Código: 98H04-06 POD - Programação Orientada a Dados
Prof. Dr. Dalvan Griebler, dalvan.griebler@pucrs.br

Atualizado: 08/05/2022
Co-author: Adriano Marques Garcia

Trabalho em Grupo sobre Python
Sistema Bibliotecário

Contexto

Durante a disciplina de programação orientada a dados, vocês estão aprendendo a programar na linguagem de programação em **Python**, que é multi-paradigma. Neste trabalho, o objetivo é avaliá-lo relativo a parte de Programação Orientada a Objetos (POO), Módulos e Manipulação de Arquivos. O trabalho também será uma forma de exercitar e aplicar os conceitos de forma prática.

A PUCRS decidiu implementar um novo sistema de gerenciamento para a biblioteca. Eles precisam de uma solução para manipular os dados dos usuários (alunos e funcionários), fazer o controle do acervo e gerenciar o empréstimo de livros. Aproveitando que este semestre está ocorrendo a disciplina de Programação Orientada a Dados, convidaram o professor para criar uma lista de requisitos para um protótipo do novo sistema de gerenciamento da biblioteca, com o objetivo de avaliar o seu conhecimento e o quão apto você está para atuar no desenvolvimento deste projeto.

Restrições

- Está permitido o uso de todos os métodos da classe **str** (string) e estrutura de dados composta (listas, dicionários, tuplas, conjuntos e arranjos).
- Apenas os módulos **abc** e **sys** podem ser importados para implementar o sistema ou a aplicação bancária.
- Outros métodos como **len()** que são nativos e não dependem de importação de módulos estão autorizados (lista de métodos). **Métodos que não foram apresentados em aula, devem ser justificados e explicados através de comentário no código**.

Requisitos do Código

O sistema de uma biblioteca deve ser devolvido para que possa ser usado como um pacote do Python. Além disso, deve ser desenvolvido uma aplicação em Python que utilize este pacote para realizar uma sequência de operações que estarão pré-definidas em um arquivo.

Prof. Dr. Dalvan Griebler, dalvan.griebler@pucrs.br

Especificações e requisitos do sistema de biblioteca:

1) Todo o sistema de biblioteca deve ser implementado como um pacote do Python chamado **Biblioteca**, composto de um módulo chamado **bibSys**.
2) Esse módulo **bibSys** precisa implementar pelo menos três classes principais: **Usuario**, **Livro**, e **Acervo**.

3) **Classe Usuario**:
    a) A classe Usuário deve ser uma **classe abstrata** que implementa métodos para cadastrar usuários e retornar dados de usuários.
    b) Esses métodos deverão ser os responsáveis por manipular os dados no arquivo **usuarios.dat** na base de dados.
    c) Essa classe deve ser do tipo abstrata e será estendida pelas Subclasses **Aluno** e **Funcionario**.
    d) A classe Usuário e suas subclasses precisam permitir que novos usuários sejam cadastrados, permitir obter os dados de usuários cadastrados, realizar cálculo de multa baseado nos atributos de cada indivíduo e registrar a multa na base de dados.
        i) Os métodos para **cadastrar usuário** e **calcular multa** devem ser **métodos abstratos** da classe Usuário.
    e) Você deve garantir que um mesmo usuário (mesmo CPF) não possa ser cadastrado duas vezes.
    f) Para cada usuário é preciso armazenar os seguintes dados:
        i) Nome
        ii) CPF (um número de 11 dígitos)
        iii) Matrícula (um número de 8 dígitos)
        iv) Multa (inicialmente atribuída como zero)
            (1) Devem ser utilizados decoradores ou propriedades para controlar o acesso a esse atributo.
        v) Curso (apenas para alunos) ou departamento (para funcionários).
    g) O usuário do tipo aluno precisa ter um prazo limite de 20 dias para devolver um livro (após esse período será aplicada multa). A multa para os alunos é de 0,25 PUCoins por dia de atraso.
    h) O usuário do tipo funcionário precisa ter um prazo limite de 10 dias para devolver um livro (após esse período será aplicada multa). A multa para os funcionários é de 0,50 PUCoins por dia de atraso.

