Departamento de Ciência da Computação – DCC – UFJF

Professor: Gleiph Ghiotto Lima de Menezes Disciplina: Orientação a objetos

Nome:_____ Matrícula:_____

Segunda prova

Orientações gerais

- A prova é individual e provas semelhantes receberão nota zero;
- Colocar nome e matrícula em todos os arquivos, a não observação desse requisito pode acarretar nota zero;
- As questões devem ter o código compilando e executando utilizando o Maven. Caso contrário, receberão nota zero;
- Sugiro utilizar os projetos em anexo no Classroom; e
- Leia atentamente a prova e boa prova!

Questão 1 (25 pontos)

Enunciado

Uma empresa organiza eventos e deseja considerar a data de nascimento dos participantes para permitir o acesso aos eventos que possuem vagas limitadas. Essa solução será automatizada utilizando um programa Orientado a Objetos na linguagem Java. A implementação inicial vai tratar de dois tipos de eventos: Natal e Carnaval. Porém, a quantidade de eventos podem crescer ao longo do tempo.

Faça a implementação do programa seguindo a descrição a seguir:

Implemente uma classe **DataException** que vai ser lançada quando um usuário fornecer uma data inválida. **(2 pontos)**

Implemente uma classe **Data** com:

- Os atributos dia, mês e ano encapsulados; **(1 pontos)**
- Implemente um método *parser* estático que recebe como parâmetro uma data (String) no formato dd/mm/aaaa onde dd representa o dia que pode variar de 1 a 30 e pode ser representado por um ou dois caracteres (*e.g.*, o dia primeiro pode ser representado por 01 ou 1); mm representa o mês que vai variar de 1 a 12 podendo ser representado por um ou dois caracteres (*e.g.*, fevereiro pode ser 2 ou 02); e aaaa representa o ano que sempre terá quatro caracteres

numéricos. Esse método deve retornar um objeto do tipo Data. Caso a transformação não seja possível, retorne uma exceção do tipo DataException; **(3 pontos)**

- Implemente um método diferença que recebe uma Data data como parâmetro e retorna a diferença entre a data fornecida e o próprio objeto em dias (this). Vale ressaltar que a diferença é feita considerando a quantidade de dias de data menos a quantidade de dias do objeto (this). Supondo que data seja 01/01/2023 e this seja 02/01/2023, o valor retornado deve ser de -1 dia. Por outro lado, supondo que data seja 02/01/2023 e this seja 01/01/2023, o valor retornado deve ser de 1 dia. **(1 ponto)**

OBS: Considere que todos os meses possuem 30 dias.

Crie uma classe **Pessoa** que possui:

- Os tributos dataNascimento do tipo Data, definido anteriormente, o nome e uma lista de Evento (definido a seguir), chamada de eventos, que ela vai participar; **(1 ponto)**
- Um método podeParticiparEvento que recebe um Evento com parâmetro e retorna true se a pessoa não tiver outro evento para o mesmo dia. Caso contrário, retorna false; **(2 pontos)**
- Um método agendarEvento recebe um evento e só o adiciona à lista de eventos se a pessoa puder participar (use o método anterior). **(2 pontos)**

Crie uma abstrata classe **Evento** que possui:

- Os atributos valor, data (tipo Data), nome, capacidade e uma lista de Pessoa chamada de participantes.**(1 ponto)**
- Um método temVaga que retorna true caso a quantidade de participantes não tenha atingido a capacidade do evento;**(2 pontos)**
- Um método abstrato pessoaPodeParticipar que recebe uma Pessoa como parâmetro e retorna um booleano;**(1 ponto)**
- Um método adicionaPessoa que recebe uma pessoa como parâmetro e só adiciona a pessoa aos participantes caso ela possa participar (utilize o método pessoaPodeParticipar). **(2 pontos)**

Crie uma classe **Natal** que estende **Evento** que possui:

