Desenvolva um módulo em Python designado por **epp5.py**. Neste módulo deverá definir as funções descritas abaixo. Não deverá preocupar-se em efetuar a geração/tratamento de exceções quando tal não lhe seja explicitamente pedido.

Uma transação é representada por um dicionário com 4 pares chave/valor:

* "preço unitário": número real positivo ou negativo que representa o preço unitário do produto vendido/comprado;
* "quantidade": número inteiro positivo que representa a quantidade de produtos vendidos/comprados;
* "produto": cadeia de caracteres arbitrária que corresponde ao produto vendido/comprado;
* "data": um tuplo com 3 inteiros correspondentes ao dia, mês e ano que representa a data em que a transação foi efetuada.

O valor total de uma transação corresponde à multiplicação do preço unitário pela quantidade de produtos vendidos/comprados.

Exemplo:

{"preço unitário": 1.0, "quantidade": 4, "produto": "garrafa de água", "data": (1, 1, 2024)}

As transações podem ser agrupadas em tópicos. Cada tópico pode ter sub-tópicos. Um tópico é representado por um dicionário com 2 pares chave/valor:

* "tópico": cadeia de caracteres arbitrária que corresponde ao nome do tópico;
* "transações": lista em que os elementos podem ser uma transação ou outro tópico

Exemplo:

{"tópico": "Garrafas", "transações": [{"preço unitário": 1.0, "quantidade": 4, "produto": "garrafa de água", "data": (1, 1, 2024)}, {"tópico": "Refrigerantes", "transações": [{"preço unitário": -2.5, "quantidade": 3, "produto": "Fanta", "data": (1, 2, 2024)}]}]}

1. Função **balanco\_topico [6 valores]**

Escreva uma função **recursiva**em Python, chamada **balanco\_topico**, que recebe como argumento um tópico e devolve um número real correspondente à soma do valor total de todas as transações presentes nesse tópico, incluíndo as transações dos sub-tópicos. Note que cada sub-tópico pode também conter os seus próprios sub-tópicos. Para identificar se um elemento é um sub-tópico, pode verificar se contém a chave "tópico". Para identificar se um elemento é uma transação, pode verificar se contém a chave "produto".

Exemplos

**>>> balanco\_topico({"tópico": "Garrafas", "transações": [{"preço unitário": 1.0, "quantidade": 4, "produto": "garrafa de água", "data": (1, 1, 2024)}, {"tópico": "Refrigerantes", "transações": [{"preço unitário": -2.5, "quantidade": 3, "produto": "Fanta", "data": (1, 2, 2024)}]}]})**

**-3.5**

**>>> balanco\_topico({"tópico": "Garrafas", "transações": [{"preço unitário": 1.0, "quantidade": 4, "produto": "garrafa de água", "data": (1, 1, 2024)}, {"tópico": "Refrigerantes", "transações": [{"preço unitário": -2.5, "quantidade": 3, "produto": "Fanta", "data": (1, 2, 2024)}, {"tópico": "Sem açúcar", "transações": [{"preço unitário": 2, "quantidade": 2, "produto": "Coca-cola Zero", "data": (1, 1, 2024)}]}]}]})**

**0.5**

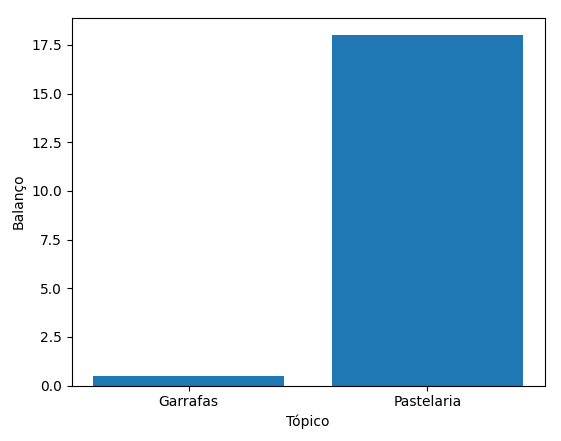
2. Função**balanco\_mensal [4 valores]**

Escreva uma função **recursiva** em Python, chamada **balanco\_mensal**, que recebe como argumento uma lista de tópicos, um número inteiro positivo correspondente ao mês e um número inteiro positivo correspondente ao ano e devolve a soma do valor total de todas as transações efetuadas nesse mês e ano e pertencentes aos tópicos presentes na lista, incluíndo os sub-tópicos de cada tópico.

