UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
TESTE DE SOFTWARE - IMD
PROFA.: ROBERTA COELHO

LISTA DE EXERCICIOS

PROBLEMA 1: Um aplicativo desenvolvido para o relógio da APPLE checa o risco de AVC com base na temperatura corporal (medida por um sensor em Celsius) e na frequência cardíaca (medida em quantidade de batimentos por minuto). O método abaixo representa a funcionalidade principal deste aplicativo.

```
public class AVCRisk {

/*
O método retorna o risco do usuário do relógio desenvolver um AVC:
- Retorna FALSE se: temperatura <= 39  || card <=118
- Retorna TRUE se: temperatura > 39 && card > 118
Caso os sensores apresentam algum problema eles podem enviar um dado NEGATIVO este método
lança a exceção checada chamada InvalidArgumentException.
*/
 public boolean checkRisk (double temp, int card) throws InvalidArgumentException {
    …
 }
}
```

PROBLEMA 2: Considere o metodo abaixo que converte temperaturas de Celsius para Fahrenheit e vice versa. O codigo desta classe foi adicionado no SIGAA.

```
public class Temperature Transformer{
public Temperature convert(Temperature temp) throws Exception{
    …
 }
}
```

Responda as questões abaixo para os 2 problemas acima (1 de cada vez :))

1. Como você pode particionar o domínio de entrada do método acima? Em outras palavras quais classes de equivalência podem ser definidas para este método?

2. Liste os casos de teste são necessários para cobrir todas as classes.

3. Os testes que você criou acima atendem ao critério de Análise de Valor Limite? Caso não atendam quais seriam os novos casos de testes que você precisaria criar para atender este critério e por quê?

4. Construa uma Classe de Testes Parametrizados JUnit para os casos de testes definidos acima (considerando apenas os casos de testes que não devem lançar exceção).

5. Construa uma Classe de Testes Parametrizados JUnit agora considerando parâmetros que lancem exceção.