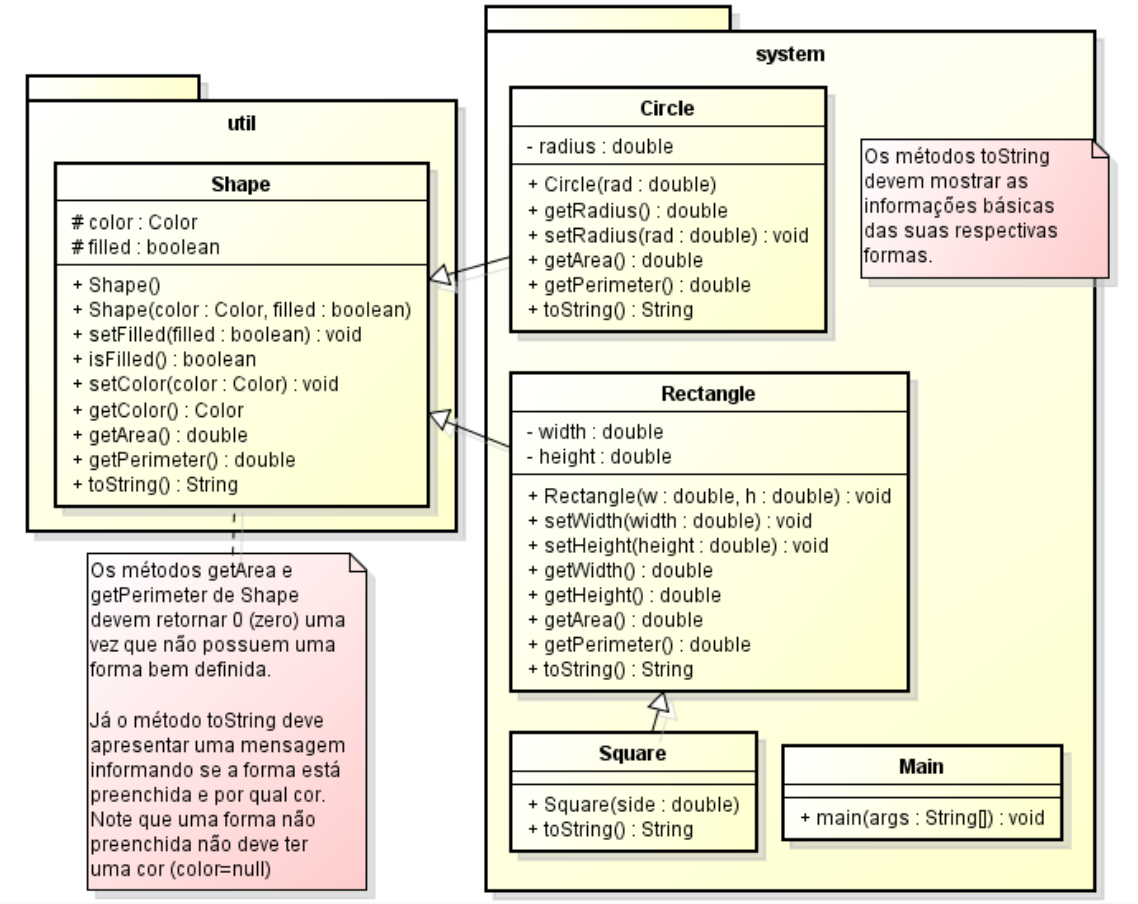
Exercícios

Faça as soluções solicitadas usando JAVA:

1. Construa uma aplicação de formas geométricas com herança. A aplicação a ser construída deverá seguir o modelo apresentado no seguinte diagrama de classes. O símbolo + indica encapsulamento público, - indica privado, # indica protegido e ~ indica encapsulamento do pacote. A seta entre uma classe e outra indica um relacionamento de herança onde a classe apontada é a superclasse.



