

# A Importância do desenvolvimento orientado a tipo na prevenção de erros operacionais

**Programa de Pós-graduação de Sistemas em Informação da Universidade de São Paulo - PPgSI-USP**

Pesquisador Jean Pierre de Brito  
Orientador Daniel Cordeiro

O desenvolvimento orientado a tipo é uma abordagem no desenvolvimento de software que envolve a definição clara e estruturada dos tipos de dados utilizados no código-fonte. Em Haskell, uma linguagem de programação funcional, essa abordagem ganha destaque.

Ela busca prevenir erros operacionais por meio da especificação precisa dos tipos, evitando incompatibilidades e inconsistências que podem comprometer a funcionalidade e confiabilidade dos sistemas e aplicativos. Essa prática oferece diretrizes práticas para melhorar a qualidade e a confiabilidade dos programas, contribuindo para evitar falhas de funcionamento, perda de dados e prejuízos financeiros.

Desenvolvimento orientado a tipo como abordagem para prevenção de erros operacionais em sistemas e aplicativos, estudos de caso e diretrizes práticas.

O desenvolvimento orientado a tipo é uma abordagem que envolve a definição clara e estrita dos tipos de dados utilizados no código-fonte, o que pode **reduzir a possibilidade de erros causados por incompatibilidade ou inconsistência** de tipos.

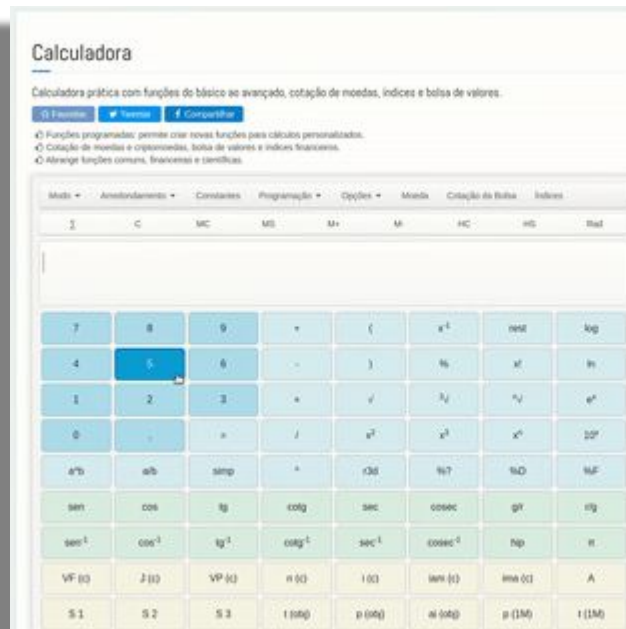
A prevenção de erros operacionais em sistemas e aplicativos é uma tarefa crítica e desafiadora, uma vez que esses erros podem levar a falhas de funcionamento, **perda de dados e prejuízos financeiros**.

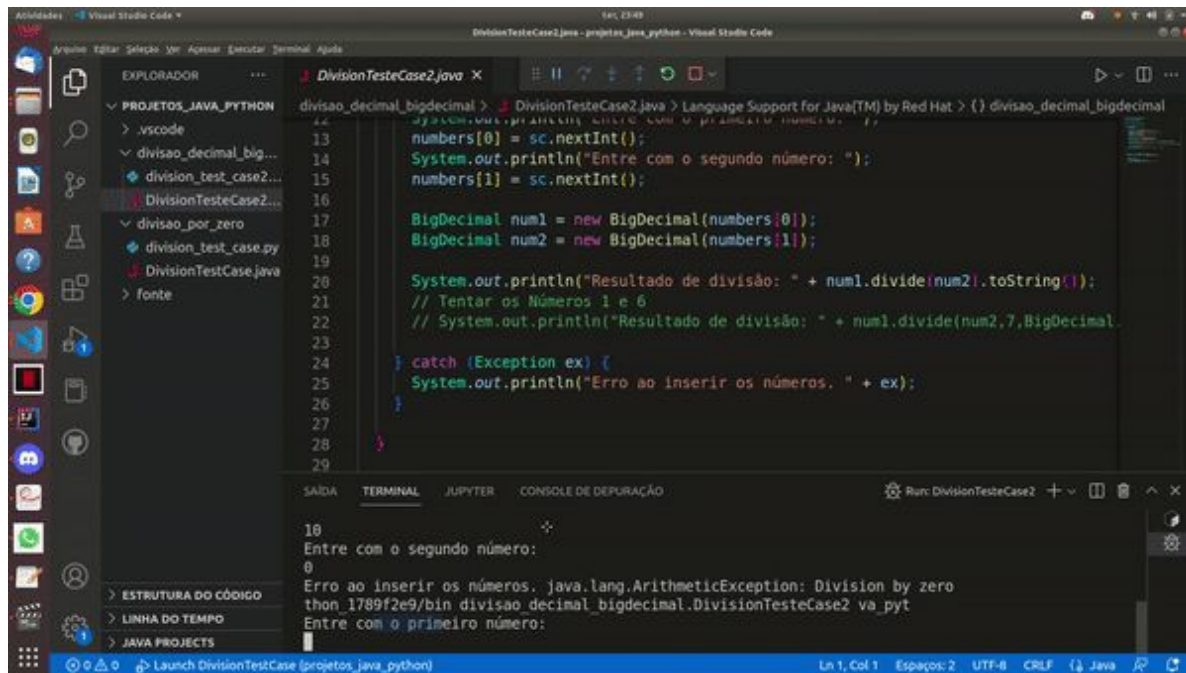
O desenvolvimento orientado a tipo é uma abordagem que pode ser utilizada para prevenir esses erros e **garantir a qualidade e a confiabilidade dos sistemas** e aplicativos desenvolvidos. Essa abordagem envolve a definição clara e estrita dos tipos de dados utilizados no código-fonte, o que reduz a possibilidade de erros causados por incompatibilidade ou inconsistência de tipos.

