Se quieren analizar los datos de las reservas de un hotel. Para ello se dispone de un archivo en formato CSV codificado en UTF-8. En cada línea del archivo se recoge la siguiente información: el nombre y DNI del cliente, las fechas de entrada y salida, el tipo de habitación, el número de personas, el precio por noche, y qué servicios adicionales se han contratado.

Las primeras líneas son las que se muestran a continuación:

```
nombre, dni, fecha_entrada, fecha_salida, tipo_habitacion, num_personas, precio_noche, servicios_adicionales
Ana Fernández,98762912S,2022-01-02,2022-01-06,Suite,4,202.97,"Parking,Gimnasio,Spa"
María Fernández,25061289Y,2022-01-01,2022-01-03,Familiar,4,83.77,
```

Observe que algunas reservas no incluyen servicios adicionales, en cuyo caso aparece una cadena vacía en dicho campo; en otras ocasiones, puede haber varios servicios contratados, apareciendo sus nombres separados por comas. **El orden en que aparecen las reservas en el CSV es arbitratio, no necesariamente cronológico.**

Utilice obligatoriamente la `NamedTuple` **Reserva** que se define a continuación:

```
from typing import NamedTuple, List
from datetime import date
Reserva = NamedTuple("Reserva",
                     [("nombre", str),
                      ("dni", str),
                      ("fecha_entrada", date),
                      ("fecha_salida", date),
                      ("tipo_habitacion", str),
                      ("num_personas", int),
                      ("precio_noche", float),
                      ("servicios_adicionales", List[str])
                     ])
```

Se pide implementar las siguientes funciones (en el módulo `reservas.py`) y sus tests correspondientes (en el módulo `reservas_test.py`). Las puntuaciones indicadas para cada ejercicio incluyen la realización de los tests. Tenga en cuenta que puede definir funciones auxiliares cuando lo considere.

Para cada función solicitada, se le proporciona el prototipo de la función, así como una posible salida del test correspondiente.

NOTA: Si necesita calcular el número de días que van de una fecha `fecha1` a otra `fecha2`, puede restarlas (se obtiene un objeto de tipo `timedelta`) y después acceder al atributo `days`, es decir: `(fecha2 - fecha1).days`

1. **lee_reservas**: recibe la ruta de un fichero CSV, y devuelve una lista de tuplas de tipo Reserva conteniendo todos los datos almacenados en el fichero. (1 punto)

```
def lee_reservas(ruta_fichero: str) -> List[Reserva]
```

```
Test lee_reservas
Total reservas: 496
Las tres primeras:
        Reserva(nombre='Ana Fernández', dni='98762912S', fecha_entrada=datetime.date(2022, 1,
2), fecha_salida=datetime.date(2022, 1, 6), tipo_habitacion='Suite', num_personas=4,
precio_noche=202.97, servicios_adicionales=['Parking', 'Gimnasio', 'Spa'])
        Reserva(nombre='María Fernández', dni='25061289Y', fecha_entrada=datetime.date(2022,
1, 1), fecha_salida=datetime.date(2022, 1, 3), tipo_habitacion='Familiar', num_personas=4,
precio_noche=83.77, servicios_adicionales=[])
        Reserva(nombre='Laura López', dni='13728274B', fecha_entrada=datetime.date(2022, 1,
2), fecha_salida=datetime.date(2022, 1, 10), tipo_habitacion='Estandar', num_personas=1,
precio_noche=87.58, servicios_adicionales=[])
```

2. **total_facturado**: recibe una lista de reservas, una fecha inicial y una fecha final, y devuelve el total facturado en todas las reservas cuya fecha de entrada esté comprendida entre esas fechas. La cantidad facturada correspondiente a una reserva se calcula multiplicando el número de días totales de la reserva por el precio por noche. Si la fecha inicial es `None` se hace el cálculo sin limitar la fecha mínima de las reservas. Si la fecha final es `None` se hace el cálculo sin limitar la fecha máxima de las reservas. (1 punto)

```
def total_facturado(reservas: List[Reserva],
                    fecha_ini: Optional[date] = None,
                    fecha_fin: Optional[date] = None) -> float
```

```
Test total_facturado
En todo el periodo de datos: 244275.89000000028
Desde 1 de febrero de 2022 hasta 28 de febrero de 2022: 19098.12
Desde 1 de febrero de 2022 (fecha final None): 221532.13000000015
Hasta 28 de febrero de 2022 (fecha inicio None): 41841.88
```

