UNIVERSIDADE ESTADUAL PAULISTA

FACULDADE DE CIÊNCIAS E TECNOLOGIA

PROJETO PRÁTICO – PARTE 1

RELATÓRIO

LINGUAGENS FORMAIS E AUTÔMATOS

PROF. DR. CELSO OLIVETE JÚNIOR

BRUNO SANTOS DE LIMA

LEANDRO UNGARI CAYRES

PRESIDENTE PRUDENTE

JANEIRO - 2017

BRUNO SANTOS DE LIMA

LEANDRO UNGARI CAYRES

PROJETO PRÁTICO – PARTE 1

RELATÓRIO

Projeto prático parte 1 da disciplina de Linguagens Formais e Autômatos, lecionada pelo docente Dr. Celso Olivete Júnior, no curso Bacharelado em Ciência da Computação – Departamento de Matemática e Computação da Faculdade de Ciências e Tecnologia (FCT Unesp – Presidente Prudente).

PRESIDENTE PRUDENTE

JANEIRO – 2017

Sumário

[**1 INTRODUÇÃO** 4](#_Toc472161646)

[**2 EXPRESSÃO REGULAR** 4](#_Toc472161647)

[**2.1 Implementação do simulador de Expressão Regular** 8](#_Toc472161648)

[**3 AUTÔMATOS FINITOS** 8](#_Toc472161649)

[**3.1 Implementação do simulador de Autômatos Finitos** 9](#_Toc472161650)

[**4 GRAMÁTICA REGULAR** 9](#_Toc472161651)

[**4.1 Implementação do simulador de Gramática Regular** 9](#_Toc472161652)

# **1 INTRODUÇÃO**

Neste projeto da disciplina de Linguagens Formais e Autômatos o objetivo da parte 1 é desenvolver uma ferramenta na qual o usuário da mesma consiga trabalhar e simular Expressões Regulares (ER), Autômatos Finitos Determinísticos (AFD) e Não-Determinísticos (AFND) e Gramaticas Regulares (GR).

A ferramenta que realizará a simulação das ER’s, AFD’s, AFND’s e GR’s foi implementada utilizando a linguagem de programação Java, com a utilização do JavaFX para construção da interface gráfica, necessitando assim que o usuário utilize a versão 8 do Java. Como IDE foi utilizado o NetBeans, contando também com um sistema de controle de versão Git para versionamento do projeto.

Este relatório está divido como segue: na seção 2 é apresentado os conceitos de Expressão Regular bem como o funcionamento da ferramenta para este propósito, a seção 3 descreve Autômatos Finitos e apresenta o simulador de AFD e AFND, por fim a seção 4 discorre sobre Gramaticas Regulares mostrando o funcionamento da ferramenta para a simulação de GR’s.

# **2 EXPRESSÃO REGULAR**

Quando citamos Linguagens Regulares temos as chamadas Expressões Regulares (ER) que é uma notação para representar um determinado padrão de strings. As Expressões Regulares denotam Linguagens Regulares e são muito utilizadas em sistemas de processamento de strings, por exemplos em aplicações que utilizam pesquisas em textos.

Na ferramenta desenvolvida neste projeto temos um simulador de Expressão Regular, seu funcionamento é simples basta inserir a regra, ou seja, a Expressão Regular e em seguida informa a string, palavra de entrada, que a ferramenta irá informa se a palavra de entrada é válida de acordo com a Expressão Regular informada anteriormente. Abaixo temos um instantâneo da aplicação mostrando a tela na qual essa verificação, simulação, é utilizada.