## Raiz Quadrada de Número negativo



## Divisão por zero



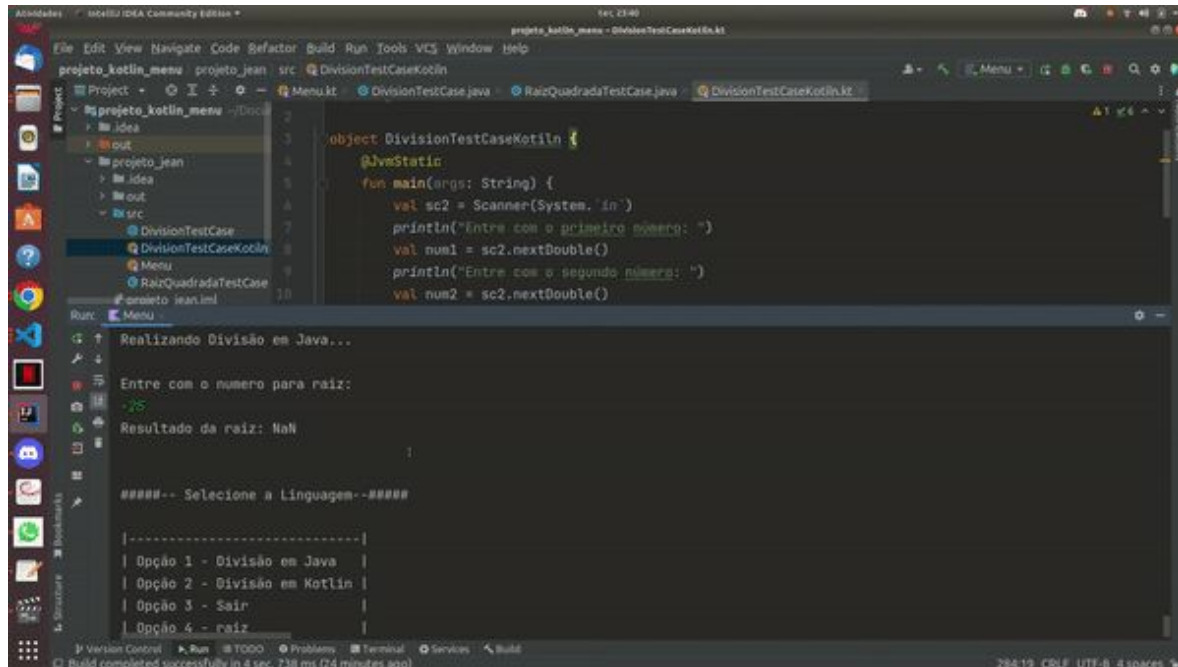


The screenshot shows the Visual Studio Code interface. The Explorer on the left lists a project named 'PROJETOS\_JAVA\_PYTHON' containing several files, including 'DivisionTesteCase2.java'. The main editor displays the code for this file, which uses `BigDecimal` for division and includes a try-catch block for `ArithmeticException`. The bottom panel shows the 'TERMINAL' output, which indicates a runtime error: `java.lang.ArithmeticException: Division by zero`, triggered by the program's attempt to divide by zero.

```
DivisionTesteCase2.java
13  divisao_decimal_bigdecimal > DivisionTesteCase2.java > Language Support for Java(TM) by Red Hat > {} divisao_decimal_bigdecimal
14  numbers[0] = sc.nextInt();
15  System.out.println("Entre com o segundo número: ");
16  numbers[1] = sc.nextInt();
17
18  BigDecimal num1 = new BigDecimal(numbers[0]);
19  BigDecimal num2 = new BigDecimal(numbers[1]);
20
21  System.out.println("Resultado de divisão: " + num1.divide(num2).toString());
22  // Tentar os Números 1 e 6
23  // System.out.println("Resultado de divisão: " + num1.divide(num2,7,BigDecimal.ROUND_HALF_UP));
24
25  } catch (Exception ex) {
26  System.out.println("Erro ao inserir os números. " + ex);
27  }
28
29  }
```

