Orientação a Objetos - FGA0158

Semestre: 2021_1

Grupo 3:

João Pedro Alves Machado - 19/0015276
Felipe Costa Gomes - 19/0012757
Sidney Fernando Ferreira Lemes - 19/0037997
Lucas Pereira Pires - 17/0108996
Rodrigo Ribeiro Lopes Trindade - 17/0113922
Douglas Evaristo de Sousa - 16/0049342

Detalhamento da implementação

ENTIDADES:

Para o desenvolvimento do projeto as principais classes criadas foram todas no pacote entidade, dando corpo ao projeto.

- **Pessoa:** Classe criada onde são armazenados os dados de cada membro da república cadastrado. Os métodos getNome, getEmail e getRenda são usados para se obter o nome, email e renda, respectivamente, da pessoa. Por outro lado, os métodos setRenda e toString são utilizados para atualizar a renda e registrar textualmente os dados cadastrados.
- **Categoria:** Classe criada para armazenar o nome da categoria cadastrada, utilizando o método getNome.
    - **Subcategoria:** Classe filha da classe Categoria, ou seja, ela é criada através de *herança*. Nessa classe se armazena o dado da subcategoria cadastrada. O método getCategoria é utilizado para se obter a classificação da subcategoria.
- **Despesa:** Classe criada para registrar as despesas da república onde getDescrição, getValor e getCategoria fornecem, respectivamente, as informações sobre a descrição, valor e categoria de cada despesa. O método toString é aplicado para registrar textualmente os dados cadastrados.
- **Republica:** É a classe principal das entidades, em programa, mantém os dados de referência para as instâncias do tipo Pessoa e Despesa, de forma a gerenciar a manipulação destes dados pelo programa principal. Os métodos de cadastro são utilizados para adicionar uma instância do tipo específico na lista de dados de programa da república, os métodos “ler” fornecem uma inicialização dos dados

salvos nos arquivos de persistência, enquanto os métodos “gravar” atualizam estes dados.

SERVICOS:

Pacote criado para implementar o cálculo da dívida em duas opções: regra igualitária (as despesas mensais são divididas igualmente por todos os moradores da república) e regra proporcional (as despesas mensais são divididas considerando a capacidade de pagamento de cada morador conforme sua renda mensal).

- **CalculoDivida:** Classe abstrata criada para que, através dos dados recebidos da Republica, seja capaz de calcular a renda total e a despesa total através dos métodos rendaTotal e despesaTotal.
    - **RegraIgualitaria:** Classe filha capaz de calcular a dívida por pessoa, através do método dívida, dividindo o valor total das despesas (obtido em depesaTotal) pelo número de pessoas na república.
    - **RegraProporcional:** Classe filha capaz de calcular a dívida por pessoa, através do método dívida, dividindo o valor total entre as pessoas da república de forma proporcional com a renda de cada um.

Há um polimorfismo de sobreposição no método “dívida” nas classes filhas, RegraIgualitária e RegraProporcional.

APP:

Pacote que contém o programa principal e os meios de interface com o usuário.

- **Main:** Classe responsável pela parte principal do programa que irá coletar informações sobre o mês e o ano, apresentando uma lista de opções para cada dado. Em seguida a classe fornece as opções de divisão das dívidas e inicia o menu do programa apresentando as opções:
    1. **Cadastrar pessoa:** irá criar uma nova pessoa no programa através do método criaPessoa da UI e registrar estas através do método cadastroPessoa da Republica.
    2. **Cadastrar despesa:** irá criar uma nova pessoa no programa através do método criaDespesa da UI e registrar estas através do método cadastroDespesa da Republica.
    3. **Excluir pessoa:** Permite selecionar uma das pessoas cadastradas para ser removida do cadastro.
    4. **Excluir despesa:** Permite selecionar uma das despesas cadastradas para ser removida do cadastro.
    5. **Calcular dívida:** Permite escolher uma pessoa para calcular a dívida desta.

6. **Imprimir relatório:** Imprime o relatório das pessoas, mostrando a dívida de cada uma calculada através da regra solicitada.
7. **Mudar Regra de calculo:** Permite alterar o cálculo da dívida.
0. **Sair:** Fecha o programa

Por fim o método gravarPessoa e gravarDespesa irão registrar as pessoas e as despesas no arquivo txt.

- **UI:** Do inglês User Inteface (Interface de Usuário), a classe mantém os métodos estáticos que fornecem o contato com o usuário de forma a enviar dados para o programa principal, ou exibir dados ao usuário.
    - O método selecionar é um método genérico que permite que o usuário selecione um dado dentro de uma lista pré-determinada por uma caixa de seleção. Além deste existem alguns métodos de selecionar mais específicos, que são utilizados para casos específicos, com uma personalização maior.
    - Os métodos criaPessoa e criaDespesa recolhem os dados necessários para criar uma instância do tipo especificado e retornam a referência a ela.
    - O método showMenu exibe a string menu, definida na classe Main, enquanto retorna o valor inteiro, passado pelo usuário, referente à opção da próxima funcionalidade realizada pelo programa.
    - O método mensagem recebe uma String como parâmetro, que é então exibida ao usuário. Os métodos de alerta são semelhantes, mas exibem uma caixa de alerta com a mensagem recebida, e pode ou não ter o título personalizado.

EXCEPTIONS:

As exceções servem para tratamento de eventuais erros que ocorrem no meio da aplicação, seja por valores inválidos inseridos pelo usuário. No caso do programa da republica, foram implementadas algumas exceções para verificar o tipo de valor informado pelo usuário e assim impedindo que o programa quebre ou pare de funcionar, mostrando sempre uma mensagem de erro e requisitando uma nova ação do usuário, de forma que a mediação entre a lógica do programa e o usuário ocorra tudo bem. Além dessas, foram implementas 5 exceções que foram exigidas no corpo do trabalho, que são:

- **DadosPessoaisIncompletosException**: verifica os dados pessoais inseridos são válidos e retorna uma mensagem pedindo para inserir os dados novamente caso algum dado pessoal inserido seja inválido.
- **RendimentoInvalidoException:** verifica se o rendimento cadastrado é numérico e positivo e retorna uma mensagem pedindo para cadastrar um rendimento válido caso seja inválido.
- **CategoriaNaoInformadaException**: verificar se a categoria foi cadastrada e retorna uma mensagem pedindo para cadastrar uma categoria válida caso não seja.
- **DescricaoNaoInformadaException:** verifica se a descrição informada é válida e retorna uma mensagem pedindo para cadastrar uma descrição valida caso não seja.