UDESC - Fundação Universidade do Estado de Santa Catarina
Professor: Fabiano Baldo
Disciplina: Programação Orientada a Objetos

A lista deve ser entregue até as 23h59 do dia definido na atividade do Moodle, os arquivos devem ser compactados em um arquivo *.zip* ou *.tar*. O arquivo compactado deverá conter o projeto apenas os arquivos **.java**. **Não serão aceitos projetos com os códigos-fonte no formato *.class*!**

Lista 2: Coleções e Generics

Exercício 1:

Crie um programa em Java que leia do usuário as informações relacionadas a várias pessoas. Cada pessoa possui nome, idade, cpf e cidade. O programa termina a leitura quando o usuário digita o valor -1. A cada nova pessoa lida, o programa deve exibir a lista de pessoas (ordenada alfabeticamente pelo nome) com suas respectivas informações, agrupando-as de acordo com as faixas etárias a seguir:

- 1 até 12: crianças;
  - Igor Pereira, 5, 472.938.253-43, Jaraguá do Sul;
  - Lucas da Silva, 6, 272.938.273-93, Joinville;
- 13 até 18: adolescentes;
  - João Pedro Oliveira, 17, 475.458.343-21, Florianópolis;
  - Luiz Otávio Ramos, 14, 343.945.234-04, Joinville;
- 19 até 25: jovens;
- 26 até 59: adultos;
- 60 ou mais: idosos;

Dica, utilize a classe **Scanner** passando um File para ler um arquivo com as pessoas a serem cadastradas:

```
Scanner sc;
try {
    sc = new Scanner(new File("nomeArquivo.extensao"));
} catch (FileNotFoundException e) {
    sc = new Scanner(System.in);
}
sc.nextInt();
... // Utilize o scanner normalmente
```

Exercício 2:

Crie a classe **Matriz<T>**, ela deve possuir como atributos a ordem da matriz e uma matriz de elementos do tipo T a serem alocados pelo método construtor, que deve receber dois parâmetros n e m, estes correspondentes a ordem da matriz. Crie um enumerável Quadrante, contendo os valores: PRIMEIRO, SEGUNDO, TERCEIRO e QUARTO. A classe Matriz, deve possuir os seguintes métodos:

- **boolean set(T objeto, int i, int j):** esse método deve colocar o objeto na posição **i,j** da matriz e retornar true. Caso um dos valores i ou j forem maiores que a ordem da matriz, o método deve retornar false.
- **T get(int i, int j):** esse método deve retornar o objeto na posição **i,j** da matriz. Caso um dos valores i ou j forem maiores que a ordem da matriz, o método deve retornar null.
- **List<T> getLinha(int linha):** esse método deve retornar uma lista contendo os elementos presentes na linha especificada como parâmetro;
- **List<T> getColuna(int coluna):** esse método deve retornar uma lista contendo os elementos presentes na coluna especificada como parâmetro;
- **List<T> getElementosQuadrante(Quadrante quadrante):** esse método deve retornar uma lista contendo os elementos presentes no quadrante passado como parâmetro. A Figura 1 apresenta um exemplo da chamada desse método para uma matriz:

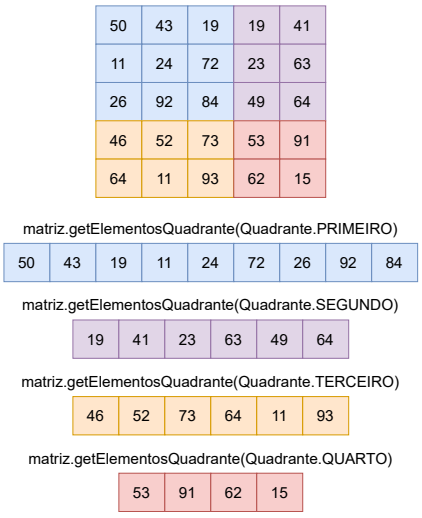

Figure 1: Exemplo de chamadas do método **getElementosQuadrante()** para cada um dos valores do enumerável Quadrante em uma matriz não-quadrada.

Exercício 3:

Utilizando a classe criada no exemplo anterior, crie um programa em Java que leia do console uma matriz 5×5, até que o usuário digite -1 e exiba no console o menor elemento presente na matriz. Para encontrar o menor elemento, o programa deve encontrar o menor elemento dos quadrantes e exibir o menor deles.