3. **reservas_mas_largas**: recibe una lista de reservas y un entero n, y devuelve las n tuplas (nombre, fecha_entrada) más largas (es decir, con mayor número de días entre la fecha de salida y la fecha de entrada). (1 punto)

```
def reservas_mas_largas(reservas:List[Reserva],n:int = 3) -> List[Tuple[str, date]]
```

```
Test reservas_mas_largas
Con n= 3: [('Laura López', datetime.date(2022, 1, 2)), ('Sofía García', datetime.date(2022, 1,
2)), ('Miguel Sánchez', datetime.date(2022, 1, 2))]
```

4. **cliente_mayor_facturacion**: recibe una lista de reservas y un conjunto de servicios, y devuelve una tupla (dni, total_facturado) con el dni del cliente al que se le ha facturado más, y el total facturado correspondiente, teniendo en cuenta solo aquellas reservas en las que se haya contratado alguno de los servicios adicionales indicados. Si no se especifican servicios (el parámetro es `None`), se procesarán todas las reservas. (1,5 puntos)

```
def cliente_mayor_facturacion(reservas: List[Reserva],
                              servicios: Optional[Set[str]] = None
                              ) -> Tuple[str, float]
```

```
Test cliente_mayor_facturacion
Sin filtrar por servicios: ('63550791C', 3893.2200000000003)
Con Parking: ('71828448T', 3008.17)
Con Parking o Spa: ('38747931S', 3216.0699999999997)
```

5. **servicios_estrella_por_mes**: recibe una lista de reservas y un conjunto de tipos de habitación, y devuelve un diccionario en el que para cada uno de los meses del año (enero, febrero, ...) se indica cuál es el servicio adicional que fue solicitado en un mayor número de reservas, de entre aquellas reservas de uno de los tipos de habitación indicados. Si no se especifica un conjunto de tipos de habitación (el parámetro es `None`) se procesarán todas las reservas. Las claves del diccionario deben ser los nombres de los meses (por ejemplo, "Enero"). Utilice la fecha de entrada para decidir a qué mes pertenece cada reserva. (2 puntos)

```
def servicios_estrella_por_mes(reservas: List[Reserva],
                       tipos_habitacion: Optional[Set[str]] = None) -> Dict[str, str]
```

```
Test servicios_estrella_por_mes
Todos los tipos de habitación:
         ('Enero', 'Parking')
         ('Febrero', 'Gimnasio')
         ('Marzo', 'Parking')
         ('Abril', 'Gimnasio')
         ('Mayo', 'Gimnasio')
         ('Junio', 'Parking')
         ('Julio', 'Gimnasio')
         ('Agosto', 'Gimnasio')
         ('Septiembre', 'Piscina')
         ('Octubre', 'Spa')
         ('Noviembre', 'Gimnasio')
         ('Diciembre', 'Parking')

Habitación familiar o deluxe:
         ('Enero', 'Gimnasio')
         ('Febrero', 'Gimnasio')
         ('Marzo', 'Gimnasio')
         ('Abril', 'Gimnasio')
         ('Mayo', 'Gimnasio')
         ('Junio', 'Spa')
         ('Julio', 'Gimnasio')
         ('Agosto', 'Parking')
         ('Septiembre', 'Piscina')
         ('Octubre', 'Gimnasio')
         ('Noviembre', 'Gimnasio')
         ('Diciembre', 'Gimnasio')
```

6. **media_dias_entre_reservas**: recibe una lista de reservas, y devuelve la media de días que transcurren entre cada dos reservas consecutivas en el tiempo. Utilice la fecha de entrada de las reservas para calcular los días entre reservas. Tenga en cuenta que la lista de reservas recibida no tiene por qué estar ordenada cronológicamente. (1,5 puntos)

```
def media_dias_entre_reservas(reservas: List[Reserva]) -> float
```

```
Test media_dias_entre_reservas
0.7353535353535353
```

7. **cliente_reservas_mas_seguidas**: recibe una lista de reservas y un entero `min_reservas`, y devuelve el dni y la media de días entre reservas consecutivas del cliente que, teniendo al menos `min_reservas` en la lista, tiene la menor media de días entre reservas consecutivas. (2 puntos)

```
def cliente_reservas_mas_seguidas(reservas: List[Reserva],
                                  min_reservas: int
                                  ) -> Tuple[str, float]
```

```
Test cliente_reservas_mas_seguidas
El DNI del cliente con al menos 5 reservas y menor media de días entre reservas consecutivas
es 88681493W, con una media de días entre reservas de 9.75.
```