

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 3: Herança e Classes Abstratas

Exercício 1:

Implemente em Java a classe abstrata **Animal** que possui um atributo **nome** e um método abstrato **String emitirSom()**. Tal método deve retornar uma **String**. Crie outras três classes que estendam **Animal** e implementam o método **String emitirSom()** de acordo com o som que o animal emite. Em seguida, crie um método **main()** em uma classe própria para ele e instancie ao menos **dois** objetos de cada uma das classes. Exemplo de som emitido por uma cão chamado Rex: “Rex: Au-au”.

Exercício 2:

A partir do diagrama da Figura 1, implemente as classes **FormaGeometrica** (abstrata), **TrianguloEquilatero**, **Losango** e **Circulo**.

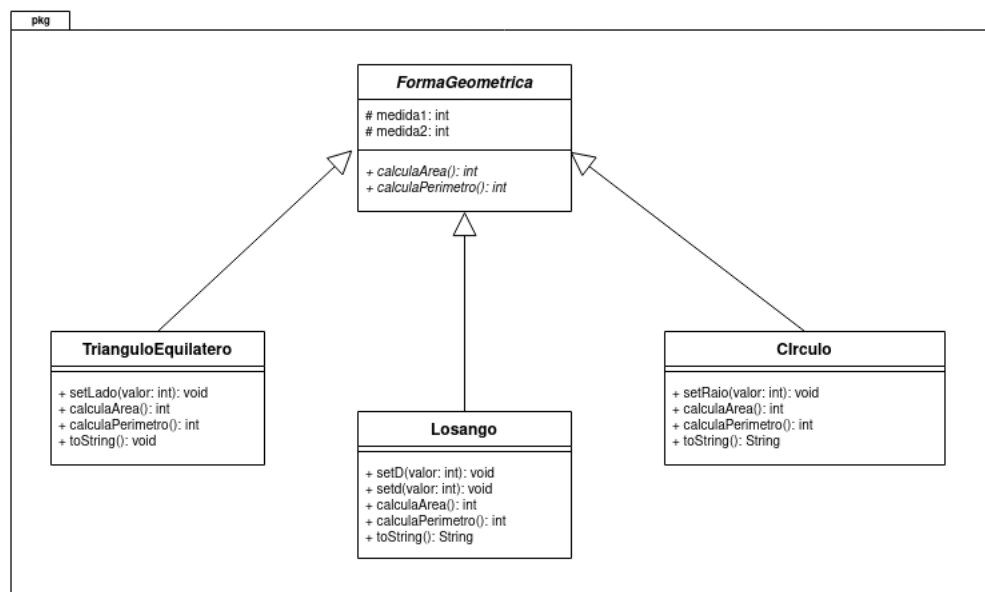


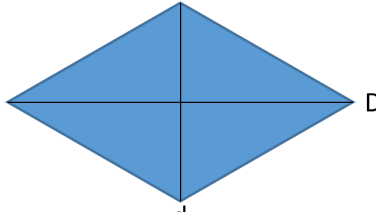
Figure 1: Diagrama UML das classes **FormaGeometrica**, **TrianguloEquilatero**, **Losango** e **Circulo**.

A classe abstrata `FormaGeometrica` possui duas medidas, das quais as classes filhas interpretam de acordo com as medidas que elas utilizam, setando tais valores por meio dos setters específicos nas classes filhas. A classe `TrianguloEquilatero` possui um atributo `lado`. Para alterar tal valor, a classe possui um setter, o qual modifica o atributo presente na superclasse. As fórmulas para o cálculo da área e do perímetro podem ser vistas na Figura 2.

$$A = \frac{\sqrt{3}}{4} l^2 \quad p = 3l$$

Figure 2: Cálculo da área e do perímetro de um Triângulo Equilátero. Fonte: [Wikipedia](#).

Já a classe `Losango` possui dois setters, um para cada uma das diagonais do losango. Além de implementar os métodos para calcular a área e o perímetro tal como apresentado na Figura 3.

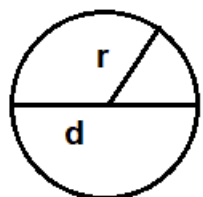


The diagram shows a blue rhombus with its two diagonals intersecting at the center. The horizontal diagonal is labeled 'D' at its right end, and the vertical diagonal is labeled 'd' at its bottom end.

$$A = \frac{D \cdot d}{2}$$
$$l = \sqrt{\frac{D^2}{4} + \frac{d^2}{4}}$$
$$P = 4 * \sqrt{\frac{D^2}{4} + \frac{d^2}{4}}$$

Figure 3: Cálculo da área e perímetro do Losango. Fonte: [Toda Matéria](#).

Na classe `Circulo`, há apenas o método `setRaio()` responsável por setar o valor do raio e do diâmetro ($2 \times r$) nos atributos de sua superclasse. Além de implementar os métodos de cálculo do perímetro e da área usando as fórmulas da Figura 4.



The diagram shows a circle with a horizontal diameter line passing through the center, labeled 'd' at both ends. A radius line is drawn from the center to the upper right edge, labeled 'r'.

$$A = \pi * r^2$$
$$P = \pi * d$$

Figure 4: Cálculo da área e perímetro do Circulo. Fonte: [Toda Matéria](#).

Por fim, o método `toString()` deve retornar uma `String` contendo as dimensões das formas geométricas e o valor de seu perímetro e área. Crie outra classe contendo um método `main()` e instancie ao menos **dois** objetos de cada uma das classes.