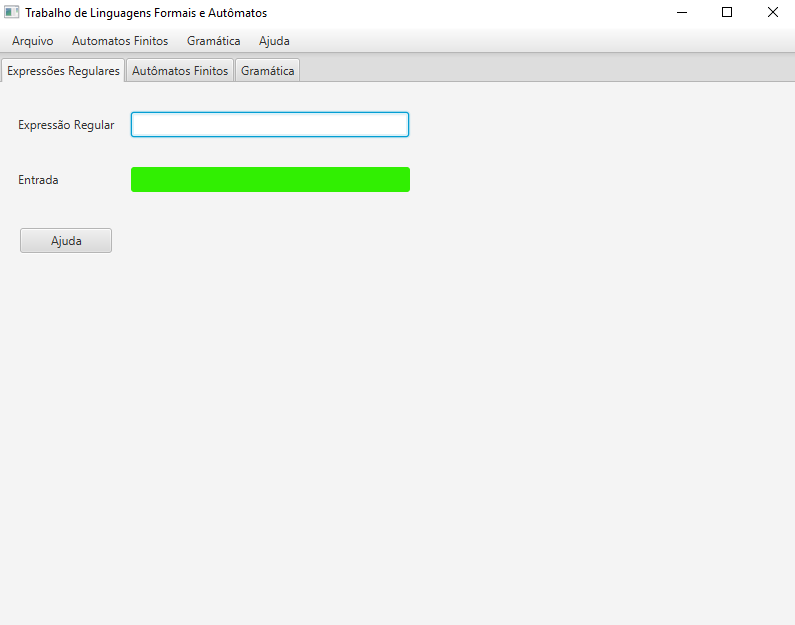


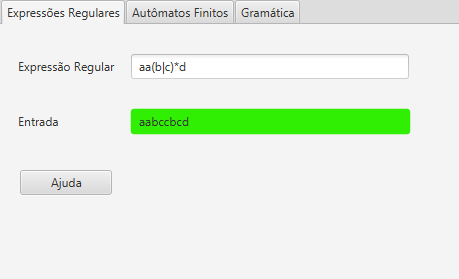
Figura 1 - Instantâneo da ferramenta na tela de Expressões Regulares

Para exemplificar como a ferramenta atua vamos definir uma Expressão Regular (ER) e em seguida fazer um teste com duas palavras de entrada, uma delas pertence a Expressão Regular (ER) enquanto a outra não é válida pela mesma. Exemplo:

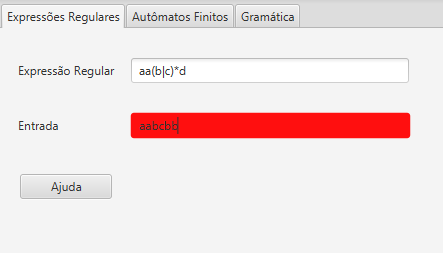
|  |  |
| --- | --- |
| Expressão Regular (ER) | aa(b|c)\*d |
| Palavra de Entrada 1 | aabccbcd |
| Palavra de Entrada 2 | aabcbb |

A Expressão Regular (ER) definida como exemplo permite as strings pertencentes a ela comecem obrigatoriamente com a ocorrência de **aa**, seguida de qualquer quantidade da letra **b** ou da letra **c**, inclusive nenhuma, terminando obrigatoriamente com a letra **d**.

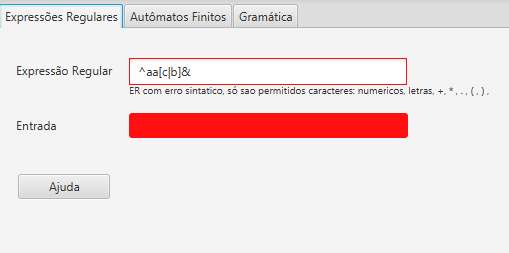
Observe no instantâneo de exemplo 1 que a Expressão Regular foi definida e que a palavra de entrada 1 (**aabccbcd**) está de acordo com a ER informada, portando está string é válida perante a ER estabelecida, com isso a ferramenta destaca a palavra de entrada 1 na cor verde.