Exercício 4:

Crie um programa em Java que possui as classes Equipe, Aluno e Turma, descritos na Figura 2. A Turma possui uma lista de Alunos, estes com nome, idade e cinco notas, tal como na figura. A classe Turma possui um método para adicionar novos alunos e um método privado **ordenarAlunosPorMedia()**, esse método deve rearranjar a lista de alunos, ordenando-os em ordem crescente de média. Além disso, a classe Turma deve possuir o método **separarEmEquipes()**, que retorna uma lista de equipes contendo 3 ou 4 alunos, estes agrupados levando em consideração a média. Agrupe os alunos pegando um ou dois elementos do começo da lista ordenada por média e um ou dois alunos do final da lista (veja Figura 3 exemplos de possíveis formas de agrupar os alunos). O programa em Java deve ler do console os alunos (utilize o mesmo método apresentado no Exercício 1) e adicioná-los a Turma. Após terminada a leitura, o programa deve chamar o método **separarEmEquipes()** e após isso exibir as equipes, seus membros e a média de cada membro.

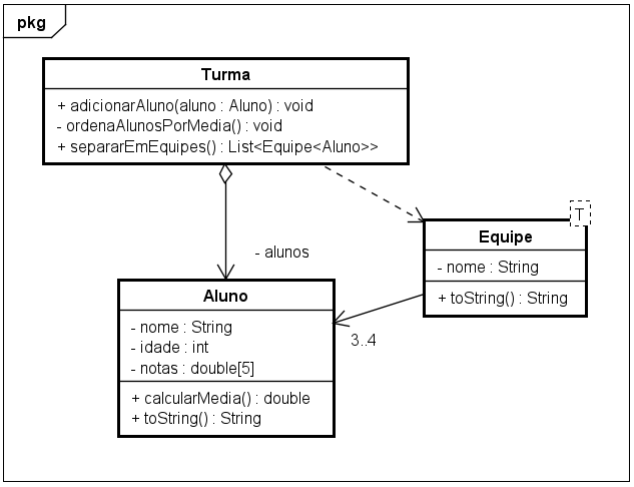

Figure 2: Diagrama de Classe contendo as classes Turma, Aluno e Par.

Nome	Média
Julia	9.7
Gabriela	9.2
Matheus	8.7
Isabela	8.4
Lucas	7.8
Marcos	7.5
Letícia	7.2
Beatriz	6.6
Rafael	5.6
Gabriel	4.7

Nome	Média
Julia	9.7
Gabriela	9.2
Matheus	8.7
Isabela	8.4
Lucas	7.8
Marcos	7.5
Letícia	7.2
Beatriz	6.6
Rafael	5.6
Gabriel	4.7

Figure 3: Exemplos de separações dos alunos em equipes levando em consideração suas médias.

Exercício 5:

Crie um algoritmo em Python que seja capaz de ler um json e transformá-lo em um ou vários objetos do tipo Filme. Cada Filme possui um titulo, ano, classificação indicativa e nota dos usuários. O programa deve ser capaz de listar todos os filmes do arquivo .json e ordená-los por nota. Na classe Filme deve haver um método que dado um pedaço de JSON recebido como parâmetro, que represente um Filme, atribua os valores do JSON aos atributos do objeto. O algoritmo deve ser capaz de receber um JSON na forma descrita a seguir:

```
[
  {
    "titulo": "A Rainy Day in New York",
    "ano": 2019,
    "classificacao": "PG-13",
    "nota": 6.6
  },
  {
    "titulo": "Jurassic World: Fallen Kingdom",
    "ano": 2018,
    "classificacao": "PG-13",
    "nota": 6.2
  },
  {
    "titulo": "2001: A Space Odyssey",
    "ano": 1968,
    "classificacao": "G",
    "nota": 8.3
  },
  {
    "titulo": "Toy Story 4",
```

```
        "ano": 2019,
        "classificacao": "G",
        "nota": 7.8
    }
]
```

Veja como utilizar a biblioteca JSON para converter uma string em um dicionário:

```
# importa a biblioteca do json
import json

# instancia uma string que contem um json
string_json = "{'cidade':'Joinville', 'bairros':['Floresta','America','Bom Retiro']}"

# transforma a string em um json/dicionario
json_do_exemplo = json.loads(string_json)

# acessando o array bairros
print(json_do_exemplo['bairros'])
```

Os dados foram retirados do dataset Complete IMDB Movies Dataset e adaptadas as tags para o português de forma a tornar o exemplo mais didático. Caso deseje, utilize o arquivo.json original e transforme-o para um objeto.