TERMINAL

```
10
11
12  Entre com o segundo número:
13  0
14
15  Erro ao inserir os números. java.lang.ArithmeticException: Division by zero
16  thon_l789f2e9/bin divisao_decimal_bigdecimal.DivisionTesteCase2 va pyt
17  Entre com o primeiro número:
```



```
object DivisionTestCaseKotlin {  
    @JvmStatic  
    fun main(args: String) {  
        val sc2 = Scanner(System.`in`)  
        println("Entre com o primeiro numero: ")  
        val num1 = sc2.nextDouble()  
        println("Entre com o segundo numero: ")  
        val num2 = sc2.nextDouble()  
    }  
}
```

Realizando Divisão em Java...

Entre com o numero para raiz:  
25  
Resultado da raiz: NaN

-----  
Opção 1 - Divisão em Java  
Opção 2 - Divisão em Kotlin  
Opção 3 - Sair  
Opção 4 - raiz

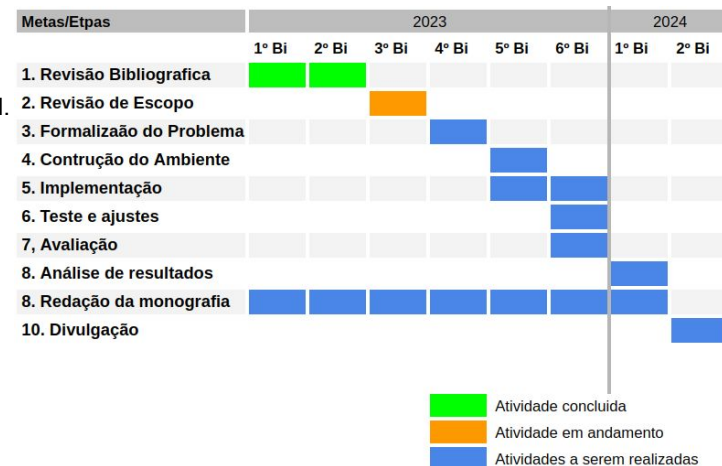
Analisar o desenvolvimento orientado a tipo como uma abordagem eficaz para prevenir erros operacionais em sistemas e aplicativos. Para isso, serão investigadas as principais causas de erros operacionais e como o desenvolvimento orientado a tipo pode ajudar a mitigá-las. Serão realizados estudos de caso e análises empíricas para avaliar a eficácia da abordagem em diferentes tipos de sistemas e aplicativos. Com base nos resultados obtidos, serão propostas diretrizes e boas práticas para a aplicação do **desenvolvimento orientado a tipo na prevenção de erros operacionais**. Além disso, a dissertação contribuirá para o avanço do conhecimento sobre o desenvolvimento orientado a tipo e sua relevância para a qualidade do software por meio de uma análise crítica e revisão da literatura existente.



A metodologia proposta para alcançar o objetivo da dissertação será baseada em uma revisão sistemática da literatura, na qual será realizada uma busca e análise de estudos sobre o desenvolvimento orientado a tipo para evitar e mitigar erros operacionais.

Atividades a serem realizadas:

1. **Revisão bibliográfica: estudo exploratório** dos fundamentos da programação funcional , técnica da programação orientada a tipo.
2. **Revisão de Escopo: revisão sistemática** para obter estudos recentes da programação funcional.
3. **Formalização do problema:** formalização do contexto, limitações e **regras** do problema.
4. **Construção do ambiente:** construção **lógica** do ambiente.
5. **Implementação:** implementação do ambiente, **algoritmos** adotados e **objetivos** para análise.
6. **Testes e ajustes:** testes referentes a possíveis **falhas** e ajustes de possíveis erros de implementação.
7. **Avaliação:** avaliação sobre os resultados adquiridos do experimento.
8. **Análise de resultados:** análise comparativa de algoritmos e dos pesos sobre objetivos.
9. **Redação da monografia:** escrita da monografia.
10. **Divulgação:** divulgação da pesquisa produzida nos meios de comunicação acadêmicos.



Ao desenvolver uma dissertação sobre programação orientada a tipo em Haskell, é possível realizar avaliações objetivas para verificar sua eficácia. Essas avaliações incluem ***avaliação de desempenho, análise de erros, comparação de manutenibilidade e estudo de caso.***