No método main da classe Main, crie quatro instâncias de formas: (1) um círculo de raio 7 com preenchimento azul, (2) um quadrado de lado 5 sem preenchimento, (3) um retângulo de lados 3x6 com preenchimento preto, e (4) uma forma (Shape) com preenchimento verde. Em seguida remova o preenchimento do retângulo.

Ainda, implemente um método adicional polimórfico denominado *showInfo(Shape)*, que recebe uma forma (Shape) e mostra alguma informação prévia na tela seguida das informações da forma passada por parâmetro usando seu método toString().

2. Escreva uma classe abstrata `Shape` com as seguintes propriedades: um atributo `shapeName` do tipo `String`, um método abstrato `double area()` e um método `toString()` que retorna o nome da forma.

Escreva três classes derivadas de `Shape`: `Sphere`, `RectangularPrism`, e `Cylinder`, sendo que a esfera possui um atributo **raio**, o prisma retangular possui **altura**, **largura** e **comprimento**, e o cilindro possui **raio** e **altura**. Implemente os métodos de área para cada um deles, lembrando que a área de superfície da esfera é dada pela fórmula: $4\pi r^2$ (onde r é o raio), a área do prisma retangular é dada por: $2(ab)+2(ac)+2(bc)$ (onde a é a altura, b é a largura e c é o comprimento) e a área do cilindro é dada pela fórmula: $2\pi r^2+2\pi rh$ (onde r é o raio e h é a altura). Implemente ainda o método `toString()` para as três formas, usando o método `toString()` da superclasse [use: super.toString()] e adicionando também os valores dos atributos. Veja abaixo um exemplo do construtor da esfera:

```
public Sphere(double r){
    super("Sphere");
    radius = r;
}
```

A seguir (próxima página) é apresentada a classe `Paint` que representa um tipo de tinta. Corrija o método *amount()* de tal forma que a quantidade correta de tinta necessária seja retornada (OBS: utilize o atributo *coverage* que indica quantos cm² podem ser pintados com um galão/latão de tinta e, obviamente a área da forma que foi passada como parâmetro).
Na próxima página também é apresentado a classe `PaintThings` que contem o programa principal que seria responsável por gerar as formas e computar a quantidade de tinta necessária. O programa já instancia um tipo de tinta; cabe a você completar o programa de modo a:
(1) Instanciar três formas: um retângulo prismático de 20 x 35 x 10, uma grande esfera (bigBall) com um raio de 15, e um tanque cilíndrico de raio 10 e altura 30;

(2) Utilize os métodos apropriados para atribuir os valores de tintas necessários as variáveis corretas.

```
//*******************************************************
// Paint.java
// Representa o tipo de tinta que tem uma área de cobertura fixa
// por galão/latão. As medidas são em cm².
//*******************************************************
public class Paint{
  private double coverage; //número de cm² por galão/latão

  /**
  * Construtor: Monta o objeto de pintura.
  */
  public Paint(double c){
    coverage = c;
  }

  /**
  * Retorna a quantidade de tinta (número de galões)
  * necessário para pintar a forma passada como parâmetro
  */
  public double amount(Shape s){
    System.out.println ("Computing amount for " + s);
    return 0;
  }
}
```

```java
//*****************************************************************
// PaintThings.java
// Computa a quantidade de tinta necessária para pintar
// várias coisas
//*****************************************************************
import java.text.DecimalFormat;
public class PaintThings{
  public static void main (String[] args){
    final double COVERAGE = 350;
    Paint paint = new Paint(COVERAGE);
    double deckAmt, ballAmt, tankAmt;
    // Instancia as três formas a serem pintadas
    // ...
    // Computa a quantidade de tinta necessária para cada forma
    // ...
    // Mostra a quantidade de tina necessária.
    DecimalFormat fmt = new DecimalFormat("0.#");
    System.out.println ("\n# de latões necessários...");
    System.out.println ("Deck: " + fmt.format(deckAmt) );
    System.out.println ("Big Ball: " + fmt.format(ballAmt) );
    System.out.println ("Tank: " + fmt.format(tankAmt) );
  }
}
```