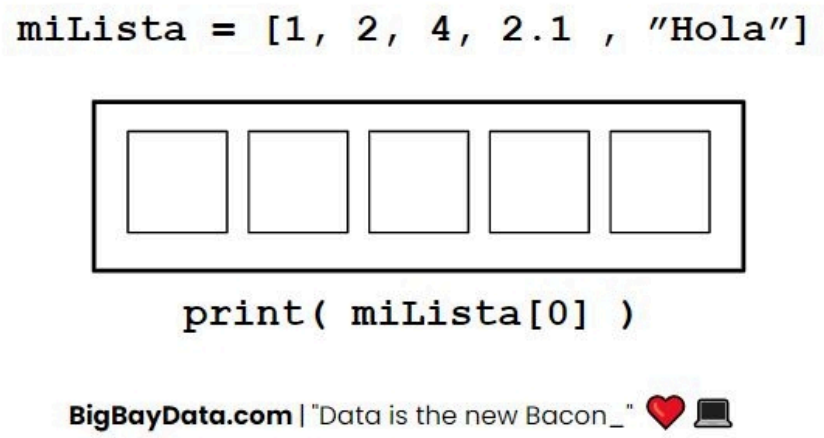
Listas en Python



> Link: https://www.bigbaydata.com/ejercicios-de-listas-en-python/

Objetivos

- Qué es una lista
- Funciones útiles de las listas
- Accesos y modificaciones
- Errores comunes con listas

Qué es una Lista

- Puedes pensar en ella como una secuencia de elementos, de lo que sea. Exactamente igual que antes con los textos.
- Las listas son bastante flexibles y versátiles, ya que pueden contener elementos de diferentes tipos, como números, cadenas, booleanos e incluso otras listas.

In [ ]:
```python
# Crear una lista de números
mi_lista_de_numeros = [1, 2, 3, 4, 5]

# Crear una lista de cadenas
mi_lista_de_cadenas = ["Hola", "Mundo", "!"]

# Crear una lista mixta
mi_lista_mixta = [1, "dos", True, 3.14]

# Crear una lista vacía
lista_vacia = []
```

Funciones más útiles de las Listas

A continuación, te describo las utilidades más interesantes de las listas:

- **l.append():** Agrega un elemento al final de la lista.
- **l.extend():** Extiende la lista agregando los elementos de otra lista.
- **l.insert():** Inserta un elemento en una posición específica de la lista.
- **l.remove():** Elimina la primera aparición del elemento especificado de la lista.
- **l.pop():** Elimina y devuelve el elemento en la posición especificada de la lista.
- **l.index():** Devuelve el índice de la primera aparición del elemento especificado a.
- **l.sort():** Ordena los elementos de la lista en su lugar.
- **l.sorted():** Devuelve una nueva lista ordenada sin modificar la original.

In [ ]:
```
my_list = [1, 2, 3]
my_list.append(4)
print(my_list)  # Salida: [1, 2, 3, 4]
```

In [ ]:
```
my_list = [1, 2, 3]
another_list = [4, 5, 6]
my_list.extend(another_list)
print(my_list)  # Salida: [1, 2, 3, 4, 5, 6]
```

In [ ]:
```
my_list = [1, 2, 3]
my_list.insert(1, 4)
print(my_list)  # Salida: [1, 4, 2, 3]
```

In [ ]:
```
my_list = [1, 2, 3, 2]
my_list.remove(2)
print(my_list)  # Salida: [1, 3, 2]
```

In [ ]:
```
my_list = [1, 2, 3]
element = my_list.pop(1)
print(element)  # Salida: 2
print(my_list)  # Salida: [1, 3]
```

In [ ]:
```
my_list = [1, 2, 3, 2]
index = my_list.index(2)
print(index)  # Salida: 1
```

In [ ]:
```
my_list = [1, 2, 3]
my_list.reverse()
print(my_list)  # Salida: [3, 2, 1]
```

In [ ]:
```
my_list = [3, 1, 2]
my_list.sort()
print(my_list)  # Salida: [1, 2, 3]
```

In [ ]:
```
my_list = [3, 1, 2]
sorted_list = sorted(my_list)
print(sorted_list)  # Salida: [1, 2, 3]
print(my_list)      # Salida: [3, 1, 2]
```