4) **Classe Livro**:
    a) A classe Livro deve implementar os métodos de empréstimo e devolução de livros.
    b) Esses métodos serão responsáveis por manipular os dados no arquivo **emprestimos.dat** na base de dados. Note que a manipulação desse arquivo deve ser realizada dentro dessa classe, onde o protocolo de gerenciamento de contexto deve ser utilizado para abrir e fechar o arquivo.
    c) Para cada livro é necessário armazenar os seguintes dados:
        i) Título

ii) Autor
iii) Ano (< ano atual)
iv) Exemplares (quantidade de exemplares do livro)

d) Livros cadastrados no acervo também precisam ter um atributo de código do livro. Esse código precisa ter 5 dígitos e é atribuído ao livro pela classe Acervo no momento do cadastro. O controle de acesso a esse código deve ser feito através de decoradores ou propriedades.
e) É necessário definir um método da classe que faça o controle de quantos exemplares de livros foram emprestados e quantos foram devolvidos.

5) Classe Acervo:

a) A classe **Acervo** deve implementar métodos de adição, remoção e pesquisa de livros no acervo. Esses métodos serão responsáveis por manipular os dados no arquivo **acervo.dat** na base de dados. Note que a manipulação desse arquivo deve ser realizada dentro dessa classe, onde o protocolo de gerenciamento de contexto deve ser utilizado para abrir e fechar o arquivo.

6) Implemente também dentro do módulo **bibSys** uma classe chamada **PUCoin** para representar o tipo de moeda que trabalha com números exatos. Isso significa que todo valor calculado e atribuído em forma de multa por atraso de livro precisa ser deste tipo de moeda. As operações matemáticas necessárias precisam ser suportadas.

7) É necessário que sejam implementadas classes para tratamento de exceções quando ocorrerem operações inadequadas no sistema. Por exemplo, registros não encontrados ou impossibilidade de realizar um empréstimo de livro.

8) É necessário também tratar erros com tipos de dados inválidos e incoerentes, como CPF e matrícula.

Especificações e requisitos da aplicação de teste e execução das operações diárias:

1) A aplicação deve receber como argumento na linha de comando uma lista de arquivos texto (**dia1.txt, dia2.txt, dia3.txt, ...**), onde cada arquivo representa um dia de funcionamento da biblioteca, contendo uma sequência de ações e operações que devem ser feitas pela aplicação.
2) A aplicação não pode parar quando se deparar com operações não permitidas ou inválidas. Os erros devem ser armazenados em um arquivo de log. Por exemplo, se um usuário foi impossibilitado de realizar um empréstimo de livro, a aplicação deve progredir para a próxima operação ou passar para o próximo dia. Este arquivo de log serve para monitorar o que foi possível realizar ou não.
3) Ao terminar de processar os arquivos de operações diárias, a aplicação deve mostrar o total de operações de empréstimo e o total de operações de devoluções bem sucedidas realizadas. Essa informação deve ser obtida da classe Livro.

Prof. Dr. Dalvan Griebler, dalvan.griebler@pucrs.br

Entrega

Vocês devem enviar ao final deste trabalho através de um *pull request* no repositório do GitHub:

- **biblioteca:** este diretório do pacote com o módulo desenvolvido.
- **database:** este diretório que contém os arquivos de registro do sistema.
- **dias:** diretório contendo os arquivos de operações diárias.
- **saida-esperada** Este diretório server para verificar se o programa está gerando a saída correta (ele contém o estado final dos arquivos da base de dados após uma execução de todas as operações na ordem dia1.txt, dia2.txt, dia3.txt).
- **app.py** aplicação onde são realizadas as operações.
- **README.md:** arquivo descrevendo como usar a aplicação e o pacote.
- **DICAS.md:** arquivo contendo dicas para guiar e auxiliar no desenvolvimento do trabalho.
- **RESUMO.md:** arquivo contendo o resumo de no máximo 400 palavras relatando as dificuldades, desafios ou outras observações que achar relevante comentar sobre a realização deste trabalho. Você também pode comentar sobre sua evolução no conhecimento. Esse resumo deve ser enviado junto dos demais arquivos.