Exercício 3:

Grafos são estruturas de dados muito utilizadas na computação. Na Figura 5 é possível ver um exemplo de um grafo. Basicamente, são compostos de vértices (as bolinhas) e arestas (as ligações entre dois vértices). Existem várias formas de representá-los computacionalmente, uma das formas utilizadas é chamada **matriz de adjacência**. Nessa matriz, quadrada e simétrica, para cada aresta que conecta o vértice i ao vértice j , o valor do elemento ij e ji na matriz é igual a 1. Caso não exista uma aresta ligando tais vértices o valor é igual a 0. Na Figura 5 é possível ver o exemplo de um grafo e sua matriz de adjacência.

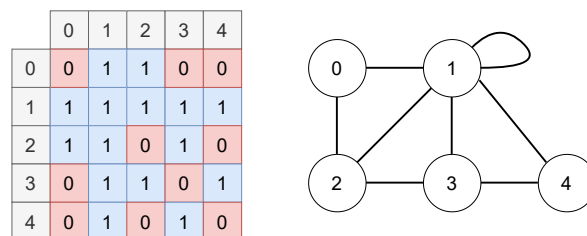


Figure 5: Exemplo de Grafo e sua matriz de adjacência.

Existe também um tipo especial de grafo, chamado de **Dígrafo**. Basicamente, em um dígrafo as arestas possuem uma direção, isto é, possuem um vértice de origem e um vértice de destino. Uma consequência disso é vista na matriz de adjacência, que agora não necessariamente será simétrica. Na Figura 6 é possível ver um exemplo de um dígrafo e sua matriz de adjacência.

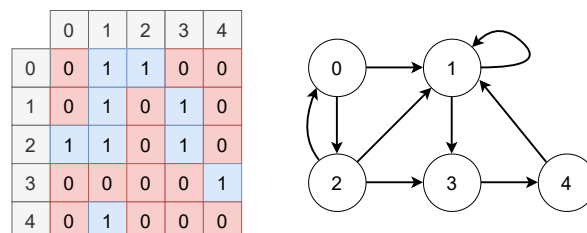


Figure 6: Exemplo de um Dígrafo e sua matriz de adjacência.

Tendo em vista esses conceitos e utilizando de listas ou mapas, implemente as classes descritas no diagrama da Figura 7.

Escolha apenas um dos atributos **matrizAdjacencia1** ou **matrizAdjacencia2** para utilizar. A seguir serão descritos os métodos a serem implementados:

- **adicionarVertice()**: deve criar uma nova linha e uma nova coluna na matriz de adjacência composta inteiramente de zeros.
- **adicionarAresta(int origem, int destino)**: dados os vértices de origem e destino, deve alterar o valor para 1 na matriz de adjacência. Na classe Grafo é necessário sobrescrever o método para tornar a matriz simétrica.

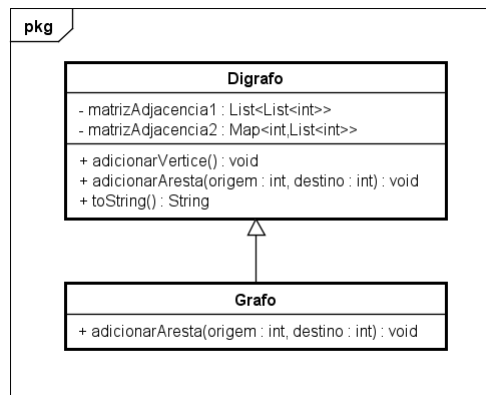


Figure 7: Diagrama UML para as classes Grafo e Digrafo.

- **toString()** deve retornar uma String exibindo a matriz de adjacência juntamente com os seus índices.

Em seguida, crie um método **main()** em uma classe própria para ele e instancie ao menos um grafo e um dígrafo com alguns vértices e arestas e exibindo o **toString()** no final.

Exercício 4:

Implemente o diagrama da Figura 8 em Python.

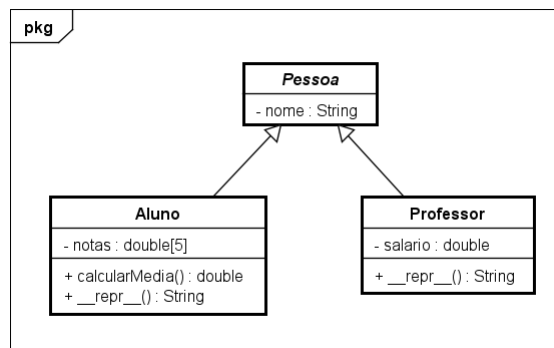


Figure 8: Diagrama UML contendo a classe abstrata Pessoa, Aluno e Professor.

A classe abstrata Pessoa possui apenas um atributo nome. Já a classe Aluno irá estender a classe Pessoa, contendo uma lista de notas e um método **calcularMedia()** que deve somar todas as notas e dividir o valor pela quantidade de notas. Além de sobrescrever o método **__repr__()** que deve retornar a representação do objeto na forma de String, contendo o nome do aluno, as notas e a média, além de informar se o aluno foi aprovado (média maior que

UDESC - Fundação Universidade do Estado de Santa Catarina

Professor: Fabiano Baldo

Disciplina: Programação Orientada a Objetos

7) ou está em exame (média inferior a sete). Por fim, a classe Professor contém apenas um atributo que representa o seu salário e o método `__repr__()` que retorna uma String contendo o nome do professor e seu salário.

Instancie **dois** objetos do tipo Professor e **cinco** objetos do tipo Aluno, exibindo no console o valor desses objetos.