Fundamentos de Programación

Curso 2024-25 - Primer examen parcial, 15 de enero de 2025

Suscripciones

Se quieren analizar los datos de suscripciones de un servicio de *streaming*. 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 inicio y fin de la suscripción (la fecha de fin puede estar vacía, en cuyo caso se supone que la suscripción sigue activa), el tipo de plan contratado, el número de perfiles creados por el usuario, el precio mensual y las características extra o *addons* contratadas (puede no haber ninguna). Las primeras líneas son las que se muestran a continuación:

```
nombre,dni,fecha_inicio,fecha_fin,tipo_plan,num_perfiles,precio_mensual,addons
Carlos Martínez,65329814W,2022-01-02,,Premium,1,25.00,"4K HDR,UltraHD"
Ana Pérez,91234762R,2022-01-05,2024-05-10,Estandar,2,15.99,"Descargas ilimitadas,Sonido Dolby"
María Gómez,84736291F,2022-01-08,2024-03-20,Premium,3,25.00,
```

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

```python
from typing import NamedTuple

Suscripcion = NamedTuple("Suscripcion",
                         [("nombre", str),
                          ("dni", str),
                          ("fecha_inicio", date),
                          ("fecha_fin", date | None), # Será None si la suscripción sigue activa
                          ("tipo_plan", str),
                          ("num_perfiles", int),
                          ("precio_mensual", float),
                          ("addons", list[str])
                         ])
```

Observaciones

- Si una suscripción sigue activa (fecha fin está vacía en el CSV) se almacenará el valor `None` en la tupla correspondiente.
- Si una suscripción no tiene *addons*, se almacenará una lista vacía en la tupla correspondiente.
- Tenga en cuenta que si lee el csv con `csv.reader`, los campos que aparecen entrecomillados en el CSV se obtendrán sin las comillas.
- La cadena de formato para parsear las fechas es `"%Y-%m-%d"`.
- La duración en días de una suscripción se calcula como `(fecha_fin - fecha_inicio).days`. Para aquellas suscripciones que siguen activas, se utilizará la fecha actual como fecha fin.
- Para calcular el importe total de una suscripción, se considera un precio diario igual al precio mensual dividido entre 30. El importe total se calculará multiplicando el precio diario por el total de días de la suscripción.

Estructura de las carpetas del proyecto

- **/src**: Contiene los diferentes módulos de Python que conforman el proyecto.
  - **suscripciones.py**: Contiene funciones para explotar los datos de suscripciones.
  - **suscripciones_test.py**: Contiene funciones de test para probar las funciones del módulo `suscripciones.py`. En este módulo está el main.
  - Puede añadir otros módulos para funciones auxiliares si así lo desea.
- **/data**: Contiene el dataset del proyecto.
  - **suscripciones.csv**: Archivo con las suscripciones de un servicio de *streaming*.
- **/doc**: Contiene la documentación del proyecto.
  - **identificacion.md**: Archivo con los datos del alumno que realiza el examen.
  - **FP2425-Parcial1-Sesión 3-Enunciado.pdf**: Enunciado del examen.

Ejercicios

Se pide implementar las siguientes funciones (en el módulo `suscripciones.py`) y sus tests correspondientes (en el módulo `suscripciones_test.py`). Las puntuaciones indicadas para cada ejercicio incluyen la realización de dichos tests. Tenga en cuenta que se pueden definir funciones auxiliares cuando se considere necesario.

Para cada función solicitada, se le proporciona el prototipo de la función (donde puede ver los parámetros, sus tipos y el tipo de salida), así como una posible salida del test correspondiente.

Ejercicio 1 *(1 punto)*

```
def lee_suscripciones(ruta_fichero: str) -> list[Suscripcion]
```

Recibe la ruta de un fichero CSV y devuelve una lista de tuplas de tipo `Suscripcion` conteniendo todos los datos almacenados en el fichero.

```
Test lee_suscripciones
Total suscripciones: 50
Las tres primeras:
        Suscripcion(nombre='Carlos Martínez', dni='65329814W',
fecha_inicio=datetime.date(2022, 1, 2), fecha_fin=None, tipo_plan='Premium',
num_perfiles=1, precio_mensual=25.0, addons=['4K HDR', 'UltraHD'])
        Suscripcion(nombre='Ana Pérez', dni='91234762R', fecha_inicio=datetime.date(2022,
1, 5), fecha_fin=datetime.date(2022, 5, 10), tipo_plan='Estandar', num_perfiles=2,
precio_mensual=15.99, addons=['Descargas ilimitadas', 'Sonido Dolby'])
        Suscripcion(nombre='María Gómez', dni='84736291F',
fecha_inicio=datetime.date(2022, 1, 8), fecha_fin=datetime.date(2022, 3, 20),
tipo_plan='Premium', num_perfiles=3, precio_mensual=25.0, addons=['4K HDR', 'Cine en
casa'])
```