Accesos y modificaciones

In [ ]:
```python
mi_lista = [10, 20, 30, 40, 50]

# Acceder al primer elemento (índice 0)
primer_elemento = mi_lista[0]  # primer_elemento = 10

# Acceder al tercer elemento (índice 2)
tercer_elemento = mi_lista[2]  # tercer_elemento = 30

# Acceder al último elemento (índice -1)
ultimo_elemento = mi_lista[-1]  # ultimo_elemento = 50

# MODIFCIACIONES... (Voy al vagón y añado el dato que me interesa)

# Modificar el segundo elemento (índice 1)
mi_lista[1] = 25  # mi_lista ahora es [10, 25, 30, 40, 50]

# Modificar el último elemento (índice -1)
mi_lista[-1] = 55  # mi_lista ahora es [10, 25, 30, 40, 55]
```

Errores más comunes con Listas

In [ ]:
```python
# 1. Rangos
mi_lista = [1, 2, 3]
print(mi_lista[3])  # Esto generará un IndexError
```

In [ ]:
```python
# 2. NO APROVECHAR LAS FUNCIONALIDADES QUE OFRECE PYTHON... Append(), extend()..
```

In [ ]:
```python
# 3. Confusión de utilidad de funciones
lista1 = [1, 2, 3]
lista2 = [4, 5, 6]

lista1.append(lista2)  # Agrega lista2 como un solo elemento
print(lista1)  # [1, 2, 3, [4, 5, 6]]

lista1 = [1, 2, 3]  # Reinicializamos lista1
lista1.extend(lista2)  # Agrega los elementos de lista2 individualmente
print(lista1)  # [1, 2, 3, 4, 5, 6]
```

Ejercicios Resueltos

In [ ]:
```python
#Ej1.
```

In [ ]:
```python
#Ej2.

# Obtener la lista de nombres de asignaturas
asignaturas = input("Inserta la lista de los nombres de las asignaturas del inst

# Almacenar las evaluaciones
evaluaciones = []

# Iterar sobre cada asignatura
for asignatura in asignaturas: #Iteramos sobre la lista de asignaturas...
    # Obtener el número de alumnos para la asignatura
    num_alumnos = int(input("Introduce el número de alumnos para la asignatura "
```

```
    # Obtener las puntuaciones de los alumnos
    puntuaciones = [] # Creamos una lista que irá añadida a la asignatura...
    for i in range(num_alumnos):
        puntuacion = float(input("Introduce la puntuación del alumno"+str(i+1)+"
        puntuaciones.append(puntuacion) #Añadimos puntuaciones...

    # Calcular la media de las puntuaciones
    total = sum(puntuaciones)
    media = total / len(puntuaciones)

    # Calcular el número de suspensos
    suspensos = 0
    for nota in puntuaciones:
        if nota < 5:
            suspensos += 1
    # suspensos = sum(1 for nota in puntuaciones if nota < 5) Esta línea hace ex

    # Almacenar la evaluación
    evaluacion = [asignatura, num_alumnos, media, suspensos]
    evaluaciones.append(evaluacion)

# Mostrar el resultado de las evaluaciones
print("Resultado de las evaluaciones este año:")
for evaluacion in evaluaciones:
    asignatura, num_alumnos, media, suspensos = evaluacion
    print("[",asignatura,",", num_alumnos, "alumnos. Nota media: ",media, "Suspe
```

