Exercício Prático - Orientação a Objetos

O notebook Jupyter deste exercício pode ser acessado clicando aqui.

1) Essa lista de exercícios terá como base a classe `Evento` criada em exercícios anteriores. Primeiramente criaremos a classe abstrata `EventoABC` com os métodos de instância abstratos `__str__(self)` e `isConcluido(self)`, indicando que todos as subclasses que dela herdarem devem implementar esses métodos.

   `EventoABC` também possui os atributos `_titulo` (string) e `_descricao` (string), cujos valores são recebidos e inicializados no construtor da classe. Note a convenção de nomenclatura indicando o caráter privado desses atributos.

2) Crie a classe `DataHora` que dará suporte ao registro de eventos de calendário.

   - A classe possui o atributo de instância `_data_hora` (datetime) privado e um atributo de classe `FORMAT` inicializado com a formatação de string aceito para `_data_hora`, ou seja, `FORMAT = '%d/%m/%Y, %H:%M'`.
   - A classe **não possui construtor customizado**. A alteração de seu atributo se dará a partir da propriedade a seguir.
   - Crie a `property` `data_hora` para manipular o atributo `_data_hora`.
     - O getter da propriedade deve retornar a data como uma string formatada (`%d/%m/%Y, %H:%M`). Use o atributo `FORMAT`. Consulte o funcionamento do método `strftime`.
     - O setter da propriedade deve receber uma string de data formatada (`%d/%m/%Y, %H:%M`) e implementar um bloco try-except que tenta converter a string em `datetime` e lança um `ValueError` caso a entrada seja inválida. Use o atributo `FORMAT`. Consulte o funcionamento do método `strptime`.
   - Crie o método de instância `isPassado(self)` que avalia se a `_data_hora` é menor que `datetime.now()` (a data e hora atual) e retorna `True` em caso positivo, e `False` caso contrário.
   - Crie o método de instância `somaDias(self, num_dias)` que recebe um inteiro `num_dias`, soma esse valor ao atributo interno `_data_hora` e retorna a string formatada do resultado da soma (código dado a seguir).

```
data_hora_somada = self._data_hora + datetime.timedelta(days=num_dias)
return data_hora_somada.strftime(FORMAT)
```

Teste a classe `DataHora` com o seguinte código (altere o que for necessário):

```
# instanciando o objeto
dh = DataHora()

# definindo a data_hora através da propriedade
dh.data_hora = '05/02/2024, 12:30'

## editando a data_hora através da função somaDias
dh.data_hora = dh.somaDias(30)

## imprimindo a data_hora editada e se é passado
print(dh.data_hora, dh.isPassado())
```

3) Crie a classe `EventoUnico`:
   - A classe deve herdar de `EventoABC`.
   - Possui o atributo de instância `_data_hora` (classe `DataHora` que criamos previamente).
   - Seu construtor deve receber e inicializar os atributos da superclasse, além do valor de `_data_hora` recebido como uma string formatada (`%d/%m/%Y, %H:%M`). Note que para alterar `_data_hora` (objeto tipo `DataHora`), você deve manipular a propriedade interna da classe.
   - Implementa os métodos abstratos da superclasse:
     - Método `isConcluido()` que invoca o método `isPassado()` de `_data_hora` e retorna o seu resultado.
     - Método `__str__` que imprime os atributos do evento na forma `"Evento: _titulo, Data: _data_hora, Descrição: _descricao, Concluido: isConcluido()"`. Note que `isConcluido()` é o método de avaliação implementado.
   - Crie o método de instância `editar_data_hora` que recebe uma string formatada e altera `_data_hora` (através de sua propriedade interna).

Teste a classe `EventoUnico` com o seguinte código:

```
# criar evento
evento = EventoUnico('Reunião',
```

```
            'Sala 302, prédio da esquina', '05/10/2023, 16:30')
print(evento)

# editar data do evento (através da propriedade)
evento.editar_data_hora('05/10/2024, 16:30')
print(evento)
```

4) Crie a classe `EventoRecorrente`:
   - A classe deve herdar de `EventoABC`.
   - Possui como atributo próprio uma lista privada de objetos `DataHora` (como você deve nomear o atributo?).
   - Seu construtor recebe os atributos da superclasse, além dos atributos `data_hora_inicial` (string formatada), `data_hora_final`(string formatada) e `intervalo_repeticao` (int), sendo o intervalo dado em dias. Preencha a coleção `DataHora` de acordo com o intervalo de repetição fornecido. Dica: crie o objeto `DataHora` inicial e use sua função interna `somaDias` para criar iterativamente as novas instâncias do intervalo até chegar em `DataHora` final.
   - Implementa os métodos abstratos da superclasse:
     - Método `isConcluido(indice)` que que invoca o método `isPassado()` do elemento `indice` da coleção de objetos `DataHora` e retorna seu resultado.
     - Método `__str__` que imprime (em um laço) **todos as ocorrências `i` do evento** na forma `"Evento: _titulo, Data: data_hora[i], Descrição: _descricao, Concluido: isConcluido(i)"`.
   - Crie o método `editar_data_hora` que recebe `data_hora_antiga` e `data_hora_nova` e altera o elemento da coleção de objetos `DataHora` que corresponde a `data_hora_antiga` fornecida.

   Teste a classe `EventoRecorrente` com o seguinte código:

```
# criar evento
eventos = EventoRecorrente(
    'Reunião', 'Sala 302, prédio da esquina',
    '05/01/2024, 16:30', '05/01/2025, 16:30', 30)
# imprimir eventos
print(eventos)
```

```
# editar um dos eventos
eventos.editar_data_hora('05/12/2024, 16:30',
                         '05/12/2024, 11:30')
# imprimir eventos
print(eventos)
```

5) Por fim, vamos só ver o polimorfismo em ação. Crie e preencha uma lista de eventos, sendo alguns do tipo `EventoUnico` e outros do tipo `EventoRecorrente`. Sobre essa lista, execute o laço de impressão a seguir:

   ```
   for evento in lista_eventos: print(evento)
   ```

   A função `print` irá invocar o método especial `__str__` das classes correspondentes dependendo do tipo do objeto recebido. Aí está o polimorfismo :)

Instruções para submissão: Baixe o arquivo 2.10-OO.ipynb (link a seguir), abra para edição no Jupyter Notebook, e preencha as respostas das atividades.

https://github.com/camilalaranjeira/python-intermediario-exercicios/blob/main/modulo2/2.10-OO.ipynb

Instruções para submissão: envie o arquivo acima preenchido com as suas respostas para a pasta modulo2 do seu repositório.