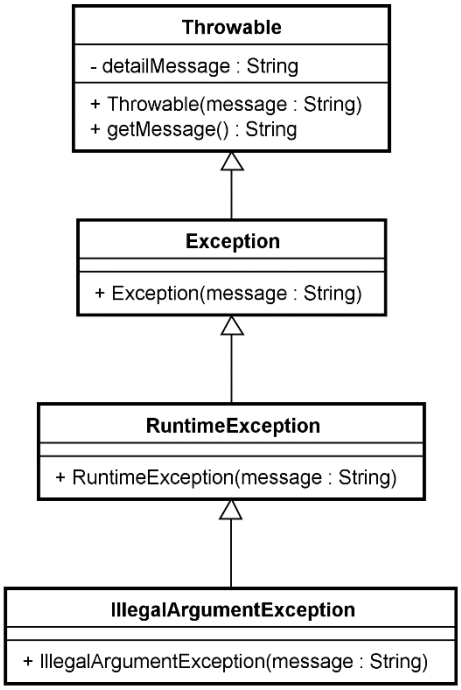
UNIVERSIDADE REGIONAL DE BLUMENAU
CENTRO DE CIÊNCIAS EXATAS E NATURAIS
DEPARTAMENTO DE SISTEMAS E COMPUTAÇÃO
PROFESSOR ANDRÉ FELIPE BÜRGER
PROGRAMAÇÃO ORIENTADA A OBJETOS

Lista de exercícios 12

O objetivo desta atividade é exercitar o tratamento de exceções. Para isso, duplique a solução do projeto da lista de exercícios 7a e realize as implementações das questões abaixo.

No método `calcularIptu()`, garanta que sejam lançadas exceções específicas



Para solucionar as questões seguintes, considere o diagrama de classes abaixo:

A classe `RuntimeException`, é uma classe para representar erros "genéricos", sem informar qualquer detalhe adicional do erro, apenas uma mensagem de erro que fornecemos no construtor desta classe.

O argumento do construtor de `RuntimeException` é utilizado para inicializar a variável `detailMessage`, que é herdada indiretamente da classe `Throwable`. Curiosamente, o método *getter* desta variável não é `getDetailMessage()`, mas sim, `getMessage()`.

A classe de exceções `IllegalArgumentException` é uma classe utilizada para representar erros que correspondem à tentativa de fornecer um argumento inesperado para um método. Considere-o para solucionar a questão 1.

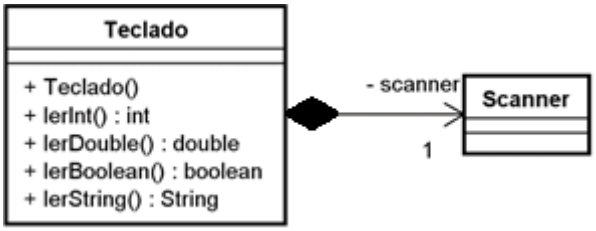
Questão 1

Realize as seguintes alterações:

- Altere o método `setArea()` da classe `Imovel` para que seja recusado áreas de imóveis negativas ou zero.
- Altere o método `calcularIptu()` para recusar a realização de cálculo quando não houver finalidade de imóvel definida ainda;
- Altere o método `setCoeficiente()` da classe Bairro para recusar coeficientes negativos ou zero.
- Quando o botão *Calcular* for acionado e ocorrer qualquer exceção, apresente a mensagem de erro ao usuário. Aqui você pode se utilizar do `JOptionPane.showMessageDialog` para apresentar as mensagens de forma gráfica ao usuário. Não esqueça de tratar também as exceções do tipo `NumberFormatException` para quando o usuário digitar alguma informação que não seja um número nos JTextField correspondentes.

Questão 2

Crie a classe `Teclado`, como apresentado no diagrama abaixo:



O objetivo desta classe é possibilitar a leitura de dados através do teclado, garantindo que os métodos de leitura obtenham dados compatíveis com os tipos esperados. Assim, o método `lerInt()` deve ler do teclado um número e retorná-lo. Caso o dado informado pelo usuário não seja um número inteiro e além disso, compatível com o tipo `int`, o método deve insistir e solicitar o dado novamente ao usuário, até que valor digitado pelo usuário seja compatível com o tipo de retorno do método. O mesmo comportamento deve ser implementado com os demais métodos desta classe - exceto o método `lerString()`, que não possui regra de digitação. No caso do método `lerBoolean()`, considerar que o usuário poderá digitar: *sim*, *verdadeiro*, *positivo* ou apenas a letra "*s*". Em qualquer um destes casos, o método deverá resultar em `true`. Se o usuário digitar *nao*, *falso, negativo* ou a letra "n", o método deverá resultar em `false`. Em qualquer outro caso, o método deverá insistir para que o usuário digite um texto correto.

O construtor da classe `Teclado` deverá instanciar um objeto da classe `Scanner` e utilizá-lo em todos os métodos de leitura.

Após finalizar a classe Teclado, crie um programa para utilizar todos os métodos da classe Teclado.