 caras, etc):

a) Escriba un programa que pida un número de dados, que pida un valor a conseguir y que tire después el número de dados indicado. El jugador gana si saca el valor ganador.

b) Escriba un programa que pida un número de dados, que tire el número de dados indicado y diga cuál es el valor más alto obtenido.

c) Escriba un programa que pida un número de dados y tire esa cantidad para dos jugadores. El jugador que saque el valor más alto, gana.

d) Escriba un programa que pida un número de dados y tire esa cantidad de dados. El primer jugador obtiene un punto por cada dado par. El segundo jugador obtiene un punto por cada dado impar. El jugador que saque más puntos, gana.

e) Escriba un programa que pida un número de jugadores y tire un dado para cada jugador. El jugador que saque el valor más bajo, gana.

f) Escriba un programa que pida un número de dados y tire esa cantidad de dados para dos jugadores. El jugador que saque más puntos sumando su valor más alto y su valor más bajo, gana.

g) Escriba un programa que pida un número de dados y tire esa cantidad de dados. Si no salen dos dados iguales seguidos, el jugador gana. Si salen, pierde.

EJERCICIO 4

Realizar un programa que genere contraseñas aleatorias. El programa pide cuantas contraseñas que se van generar y después la longitud de las mismas. Los caracteres admitidos en las contraseñas son:

0123456789@ABCDEFGHIJKLMNOPQRSTUVXYZ\]_abdefghijklmnpqrstuvwxyz

Consejo: Partir de un String con los caracteres admitidos para una contraseña, usar list para convertirlo en una lista y choice para seleccionar valores aleatorios de la lista

EJERCICIO 5

Imagina que estás desarrollando un programa para una escuela que necesita gestionar las notas de los alumnos. La escuela nos proporciona las siguiente notas de ejemplo:

María => 7.9
Carlos => 9.2
Pedro => 5.2
Isabel => 4.7
Luis =>3.9
Miguel => 6.0

Se te pide programar las siguientes funcionalidades usando diccionarios:

- Hacer la media de todos los alumnos.
- Buscar un alumno por nombre y nos muestre su nota media, si no existe el alumno mostrar un mensaje informativo.
- Mostrar el alumno con mayor nota y el alumno con menor nota.
- Crear otro diccionario con solo los aprobados.
- Mostrar los nombres de los alumnos de menor a mayor.

EJERCICIO 6

Escribir un programa que, partiendo de una estructura de datos con los precios de las frutas de una tienda, permita realizar las siguientes acciones:

Fruta	Precio/Kg
Plátano	1.35
Manzana	1.80
Pera	0.85
Naranja	0.70

- Dado un número de kilos X mostrar el precio total de llevarse X kilos de todas las frutas. Hacer un resumen de cada cantidad de dinero por fruta y mostrar el total al final.
- Mostrar la fruta con menor precio por kilo.
- Crear otro diccionario a partir del anterior con las frutas ordenadas de mayor a menor precio.
- Crear otro diccionario a partir del anterior con solo las frutas que valgan más de 1€/kh.

EJERCICIO 7

Escribir un programa que permita gestionar la base de datos de una tienda de videojuegos. Los datos se deben transformar a una lista de diccionarios a partir del siguiente string:

```
Juego;Precio;Genero;Disponible
Super Mario Bros;30.0;Plataformas;True
Silent Hill;12.0;Terror;False
Resident Evil;45.0;Terror;False
Halo 5;20.0;Shooter;True
Final Fantasy VII;19.90;JRPG;True
PES 2021;9.99;Deportivo;True
FIFA 2022;59.99;Deportivo;False
```

Posteriormente hacer un menú de opciones que permita hacer lo siguiente:

1) Buscar un juego por nombre.

2) Listar todos los juegos que esten disponibles.

3) Mostrar los juegos (solo nombre y precio) ordenados según el precio de manera descendente.

4) Mostrar los juegos (solo su nombre) en orden alfabetico normal.

5) Mostrar nombre y el precio del juego más barato.

6) Crear un string a partir de la lista con el mismo formato del que se ha utilizado de entrada.

7) Crear un diccionario con los juegos cuyo precio sea menor de X y esten disponibles. La clave debe ser el nombre y el valor el precio.

EJERCICIO 8

Escribir un programa que permita gestionar la base de datos de clientes de una empresa. Los datos del cliente son DNI/CIF, nombre, dirección, teléfono, correo y deuda neta. Los datos se almacenan en un estructura de lista de diccionarios. El programa debe preguntar al usuario por una opción del siguiente menú:

(1) Buscar cliente por nombre

(2) Añadir un cliente preguntando sus datos por teclado

(3) Listar todos los clientes

(4) Mostrar clientes ordenados por deuda mayor a menor. No modificar la estructura original.

(5) Backup datos de clientes (string con separadores y saltos de lineas), Por ejemplo:

DNI,Nombre,Dirección,Teléfono,Correo,Deuda Neta,Alta
12345678A,John Doe,Calle 123,667-123456,johndoeQAexample.com,1000.50,True
98765432B,Jane Smith,Avenida 456,643-234567,janesmithQexample.com,750.20,False
23456789C,Michael Johnson, Plaza 789,645-345678,michaeljohnsonQdexample.com,2000.75,True
34567890D,Emily Davis,Carrera 1011,634-456789,emilydavisQexample.com,300.00,False
45678901E,Christopher Wilson,Alameda 1314,666-567890,chriswilson@example.com,150.60,True

(6) Recuperar datos de backup generado en el punto anterior. Previamente se borran todos los datos que hubiera en la estructura.

(7) Crear un diccionario cuya clave sean los nombres de los clientes y el valor su deuda si el cliente tiene deuda mayor de 500€

- Se recomienda programar las opciones usando funciones para organizar el codigo
- Realizar otra versión usando un estructura de diccionario de diccionarios.