In [1]:

```
#Ej3.

# Pedir al usuario que introduzca los nombres
print("Introduce los nombres... (-1 para terminar)")
nombres = []
nombre = input()
while nombre != "-1":
    nombres.extend(nombre.split(","))
    nombre = input()

# Inicializar listas para almacenar los nombres y sus frecuencias
nombres_unicos = []
frecuencia_nombres = []

# Contar la frecuencia de cada nombre
'''
La estrategia es la siguiente: si vemos que el nombre no existe, añadimos un 1 a
Las siguientes veces, usando index, podemos hacer un +1 en su posición.
'''

for nombre in nombres:
    if nombre not in nombres_unicos:
        nombres_unicos.append(nombre)
        frecuencia_nombres.append(1)
    else:
        indice = nombres_unicos.index(nombre) #Localizamos la posición del vagón
        frecuencia_nombres[indice] += 1 #Añadimos 1 unidad en ese vagón...

# Encontrar el nombre más común y su frecuencia
indice_max_frecuencia = frecuencia_nombres.index(max(frecuencia_nombres)) #Obten
nombre_mas_comun = nombres_unicos[indice_max_frecuencia]
frecuencia_max = frecuencia_nombres[indice_max_frecuencia]
```

```
# Mostrar el nombre más común y su frecuencia
print("El nombre más común es:", nombre_mas_comun)
print("Frecuencia:", frecuencia_max)
```

```
Introduce los nombres... (-1 para terminar)
El nombre más común es: juan
Frecuencia: 2
```

In [ ]:
```
#Ej4.
# Pedir al usuario que introduzca los nombres
print("Introduce los nombres... (-1 para terminar)")
nombres = []
nombre = input()
while nombre != "-1":
    nombres.extend(nombre.split(","))
    nombre = input()

# Eliminar duplicados
nombres_unicos = []
for nombre in nombres:
    if nombre not in nombres_unicos: # Evaluamos que no estén...
        nombres_unicos.append(nombre)

# Inicializar listas para almacenar las frecuencias
frecuencia_nombres = [0] * len(nombres_unicos)

# Contar la frecuencia de cada nombre
for nombre_completo in nombres:
    for indice, nombre in enumerate(nombres_unicos):
        if nombre in nombre_completo:
            frecuencia_nombres[indice] += 1

# Encontrar el nombre más común y su frecuencia
indice_max_frecuencia = frecuencia_nombres.index(max(frecuencia_nombres))
nombre_mas_comun = nombres_unicos[indice_max_frecuencia]
frecuencia_max = frecuencia_nombres[indice_max_frecuencia]

# Mostrar el nombre más común y su frecuencia
print("El nombre más común es:", nombre_mas_comun)
print("Frecuencia:", frecuencia_max)
```

In [4]:
```
#Ej5.

# Pedir al usuario que ingrese un número
numero = int(input("Ingresa un número para calcular su tabla de multiplicar: "))

# Inicializar la lista para almacenar los resultados de la multiplicación
tabla_multiplicar = []

# Calcular la tabla de multiplicar y almacenar los resultados en la lista
for i in range(21): # Recordemos hacer i + 1 en el for
    resultado = numero * i
    tabla_multiplicar.append(resultado)

# Mostrar la tabla de multiplicar
print("Tabla de multiplicar del número ",numero,":")
for i in range(21):
    resultado = tabla_multiplicar[i]
    print(numero,"x", i," = ", resultado)
```

```
Tabla de multiplicar del número  3 :
3 x 0  =  0
3 x 1  =  3
3 x 2  =  6
3 x 3  =  9
3 x 4  =  12
3 x 5  =  15
3 x 6  =  18
3 x 7  =  21
3 x 8  =  24
3 x 9  =  27
3 x 10  =  30
3 x 11  =  33
3 x 12  =  36
3 x 13  =  39
3 x 14  =  42
3 x 15  =  45
3 x 16  =  48
3 x 17  =  51
3 x 18  =  54
3 x 19  =  57
3 x 20  =  60
```

In [ ]:
```python
#Ej6.

# Inicializar lista con los primeros 10 números primos. Daba igual qué números c
primos = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]

# Ordenar de mayor a menor
primos.sort(reverse=True)

# Imprimir la lista de números primos ordenados de mayor a menor
print("Lista de números primos ordenados de mayor a menor:")
print(primos)
```

In [ ]:
```python
#Ej7.

# Simular una cesta de la compra
cesta_compra = ["Manzanas", "Plátanos", "Naranjas", "Leche", "Pan"]

# Eliminar el último elemento de la lista
ultimo_elemento = cesta_compra.pop()

# Invertir los elementos de la lista
cesta_compra.reverse()

# Mostrar la lista resultante
print("Después de eliminar el último elemento y invertir la lista, la cesta de l
print(cesta_compra)

# Este tipo de mecánica se implementa para las compras de empresas como Carrefou
# o páginas de compras de muchos productos baratos y gestión de cestas...
```

