FUNDAMENTOS DE PROGRAMACIÓN. Curso 2022/23

PRIMER EXAMEN PARCIAL. 19 de enero de 2023

Se tienen datos sobre un conjunto de películas estrenadas entre 2001 y 2020. Los datos tienen esta forma en un fichero csv (se muestra una única línea del fichero):

```
19/12/2019;Star Wars: The Rise of Skywalker;J.J. Abrams;Acción, Aventura,
Fantasía;142;275;1074;Daisy Ridley, Adam Driver, John Boyega
```

La información de cada línea se corresponde con lo siguiente:

- fecha de estreno: fecha en la que se estrenó la película, de tipo date.
- título: título de la película, de tipo str.
- director: nombre del director, de tipo str.
- géneros: géneros de la película, separados por comas, de tipo list de str. Fíjese en que pueden aparecer espacios después de las comas.
- duración: duración de la película en minutos, de tipo int.
- presupuesto: presupuesto de producción de la película en millones de dólares, de tipo int.
- recaudación: recaudación de la película en millones de dólares, de tipo int.
- reparto: actores y actrices principales de la película, separados por comas, de tipo list de str. Fíjese en que pueden aparecer espacios después de las comas.

Por ejemplo, en la línea mostrada anteriormente, vemos que la película "Star Wars: The Rise of Skywalker" fue estrenada el 19 de diciembre de 2019, estuvo dirigida por J.J. Abrams, sus géneros son "Acción", "Aventura" y "Fantasía", dura 142 minutos, su presupuesto fue de 275 millones de dólares, recaudó 1074 millones de dólares y su reparto lo encabezan "Daisy Ridley", "Adam Driver" y "John Boyega".

Usaremos esta definición de namedtuple:

```
Pelicula = namedtuple("Pelicula", "fecha_estreno, titulo, director, generos, duracion, presupuesto, recaudacion, reparto")
```

Como ejemplo, para la película anterior obtendremos la siguiente tupla (fíjese en el tipo de cada uno de los campos, y en que los espacios detrás de las comas de los géneros y el reparto han sido eliminados):

```
Pelicula(fecha_estreno=datetime.date(2019, 12, 19), titulo='Star Wars: The Rise of Skywalker', director='J.J. Abrams', generos=['Acción', 'Aventura', 'Fantasía'], duracion=142, presupuesto=275, recaudacion=1074, reparto=['Daisy Ridley', 'Adam Driver', 'John Boyega'])
```

Implemente las siguientes funciones en un módulo peliculas.py:

1. **lee_peliculas**: recibe una cadena de texto con la ruta de un fichero csv, y devuelve una lista de tuplas `Pelicula` con la información contenida en el fichero. Utilice `datetime.strptime(cadena, "%d/%m/%Y").date()` para parsear las fechas. (1 punto)

2. **pelicula_mas_ganancias**: recibe una lista de tuplas de tipo `Pelicula` y una cadena de texto `genero`, con valor por defecto `None`, y devuelve el título y las ganancias de la película con mayores ganancias, de entre aquellas películas que tienen entre sus géneros el `genero` indicado. Si el parámetro `genero` es `None`, se busca la película con mayores ganancias, sin importar sus géneros. Las ganancias de una película se calculan como la diferencia entre la recaudación y el presupuesto. (1 punto)

3. **media_presupuesto_por_genero**: recibe una lista de tuplas de tipo `Pelicula` y devuelve un diccionario en el que las claves son los distintos géneros y los valores son la media de presupuesto de las películas de cada género. (1,5 puntos)

4. **peliculas_por_actor**: recibe una lista de tuplas de tipo `Pelicula` y dos enteros `año_inicial` y `año_final`, con valor por defecto `None`, y devuelve un diccionario en el que las claves son los nombres de los actores y actrices, y los valores son el número de películas, estrenadas entre `año_inicial` y `año_final` (ambos incluidos), en que ha participado cada actor o actriz. Si `año_inicial` o `año_final` son `None`, se contarán las películas sin filtrar por año inicial o final, respectivamente. (1,5 puntos)

5. **actores_mas_frecuentes**: recibe una lista de tuplas de tipo `Pelicula`, un entero `n` y dos enteros `año_inicial` y `año_final`, con valor por defecto `None`, y devuelve una lista con los `n` actores o actrices que han participado en más películas estrenadas entre `año_inicial` y `año_final` (ambos incluidos). La lista de actores o actrices debe estar ordenada alfabéticamente. Si `año_inicial` o `año_final` son `None`, se contarán las películas sin filtrar por año inicial o final, respectivamente. Haga uso de la función `peliculas_por_actor` para implementar esta función. (1 punto)