Ejercicio 2 *(1,5 puntos)*

```
def suscripciones_mas_rentables(suscripciones: list[Suscripcion],
                    n: int = 3,
                    tipos_plan: set[str]|None = None) -> list[tuple[str, float]]
```

Devuelve las `n` suscripciones más rentables (las que acumulen un mayor importe total), de entre las que sean de alguno de los tipos de plan indicados por `tipos_plan`. Si `tipos_plan` es `None`, se usarán todas las suscripciones para calcular el resultado. Para cada suscripción, se devuelve una tupla con el dni y el importe total de la suscripción. Consulte en la sección *Observaciones* cómo se calcula el importe total de una suscripción.

```
Test suscripciones_mas_rentables
Las 3 suscripciones más rentables (sin filtrar):
        ('65329814W', 924.1666666666667)
        ('15379248E', 715.0)
        ('20594831K', 684.1666666666667)

Filtrando por tipo de plan ('Básico'):
        ('10293847A', 363.303)
        ('94127530D', 351.315)
        ('70249831H', 267.06600000000003)
```

Ejercicio 3 *(1,5 puntos)*

```
def plan_mas_perfiles(suscripciones: list[Suscripcion],
                      fecha_ini: date|None = None,
                      fecha_fin: date|None = None) -> tuple[str, int]
```

Devuelve una tupla con el tipo de plan y el total de perfiles correspondiente al plan que suma mayor número total de perfiles entre todas las suscripciones cuya `fecha_inicio` esté dentro del rango indicado por los parámetros `fecha_ini` y `fecha_fin`. Ambas fechas `fecha_ini` y `fecha_fin` están incluidas en el rango. Si `fecha_ini` es `None` no se establece un límite inferior; si `fecha_fin` es `None` no se establece un límite superior.

```
Test plan_mas_perfiles
Sin filtrar fechas, el plan con más perfiles acumulados es: Estandar con 42
Entre el 1 de febrero de 2023 y el 31 de marzo de 2023: Premium con 18
```

Ejercicio 4 *(2 puntos)*

```
def media_dias_por_plan(suscripciones: list[Suscripcion]) -> dict[str, float]
```

Calcula la duración media (en días) de las suscripciones finalizadas (aquellas con `fecha_fin` no `None`) agrupadas por tipo de plan. Devuelve un diccionario en el que las claves son los tipos de plan y los valores la media de días.

```
Test media_dias_por_plan
Duración media (en días) de suscripciones finalizadas por tipo de plan:
{'Estandar': 400.57142857142856, 'Premium': 503.3076923076923, 'Básico': 379.25}
```

Ejercicio 5 *(2 puntos)*

```
def addon_mas_popular_por_año(suscripciones: list[Suscripcion]) -> dict[int,str]
```

Devuelve un diccionario que hace corresponder a cada año de inicio de suscripción el *addon* que aparece en más suscripciones ese año.

```
Test addon_mas_popular_por_año
{2022: '4K HDR', 2023: 'Sonido Dolby', 2024: 'UltraHD'}
```

Ejercicio 6 *(2 puntos)*

```
def evolucion_años(suscripciones: list[Suscripcion]) -> list[tuple[int, int]]
```

Calcula la variación anual (incrementos o decrementos) en el número total de suscripciones, devolviendo una lista de tuplas con el año y la diferencia respecto al año anterior. La lista estará ordenada cronológicamente por año.

Para ello, considere lo siguiente:

- Cuando una suscripción se inicia en un año determinado, se suma una unidad al total de suscripciones de ese año.
- Cuando una suscripción finaliza en un año determinado, se resta una unidad del total de suscripciones de ese año.
- Si una suscripción se inicia y finaliza dentro del mismo año, su efecto se anula y no se contabiliza en la variación.

```
Test evolucion_años
Evolución de suscripciones por año:
[(2023, 3), (2024, -46)]
```