- um método pessoaPodeParticipar concreto que recebe uma pessoa como parâmetro e retorna true caso a pessoa tenha idade maior ou igual a 5 anos e o evento ainda tenha pelo menos uma vaga disponível.**(2 pontos)**

Crie uma classe **Carnaval** que estende **Evento** que possui:

- um método pessoaPodeParticipar concreto que recebe uma pessoa como parâmetro e retorna true caso a pessoa tenha idade maior ou igual a 18 anos e o evento ainda tenha vaga.**(1 ponto)**

Faça as classes anteriores integrarem com a Main descrita abaixo, se necessário, faça os seus ajustes e gerar o resultado esperado. **(4 pontos)**

OBS1: O código da main está comentado no código fornecido.
OBS2: Vocês podem criar métodos de acesso e outros métodos para facilitar a implementação.

Classe Main:

```
public class Main {

    public static void main(String[] args) {

        List<Evento> eventos = new ArrayList<>();
        List<Pessoa> pessoas = new ArrayList<>();

        try {
                                        Pessoa pessoa = new
Pessoa(Data.parser("00/03/1987"), "Gleiph");
        } catch (DataException ex) {
                System.out.println("Pessoa com data de nascimento
inválida");
        }

        try {
                                Evento evento = new Natal(100,
Data.parser("25/12/12356"), "Festa de Natal", 2);
        } catch (DataException ex) {
            System.out.println("Evento com data inválida");
        }

        try {
                pessoas.add(new Pessoa(Data.parser("03/03/1987"),
"Gleiph"));
                pessoas.add(new Pessoa(Data.parser("08/12/1922"),
"Elvira"));
                 pessoas.add(new Pessoa(Data.parser("1/01/2022"),
"Maria"));
                pessoas.add(new Pessoa(Data.parser("01/03/2016"),
"Joao"));
                 pessoas.add(new Pessoa(Data.parser("07/7/2000"),
"Gabriel"));
                pessoas.add(new Pessoa(Data.parser("07/10/2000"),
"Pedro"));
                pessoas.add(new Pessoa(Data.parser("10/12/2010"),
"Mariana"));
                pessoas.add(new Pessoa(Data.parser("10/12/2005"),
"Mariane"));
                pessoas.add(new Pessoa(Data.parser("08/12/2005"),
"Mario"));

                                    eventos.add(new Natal(100,
Data.parser("25/12/2022"), "Natal DCC", 3));
                                    eventos.add(new Natal(100,
Data.parser("25/12/2022"), "Natal ICE", 4));
```

```
                                    eventos.add(new   Carnaval(200,
Data.parser("25/02/2023"), "Unidos da OO", 6));

            for (Evento evento : eventos) {
                for (Pessoa pessoa : pessoas) {
                        if(evento.pessoaPodeParticipar(pessoa) &&
pessoa.podeParticiparEvento(evento)){
                        pessoa.agendarEvento(evento);
                        evento.adicionaPessoa(pessoa);
                    }
                }
            }

            for (Evento evento : eventos) {
                                    System.out.println("Evento:  "  +
evento.getNome());

                System.out.println("Perticipantes: ");
                                       for  (Pessoa  participante  :
evento.getParticipantes()) {

System.out.println("\t"+participante.getNome());
                }
                                     System.out.println("Lucro:  "  +
(evento.getValor()*evento.getParticipantes().size()));

            }

        } catch (DataException e) {
            System.out.println("Alguma data está invalida!");
        }

    }

}
```

Saída esperada

```
Pessoa com data de nascimento inválida
Evento com data inválida
Evento: Natal DCC
Perticipantes:
     Gleiph
     Elvira
     Joao
Lucro: 300.0
Evento: Natal ICE
Perticipantes:
     Gabriel
     Pedro
     Mariana
     Mariane
Lucro: 400.0
Evento: Unidos da OO
Perticipantes:
     Gleiph
     Elvira
     Gabriel
     Pedro
Lucro: 800.0
```