UNIVERSIDADE SALVADOR

Projeto Semestral de Avaliação A3

Unidade Curricular

ESTRUTURAS DE DADOS E ANALISE DE ALGORITMOS

CALCULADORA DE NÚMEROS COMPLEXOS

Componentes do Grupo

Vinícius de Jesus Rocha Reis – RA: 12724120214

João Paulo Souza Fontes – RA: 12724113272

Professor Orientador

Prof.º Wellington Lacerda Silveira da Silva

Salvador - BA
2025.2

Repositório no GitHub:
https://github.com/Vinicius-Jreis/A3---Calculadora-de-Numeros-Complexos

1. INTRODUÇÃO

Este projeto tem como objetivo desenvolver uma calculadora que funciona diretamente pelo terminal, sendo capaz de interpretar operações com números complexos, criar uma árvore sintática e calcular o resultado das expressões.

Assim, sendo possível fazer diversos cálculos relacionados a números complexos, incluindo: conjugação, potenciação, e até mesmo usar variáveis e comparação entre expressões. Logo, promovendo o entendimento das estruturas de dados e da manipulação de expressões matemáticas.

2. OBJETIVOS DO SISTEMA

OBJETIVO GERAL

A calculadora de números complexos deve entender expressões matemáticas e construir uma árvore sintática em LISP e resolvê-las entregando o resultado da operação.

OBJETIVOS ESPECIFICOS

- Interpretar expressão caractere a caractere
- Construir árvore de expressão através de um parser recursivo
- Tratar diversos tipos de erros
- Realizar operações simples (soma, subtração, multiplicação, divisão)
- Realizar operações especiais (potenciação, conjunção)
- Realizar operações com variáveis
- Possibilidade de comparação de expressões (expressão 1 = expressão 2)

3. TECNOLOGIAS UTILIZADAS

- Java 17+
- Eclipse 2025-09 (IDE)

4. ESTRUTURA DAS CLASSES

4.1 Classe NumComplexo

Responsável pelas operações matemáticas com números complexos, como adição, subtração divisão, multiplicação, potenciação e conjunção.

4.2 Classe ArvoreNo

Responsável pela estrutura da árvore representando o nó da árvore sintática, contendo o operador, seu valor numérico e as subarvores a esquerda e direita.

4.3 Classe ConstrutorArvore

Responsável pelo papel de Parser recursivo sendo capaz de:

- Interpretar a expressão vinda do terminal de caractere a caractere;
- Reconhecer valores numérico, operadores, variáveis e conjunção;
- Suportar operadores como +, -, /, *, ^, conj();
- Montar a árvore sintática completa;
- Utilizar estrutura de dados de pilhas para facilitar na construção da árvore.

4.4 Classe AvaliadorArvore

Responsável por executar a arvore sintática construída e calcular o resultado das expressões

- Avaliar os nós;
- Substituir variáveis se preciso com o uso de hashMap;
- Interpretar a arvore e aplicar as operações matemáticas;

4.5 Classe Calculadora

Responsável por controlar a interação do sistema diretamente com o usuário, possuindo:

- Menu inicial;
- Entrada da expressão (Menu interno: digitar expressão, mostrar árvore sintática, calcular resultado e comparar expressões) ;
- Exemplos de uso;
- Opção de encerramento do sistema com “sair”.

5. EXEMPLOS DE EXPRESSOES ACEITAS

- **Adição:**

  (2+3i) + (2+2i)

  Árvore LISP: (+ (+ 2 3i) (+ 2 2i))

  Resultado: 4 + 5i

- **Subtração:**

  (2+3i) - (4+5i)

  Árvore LISP: (- (+ 2 3i) (+ 4 5i))

  Resultado: -2 - 2i

- **Multiplicação:**

  (2+3i) * (2+i)

  Árvore LISP: (* (+ 2 3i) (+ 2 i))

  Resultado: 1 + 8i

- **Divisão:**

  (2+3i) / (3-3i)

  Árvore LISP: (/ (+ 2 3i) (- 3 3i))

  Resultado: -0,17 + 0,83i

- **Potenciação:**

  (2+i) ^3

  Árvore LISP: (^ (+ 2 i) 3)

  Resultado: 2 + 11i

- **Conjunção:**

  conj(3+5i)

  Árvore LISP: (conj (+ 3 5i))

  Resultado: 3 - 5i

6. TRATAMENTO DE ERROS

- **Divisão por zero**

  Exemplos: (2+i) / (5-5)

  Mensagem: "Erro: Divisão por zero."

- **Expressão vazia**

  Exemplo:

  Mensagem: Erro: expressão vazia.

- **Parênteses incorretos**

  Exemplo: 2+i)

  Mensagem: "Erro: Parênteses não balanceados."

7. CONCLUSÃO

O desenvolvimento da Calculadora de Números Complexos permitiu aplicar de forma prática os conceitos estudados na disciplina de Estruturas de Dados e Análise de Algoritmos, cumprindo todos os requisitos propostos:

- Interpreta expressões complexas e faz a construção da árvore sintática correspondente
- Avalia a expressão da árvore e calcula os resultados
- Realiza a comparação de igualdade entre duas expressões
- Realiza o tratamento de erros de divisão, expressões vazias e parênteses incorretos

O projeto demonstra que estruturas matemáticas e estruturas de dados caminham juntas na construção de programas mais robustos e eficientes, servindo como base para aplicações mais avançadas, como compiladores, interpretadores e sistemas simbólicos.