6. **recaudacion_total_por_año**: recibe una lista de tuplas de tipo `Pelicula` y un conjunto de cadenas de texto `generos`, con valor por defecto `None`, y devuelve un diccionario en el que las claves son los años en los que se han estrenado películas, y los valores son la recaudación total de las películas estrenadas en cada año que son de alguno de los géneros contenidos en el parámetro `generos`. Si `generos` es `None`, se calcularán las recaudaciones totales de todas las películas de cada año, independientemente de su género. NOTA: Puede usar operaciones entre conjuntos para ver si existe alguna coincidencia entre los géneros de cada película y los géneros especificados por el parámetro. (1,5 puntos)

7. **incrementos_recaudacion_por_año**: recibe una lista de tuplas de tipo `Pelicula` y un conjunto de cadenas de texto `generos`, con valor por defecto `None`, y devuelve una lista de enteros con las diferencias de recaudación total de cada año con respecto al anterior registrado, de películas de alguno de los géneros indicados por el parámetro `generos`. Si `generos` es `None`, se usarán para el cálculo las recaudaciones totales de todas las películas de cada año, independientemente de su género. Haga uso de la función `recaudacion_total_por_año` para implementar esta función. (1,5 puntos)

8. Pruebe las funciones implementadas en un módulo peliculas_test.py. Se recomienda que lo vaya haciendo a medida que vaya resolviendo los distintos apartados. (1 punto)

Anexo: Pruebas de las funciones.

En esta sección, se muestra una posible ejecución de las pruebas de las funciones.

```
Test de lee_peliculas:
Total registros leídos: 36
Mostrando los tres primeros registros:
        Pelicula(fecha_estreno=datetime.date(2019, 12, 19), titulo='Star Wars: The
Rise of Skywalker', director='J.J. Abrams', generos=['Acción', 'Aventura',
'Fantasía'], duracion=142, presupuesto=275, recaudacion=1074, reparto=['Daisy Ridley',
'Adam Driver', 'John Boyega'])
        Pelicula(fecha_estreno=datetime.date(2019, 5, 23), titulo='Joker',
director='Todd Phillips', generos=['Drama'], duracion=122, presupuesto=55,
recaudacion=980, reparto=['Joaquin Phoenix', 'Robert De Niro', 'Zazie Beetz'])
```

```
        Pelicula(fecha_estreno=datetime.date(2019, 11, 22), titulo='Frozen II',
director='Chris Buck', generos=['Animación', 'Aventura', 'Comedia'], duracion=103,
presupuesto=150, recaudacion=1450, reparto=['Idina Menzel', 'Kristen Bell', 'Jonathan
Groff'])

Test de pelicula_mas_ganancias (genero=None):
('Avatar', 2552)

Test de pelicula_mas_ganancias (genero='Drama'):
('Joker', 925)

Test de media_presupuesto_por_genero:
{'Acción': 190.52631578947367, 'Aventura': 189.40740740740742, 'Fantasía':
209.46153846153845, 'Drama': 80.0, 'Animación': 140.16666666666666, 'Comedia':
140.14285714285714, 'Thriller': 103.5, 'Musical': 52.0, 'Romance': 73.5, 'Crimen':
208.33333333333334, 'Sci-Fi': 149.6, 'Misterio': 40.0, 'Familia': 76.0}

Test de peliculas_por_actor (año_inicial=None, año_final=None):
(sólo se mostrará el número de películas de tres actores concretos)
Robert Downey Jr.: 3
Christian Bale: 3
Adam Driver: 2

Test de peliculas_por_actor (año_inicial=2010, año_final=2020):
(sólo se mostrará el número de películas de tres actores concretos)
Robert Downey Jr.: 2
Christian Bale: 1
Adam Driver: 2

Test de actores_mas_frecuentes (n=3, año_inicial=2005, año_final=2015):
['Christian Bale', 'Kristen Stewart', 'Leonardo DiCaprio']

Test de recaudacion_total_por_año (generos=None):
{2019: 6302, 2020: 254, 2018: 1278, 2017: 1990, 2016: 678, 2015: 2035, 2014: 1633,
2013: 1362, 2012: 2189, 2011: 1754, 2010: 2515, 2009: 3520, 2008: 1981, 2007: 585,
2006: 109, 2005: 897, 2004: 919, 2003: 1142, 2002: 1034, 2001: 361}

Test de recaudacion_total_por_año (generos={'Drama', 'Acción'}):
{2019: 4852, 2020: 254, 2018: 1278, 2017: 1990, 2016: 678, 2015: 2035, 2014: 1633,
2013: 392, 2012: 2189, 2011: 709, 2010: 1452, 2008: 1981, 2007: 585, 2006: 109, 2002:
1034, 2001: 361, 2009: 2789}

Test de incrementos_recaudacion_por_año (generos=None):
[673, 108, -223, -22, -788, 476, 1396, 1539, -1005, -761, 435, -827, 271, 402, -1357,
1312, -712, 5024, -6048]

Test de incrementos_recaudacion_por_año (generos={'Drama', 'Acción'}):
```

```
[673, -925, 476, 1396, 808, -1337, -743, 1480, -1797, 1241, 402, -1357, 1312, -712,
3574, -4598]
```