Deusto Facultad de Ingeniería Ingeniaritza Fakultatea	EXAMEN DE PROGRAMACIÓN I	23-NOVIEMBRE-2023

Nombre: _____

DNI: _____ *Aula:*_____ *Fila:*_____ *Columna* _____

Confidential

Instrucciones

- El ejercicio a pie de máquina se entregará al terminar el examen.
- NO está permitida la conexión a internet en el examen. Tampoco la utilización de tu propio portátil ni la consulta de apuntes o de otros códigos.
- Guarda el código de cada ejercicio en un fichero que se llame “EjerX-NombreApellidos.py”, por ejemplo: Ejer1-BorjaSanz.py. Comprime todos los ficheros para su entrega. También puedes entregarlo en un único fichero.
- El ejercicio debe incluir un comentario al inicio con el nombre, apellidos y DNI del alumno o alumna.

Estructura de datos

El aeropuerto de Loiu nos ha pedido ayuda para poder gestionar los vuelos del aeropuerto. Para ello, nos ha dicho que la estructura en la que almacenan la información es la siguiente:

- **Vuelos:** Cada vuelo está representado por un diccionario con claves para almacenar detalles como el número de vuelo, destino, origen, tiempo de salida, tiempo de llegada, y una lista de pasajeros. Un ejemplo de vuelo es el siguiente:

```
vuelo = {
"numero_vuelo": "AA123",
"origen": "Ciudad A",
"destino": "Ciudad B",
"salida": "08:00",
"llegada": "11:00",
"pasajeros": []
}
```

- **Pasajeros:** De forma análoga, cada pasajero está representado por un diccionario que contiene como nombre, número de identificación, y número de asiento. Un ejemplo de un pasajero es el siguiente:

```
pasajero = {
  "nombre": "Juan Pérez",
  "identificacion": "ID123456",
  "asiento": "12A"
}
```

- **Lista de vuelos.** Finalmente, los vuelos están almacenados en una lista que contiene cada uno de los diccionarios anteriormente descritos.

Tenéis una plantilla para el examen en la que tenéis un conjunto de datos de prueba, así como funciones auxiliares que os ayudarán en el desarrollo del examen.

Deusto Facultad de Ingeniería Ingeniaritza Fakultatea	EXAMEN DE PROGRAMACIÓN I	23-NOVIEMBRE-2023

Ejercicio 1 (2 puntos)

Confidential

Crea una función llamada calculo_capacidad_aeropuerto que reciba como función la lista de vuelos y devuelva un float que indique la capacidad de pasajeros disponible en el aeropuerto, teniendo en cuenta que el aeropuerto puede gestionar a 100 pasajeros.

Ejemplo de ejecución:

```
print(calculo_capacidad_aeropuerto(vuelos)) # Devuelve 0.4
```

Ejercicio 2 (3 puntos)

Crea una función agregar_pasajero que permita agregar un nuevo pasajero a un vuelo específico. La función deberá recibir como parámetros el número de vuelo, y la información del pasajero (nombre, identificación, asiento). Si el vuelo no existe, la función debe mostrar un mensaje indicativo. En caso de que exista, deberá escribir un mensaje por pantalla indicando que se ha añadido al pasajero de forma correcta.

Ejemplo de ejecución:

```
# Nuevo pasajero
nuevo_pasajero = {"nombre": "Sofía Martín", "identificacion": "ID1010", "asiento": "12C"}

agregar_pasajero(vuelos, "AA123", nuevo_pasajero)
# Salida: Sofía Martín ha sido agregada/o al vuelo AA123.
agregar_pasajero(vuelos, "AA124", nuevo_pasajero)
# Salida: No se encontró el vuelo con número AA124.
```

Ejercicio 3 (2 puntos)

Crea una función buscar_vuelo_por_destino que permita listar todos los vuelos que van hacia un destino específico. La función recibirá como parámetro el destino y retornará una lista de vuelos que coincidan con este criterio.

Ejemplo de ejecución:

```
print(buscar_vuelo_por_destino(vuelos, "Miami"))
# Devuelve [{'numero_vuelo': 'BB456', 'origen': 'Bogotá', 'destino': 'Miami', 'salida': '14:00',
'llegada': '18:00', 'pasajeros': [{'nombre': 'Luisa Fernández', 'identificacion': 'ID1003', 'asiento':
'11A'}]}]
```

Deusto Facultad de Ingeniería Ingeniaritza Fakultatea	EXAMEN DE PROGRAMACIÓN I	23-NOVIEMBRE-2023

Ejercicio 4 (3 puntos)

Confidential

Crea una función denominada calcular_itinerarios, que presente un informe con todas las posibles conexiones (es decir, la posibilidad de llegar del origen al destino en una sóla escala).

Ejercicio 4.a (2 puntos)

La función debería recibir como parámetro un origen y un destino final, y luego buscar combinaciones de vuelos que permitan a un pasajero viajar desde el origen hasta el destino, considerando posibles escalas. En caso de que no exista una escala, mostrará los datos correspondientes.

Ejercicio 4.b (1 punto)

Además, la función podría calcular el tiempo total de viaje, incluyendo el tiempo de vuelo y las horas de espera entre conexiones. Para ello, tienes la función format_hora que tienes en la plantilla. Esta función convierte una cadena de caracteres en una variable de tipo datetime, lo que te permitirá hacer operaciones con el tiempo (por ejemplo, restar dos horas y que te devuelva el tiempo entre ellas).

Ejemplo de ejecución

```
calcular_itinerarios(vuelos, "Madrid", "Londres")
```

Salida:
Itinerario: AA123 -> DD101
Tiempo de espera entre vuelos: 1:00:00
Tiempo total de viaje: 5:30:00

```
calcular_itinerarios(vuelos, "Madrid", "Bilbao")
```

Salida:
No se han encontrado conexiones para esas ciudades