Una aplicación de gestión de gastos compartidos permite registrar los pagos realizados por los distintos miembros de un grupo de amigos o conocidos durante un viaje o cualquier otro evento, facilitando en un momento dado calcular el dinero gastado por cada uno y saldar las deudas de unos con otros.

La aplicación almacena la información sobre los pagos realizados por un grupo determinado en un único archivo en formato CSV codificado en UTF-8. Cada línea del archivo recoge la siguiente información: un número que indica el orden en que se registraron los gastos (de tipo int); el nombre del usuario que pagó el gasto (de tipo str); el concepto en el que se gastó el dinero (de tipo str); el destinatario del gasto (de tipo str); el importe del gasto (de tipo float); y la fecha (de tipo date). Las primeras líneas son las que se muestran a continuación:

```
Num. Gasto, Pagador, Concepto, Destinatario, Importe, Fecha
1, Inés, Restaurante, Todos, 20.6, 04/04/2019
2, Inés, Restaurante, Todos, 111.84, 04/04/2019
3, Miguel, Teatro, Todos, 126.85, 04/04/2019
…
9, Guille, Restaurante, Inés, 59.80, 05/04/2019
```

El destinatario del gasto indica quién se ha beneficiado del concepto pagado, pudiendo ser un gasto que sólo afecta a dos usuarios (el que paga y el que aparece en el campo destinatario); o bien un gasto aplicable a todo el grupo (si aparece "Todos" en el campo destinatario). Por ejemplo, "9, Guille, Restaurante, Inés, 59.80, 05/04/2019" indica que Guille pagó un restaurante para Inés y para él mismo, y que el importe de la comida para los dos fue de 59,80; sin embargo, el gasto "2, Inés, Restaurante, Todos, 111.84, 04/04/2019" indica que Inés pagó 111,84 en un restaurante por una comida para todos los miembros del grupo (incluyendo a ella misma).

El objetivo del ejercicio es leer estos datos, realizar distintas operaciones con ellos e implementar los tests que permitan probarlas. Cada operación se implementará en una función distinta. Se pide implementar las siguientes funciones y sus tests correspondientes, teniendo en cuenta que se pueden definir funciones auxiliares cuando se considere necesario:

1. **lee_gastos**: lee un fichero de entrada en formato CSV y devuelve una lista de tuplas de tipo Gasto conteniendo todos los datos almacenados en el fichero. (1 punto)
2. **pagadores_y_conceptos**: recibe una lista de tuplas de tipo Gasto y devuelve una lista con todos los pagadores y otra con todos los conceptos. Ambas listas deben estar ordenadas alfabéticamente y sin duplicados. (1 punto)
3. **total_importe**: recibe una lista de tuplas de tipo Gasto y dos fechas, una inicial y otra final, y devuelve el importe total gastado entre esas dos fechas. Si alguna de las fechas recibidas es None no se acotarán las fechas por el extremo correspondiente. Tenga en cuenta que las fechas recibidas serán cadenas con el mismo formato de las contenidas en el fichero CSV. (1,5 puntos)
4. **conceptos_menos_gastos**: recibe una lista de tuplas de tipo Gasto y devuelve una lista con los conceptos en los que se ha realizado un menor número de gastos. Tenga en cuenta que se buscan los conceptos para los que hay registrados menos transacciones, no los que acumulan una cantidad menor de dinero gastado. Si hay más de un concepto con el mínimo número de gastos, todos ellos deben aparecer en la lista devuelta. (1,5 puntos)
5. **pagadores_mayor_importe_medio**: recibe una lista de tuplas de tipo Gasto y un número entero n, y devuelve una lista de tuplas con los n pagadores con mayor importe medio por gasto, junto con el valor de este importe medio. (2 puntos)

6. **balance**: recibe una lista de tuplas de tipo Gasto y devuelve un diccionario con el balance por pagador. El balance de un pagador consiste en una cantidad que indica el dinero que debe recibir (si el número es positivo) o pagar (si el número es negativo) para saldar todos los gastos registrados.

   Comience con un diccionario en el que todos los pagadores tengan un balance igual a 0, y vaya actualizando el balance gasto a gasto. Para cada gasto, debe sumarse el importe total al balance del pagador. Además, debe restarse al balance de todos los beneficiarios de ese gasto el importe por persona correspondiente. Tenga en cuenta que los beneficiarios de un gasto pueden ser o bien todos los usuarios (si aparece "Todos" en el campo destinatario), o bien el usuario que realiza el pago y el que figura como destinatario. Redondee los balances de cada usuario antes de devolver el diccionario para que las cantidades tengan dos decimales. (2 puntos)
7. Complete el código de las funciones de test del módulo gastos_TEST.py para probar las funciones anteriores, teniendo en cuenta que la salida por consola de la ejecución de los test debe ser la siguiente. (1 punto)

```
EJERCICIO 1
Número de registros leídos: 40
Primer registro: Gasto(num_gasto=1, usuario='Inés', concepto='Restaurante',
destinatario='Todos', importe=20.6, fecha=datetime.date(2019, 4, 4))

EJERCICIO 2
Pagadores: ['Antonio', 'Ester', 'Guille', 'Inés', 'Jimena', 'Laura', 'María', 'Miguel',
'Rodolfo']
Conceptos: ['Bar', 'Bus', 'Cafetería', 'Cena', 'Cine', 'Golosinas', 'Museo', 'Restaurante',
'Taxi', 'Teatro', 'Tren', 'Varios']

EJERCICIO 3
La cantidad total gastada entre el 5 y el 8 de abril de 2019 fue: 1128.6
La cantidad total gastada fue: 1486.27

EJERCICIO 4
Los conceptos con menos gastos registrados son: ['Cena', 'Varios', 'Cine']

EJERCICIO 5
Los tres pagadores con un mayor importe medio en sus gastos son:
[('Miguel', 62.4675), ('Inés', 47.47), ('Antonio', 47.095)]

EJERCICIO 6
El balance final de gastos es el siguiente (se muestran los usuarios por orden alfabético):
Antonio -> -43.47
Ester -> -38.35
Guille -> -111.09
Inés -> 57.69
Jimena -> 36.2
Laura -> -60.52
María -> -59.42
Miguel -> 134.39
Rodolfo -> 84.57
```