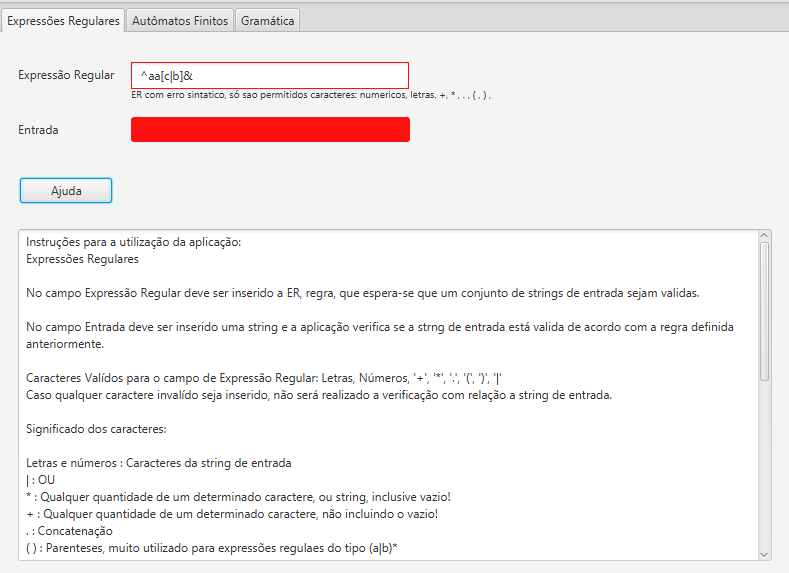
Agora observe no instantâneo de exemplo 2 que a Expressão Regular foi definida e que a palavra de entrada 2 está de incorreta com relação a ER informada, pois a ER define como regra que toda string pertencente a ela deve obrigatoriamente terminar com a letra **d** e no caso a palavra de entrada 2 (**aabcbb**) não termina com a letra **d**, portando está string é invalida perante a ER estabelecida, com isso a ferramenta destaca a palavra de entrada 2 na cor vermelha.



Além disso a aplicação também informa caso a Expressão Regular esteja sintaticamente escrita de forma errada, ou seja, se os caracteres inseridos na ER não são validos o usuário é informado com uma mensagem e um alerta de cor vermelha no campo de entrada da Expressão Regular, observe o instantâneo abaixo com a exemplificação de um erro de sintaxe na ER.



Ainda para facilitar a utilização da ferramenta, existe um botão de ajuda onde é explicado como deve ser utilizada a ferramenta com relação a ER, quais são os caracteres validos, exemplos e outras informações que podem ajudar o usuário, observe o instantâneo abaixo:



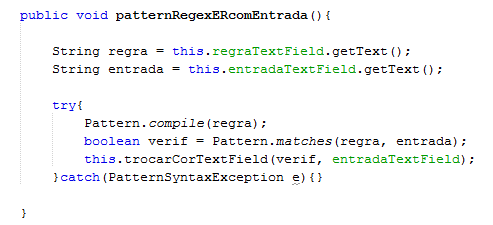
Os caracteres válidos para o campo de Expressão Regular são somente: Letras, Números, '+', '\*', '.', '(', ')', '|', Observe abaixo o significado desses caracteres abaixo:

|  |  |
| --- | --- |
| Caractere | Significado |
| Letras | Letras do alfabeto de a até z |
| Números | De 0 até 9 |
| | | Ou |
| \* | Qualquer quantidade de um determinado caractere, inclusive vazio! |
| + | Qualquer quantidade de um determinado caractere, não incluindo o vazio! |
| . | Concatenação |
| ( ) | Parênteses, muito utilizado para expressões regulares do tipo (a|b)\* |

## **2.1 Implementação do simulador de Expressão Regular**

A implementação do simulador de Expressão Regular foi desenvolvida de maneira bem simples utilizando os recursos oferecidos pela linguagem de programa Java.

Observamos abaixo o método que realiza a verificação, basicamente é utilizado o próprio *regex* nativo do Java como sugerido pelo professor, onde no caso são coletados dois dados de entrada, um dado chamado de regra (Expressão Regular) e outro de entrada (palavra de entrada). A regra é compilada por *Pattern.compile* do Java e posteriormente é verificado se a palavra de entrada respeita a regra definida que faz essa ação é o *Pattern.matches* do java retornando um *boolean* com a resposta da verificação, o restante é a parte da interface gráfica, utilizando o JavaFX que realiza.



# **3 AUTÔMATOS FINITOS**

## **3.1 Implementação do simulador de Autômatos Finitos**

# **4 GRAMÁTICA REGULAR**

## **4.1 Implementação do simulador de Gramática Regular**