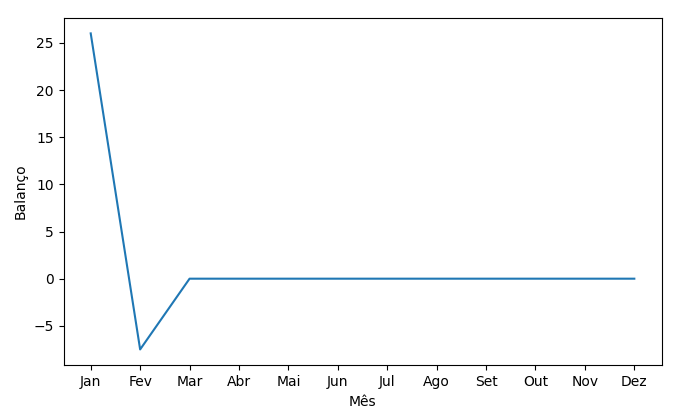
Exemplos  
  
**>>> balanco\_mensal([{"tópico": "Garrafas", "transações": [{"preço unitário": 1.0, "quantidade": 4, "produto": "garrafa de água", "data": (1, 1, 2024)}, {"tópico": "Refrigerantes", "transações": [{"preço unitário": -2.5, "quantidade": 3, "produto": "Fanta", "data": (1, 2, 2024)}, {"tópico": "Sem açúcar", "transações": [{"preço unitário": 2, "quantidade": 2, "produto": "Coca-cola Zero", "data": (1, 1, 2024)}]}]}]}, {"tópico": "Pastelaria", "transações": [{"preço unitário": 3, "quantidade": 6, "produto": "Pastel de Belém", "data": (7, 1, 2024)}]}], 1, 2024)  
26.0**

**>>> balanco\_mensal([{"tópico": "Garrafas", "transações": [{"preço unitário": 1.0, "quantidade": 4, "produto": "garrafa de água", "data": (1, 1, 2024)}, {"tópico": "Refrigerantes", "transações": [{"preço unitário": -2.5, "quantidade": 3, "produto": "Fanta", "data": (1, 2, 2024)}, {"tópico": "Sem açúcar", "transações": [{"preço unitário": 2, "quantidade": 2, "produto": "Coca-cola Zero", "data": (1, 1, 2024)}]}]}]}, {"tópico": "Pastelaria", "transações": [{"preço unitário": 3, "quantidade": 6, "produto": "Pastel de Belém", "data": (7, 1, 2024)}]}], 2, 2024)  
-7.5**

3. Função**desenha\_balanco\_por\_topico [5 valores]**

Escreva uma função em Python, chamada **desenha\_balanco\_por\_topico**,que recebe como argumento uma lista de tópicos e que, recorrendo à biblioteca Matplotlib,deverá criar e mostrar um gráfico de barras (*bar chart*) com a representação do balanço de cada tópico (*não incluíndo sub-tópicos*).  
  
Exemplo  
  
**>>> desenha\_balanco\_por\_topico([{"tópico": "Garrafas", "transações": [{"preço unitário": 1.0, "quantidade": 4, "produto": "garrafa de água", "data": (1, 1, 2024)}, {"tópico": "Refrigerantes", "transações": [{"preço unitário": -2.5, "quantidade": 3, "produto": "Fanta", "data": (1, 2, 2024)}, {"tópico": "Sem açúcar", "transações": [{"preço unitário": 2, "quantidade": 2, "produto": "Coca-cola Zero", "data": (1, 1, 2024)}]}]}]}, {"tópico": "Pastelaria", "transações": [{"preço unitário": 3, "quantidade": 6, "produto": "Pastel de Belém", "data": (7, 1, 2024)}]}])**  


4. Função**desenha\_balanco\_anual [5 valores]**  
Escreva uma função em Python, chamada **desenha\_balanco\_atual**, que recebe como argumento uma lista de tópicos e um número inteiro positivo correspondente a um ano. Recorrendo à biblioteca Matplotlib, esta função deverá criar e mostrar um gráfico de linha (*line chart*) com a representação do balanço de todos os meses do ano em consideração de acordo com a lista de tópicos fornecida.Exemplo  
 **>>> desenha\_balanco\_anual([{"tópico": "Garrafas", "transações": [{"preço unitário": 1.0, "quantidade": 4, "produto": "garrafa de água", "data": (1, 1, 2024)}, {"tópico": "Refrigerantes", "transações": [{"preço unitário": -2.5, "quantidade": 3, "produto": "Fanta", "data": (1, 2, 2024)}, {"tópico": "Sem açúcar", "transações": [{"preço unitário": 2, "quantidade": 2, "produto": "Coca-cola Zero", "data": (1, 1, 2024)}]}]}]}, {"tópico": "Pastelaria", "transações": [{"preço unitário": 3, "quantidade": 6, "produto": "Pastel de Belém", "data": (7, 1, 2024)}]}], 2024)**



Notas Importantes

* **Caso tenha dificuldade em instalar a biblioteca Matplotlib no seu ambiente de desenvolvimento local, poderá inserir uma nova célula no notebook disponibilizado no JupyterHub e utilizá-la para efetuar o desenvolvimento e teste das suas funções e depois copiar o respetivo código para o ficheiro a submeter.**
* O seu trabalho não deve conter nenhuma instrução **print** nem **input**. As suas funções devem devolver valores e não escrevê-los. Assim, não deve utilizar instruções **print**, mas sim instruções **return**.
* Os valores apresentados nos exemplos são apenas exemplificativos: as suas funções devem aceitar quaisquer valores para os respetivos parâmetros, dentro do tipo e gama especificados.
* O programa pedido deverá ser submetido num ficheiro com o nome **epp5.py**. Na primeira linha do ficheiro deverá constar, em comentário, o seu nome e número de aluno.

Critérios de Avaliação

Cada uma das funções será classificada individualmente, para a cotação indicada, de acordo com os seguintes critérios.

* Resultados corretos: 70%
  + *Se a função consegue produzir o resultado esperado, incluíndo, caso seja pedido, se consegue produzir as exceções corretas para parâmetros de entrada inválidos.*
* Qualidade do código: 30%
  + *Se o código da função está corretamente comentado (não em excesso), se está bem estruturado e se é de fácil compreensão.*