In [ ]:
```python
#Ej8.

# Definir una lista vacía para almacenar la información de los Pokémon
pokedex = []

# Definir los nombres de los primeros 10 Pokémon
```

```
nombres_pokemon = ["Bulbasaur", "Ivysaur", "Venusaur", "Charmander", "Charmeleon
                   "Charizard", "Squirtle", "Wartortle", "Blastoise", "Caterpie"

# Definir las estadísticas de los primeros 10 Pokémon
ataques = [49, 62, 82, 52, 64, 84, 48, 63, 83, 30]
hp = [45, 60, 80, 39, 58, 78, 44, 59, 79, 45]
defensa = [49, 63, 83, 43, 58, 78, 65, 80, 100, 35]
velocidad = [45, 60, 80, 65, 80, 100, 43, 58, 78, 45]
at_Esp = [65, 80, 100, 60, 80, 109, 50, 65, 85, 20]
def_Esp = [65, 80, 100, 50, 65, 85, 64, 80, 105, 20]

# Recoger los datos y guardarlos en la lista pokedex
for i in range(10):
    pokemon = [nombres_pokemon[i], ataques[i], hp[i], defensa[i], velocidad[i],
    pokedex.append(pokemon)

# Mostrar la Pokedex con las estadísticas de los primeros 10 Pokémon
print("Estadísticas de los primeros 10 Pokémon en la Pokedex:")
for pokemon in pokedex:
    print("Nombre:", pokemon[0])
    print("Ataque:", pokemon[1])
    print("HP:", pokemon[2])
    print("Defensa:", pokemon[3])
    print("Velocidad:", pokemon[4])
    print("Ataque Especial:", pokemon[5])
    print("Defensa Especial:", pokemon[6])
    print("------------------------")
```

In [ ]:

```
#Ej9.


# Crear una lista vacía para almacenar la planificación de vuelos
planificacion_vuelos = []

# Días de la semana
dias_semana = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado", "D

# Recoger la información de los vuelos por día de la semana
for dia in dias_semana:
    print(f"Ingrese la planificación de vuelos para el día {dia}:")
    vuelos_dia = []  # Lista para almacenar los vuelos del día

    '''
    Una nueva estrategia de recogida de dato: While True con un break al final.
    La idea es recoger los datos hasta que se de cierta condición de salida...
    En el bloque de bucles se ha hablado de que while True es peligroso por esa
    Si bien no es del todo buena práctica, se suele ver bastante en el mundo del

    '''
    while True: # 1. Entra en bucle...
        vuelo = []  # Lista para almacenar los datos de cada vuelo
        vuelo.append(input("Hora del vuelo (HH:MM): "))
        vuelo.append(input("Compañía aérea: "))
        vuelo.append(input("Duración estimada del vuelo: "))
        vuelo.append(input("Tipo de avión: "))

        # Agregar la lista del vuelo a la lista de vuelos del día
        vuelos_dia.append(vuelo)

        # Preguntar al usuario si desea agregar otro vuelo para este día
```

```
        continuar = input("¿Desea agregar otro vuelo para este día? (si/no): ")
        if continuar.lower() != "si":
            break   # 2. Cierra y sale del bloque while...
        '''
        Recuerda que podemos utilizar condiciones también, a modo while (continu
        '''


    # Agregar la lista de vuelos del día a la planificación de vuelos
    planificacion_vuelos.append(vuelos_dia)

# Mostrar la planificación de vuelos
print("\nPlanificación de vuelos del aeropuerto cercano:")
for i, vuelos_dia in enumerate(planificacion_vuelos):
    print("\n", dias_semana[i])
    for vuelo in vuelos_dia:
        print("Hora:", vuelo[0])
        print("Compañía:", vuelo[1])
        print("Duración estimada:", vuelo[2])
        print("Tipo de avión:", vuelo[3])
        print("-----------------------")
```

In [ ]:

```
#Ej10.
# Sé creativo y mándamelo por correo :D
'''
Manda tu solución por correo a nuestro buzón: bigbaydata@gmail.com. A cambio, te
'''
```