1- Faça o que se pede:

- Explique que é uma Exceção no contexto de programação com Java.
- Explique que significa a Manipulação ou Tratamento de Exceções.
- Cite alguns exemplos de Classes de Exceções do Java.
- Quais os tipos e diferenças de Exceções do Java. Qual a diferença entre uma exceção verificada e uma exceção não verificada?
- Explique o bloco de Tratamento de Exceções (try...catch...finally...).
- Explique a forma de lançamento de uma Exceção (throw new...).
- Explique o funcionamento da Propagação de Exceções (throws).

2- Nesta questão você deve identificar as partes problemáticas do código e reescrevê-lo utilizando tratamento de exceções. Ou seja, devem ser identificadas todas as exceções que podem ser levantadas e, para cada uma, deve ser dado o tratamento adequado que, nesse exercício, significa alertar o usuário quanto ao problema. Entretanto, nesse programa a leitura dos valores deve ser feita, mesmo que para isso o usuário tenha que tentar informar várias vezes os valores na mesma execução do programa.

```
public class Questao2 {

    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        System.out.println("Eu sei dividir!");
        System.out.print("Informe o primeiro valor: ");
        int x = teclado.nextInt();
        System.out.print("Informe o segundo valor: ");
        int y = teclado.nextInt();
        double r = (x / y);
        System.out.println("O resultado da soma é " + r);
    }
}
```

3- Crie uma classe que aceite a digitação de dois números e faça a divisão entre eles exibindo seu resultado. Sua classe deve tratar as seguintes exceções:

- ArithmeticException quando ocorrer uma divisão por zero

- InputMismatchException quando o valor informado não é numerico.

4- Crie um programa que leia um número do usuário, e chame uma função que valide se é maior ou menor que 10. Se for menor do 10 chamar uma exceção disparando uma exceção que número inválido.

5- Crie um programa que receba n números e some esses numeros enquanto a soma não for superior a 100. O programa deverá imprimir o valor somado (antes de atingir o número maior que 100) e deverá informar quantos números foram somados e qual a média. Refaça seu programa utilizando as seguintes regras:

- Utilize os tratamentos de exceção para lidar com a entrada de dados.
- Quando a soma for superior a 100, o programa deverá gerar uma exceção criada pelo programador com nome ExcecaoAcimaDeCem.
- Lance essa exceção, com o uso de throws.

6- Suponha que o método "saca" da classe Conta vai ser rescrito de forma a lançar uma exceção criada por você, cuja classe é ContaExcecao (extends Exception). A exceção é lançada sempre que o saldo da conta for inferior ao valor sacado. Implemente a classe ContaExcecao. Implemente o método saca que lança a exceção. E rescreva o código da caixa com o devido tratamento da exceção.

```
Conta minhaConta = new  Conta();
minhaConta.deposita(100 );
minhaConta.setLimite(100 );
minhaConta.saca(1000 );
```

7- Considere o problema de conversão de temperaturas entre Celsius e Fahrenheit em Java.

- Crie duas exceções, TemperatureException que herda de Exception, com mensagem: "Temperatura Inválida".
- Crie uma classe Temperatura com um método para converter temperaturas em graus de graus Fahrenheit (toCelsius(double fahrenheit)) ambas representadas em ponto-flutuante (double).
- Porém, caso o valor a ser convertido seja menor ou igual a zero absoluto (-459,67°F) deve-se lançar a exceção TemperatureException. A formula para conversão é dada por:

$$C = \frac{5(F - 32)}{9}$$

- No main criar estrutura de try, catch. Crie um finally, para printar e temperatura convertida mesmo inválida.

8- Construa uma classe chamada “ContaCorrente” , com os atributos “limite” que armazena a quantidade atual disponível do limite da conta que o usuário possui , o atributo “saldo” que é o valor que realmente é pertencente ao usuário, e o atributo “valorLimite” que consiste no valor máximo que o banco lhe fornece como valor de limite, todos float.

- Construa os métodos public void, sacar(float valor), depositar(float valor), e setValorLimite(float valor).
- Na construção dos três métodos faça com que eles lancem exceptions relacionados aos argumentos, por exemplo, sacar, depositar ou setar um valor negativo para esses eventos. Exceção: ValueInvalidException, herdando de Exception.
- Lance também uma exception no caso de tentar sacar um valor maior que o possível. ImpossibleOperationException, herdando de Exception
- Crie uma classe main, onde se deve instanciar um objeto da classe “ContaCorrente”, usar os três métodos construídos acima e tratar ás exceções propostas com os comandos “try” e “catch”, no catch imprimir na tela o método “getMessage()” da exception declarada no catch.
- Use o finally para printar o valor do saldo da conta obrigatoriamente