

Trabalho desenvolvido para a disciplina DCC146 - Aspectos Teóricos da computação durante o semestre de 2022.1

#### **Grupo:**

Denner Efisio Emanuel Reis - 201735008 Thomas Santos - 201776034 Igor Westermann Lima - 201876021

#### **Professor:**

Gleiph Ghiotto Lima de Menezes



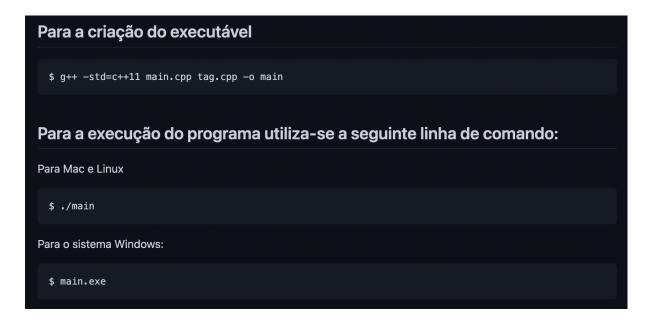
## Sumário

1. Buildando e executando o projeto .	PAG. 3
2. Organização do código	PÁG.3
3. Funcionamento	PÁG. 4
4. Testes	PÁG 5



# Buildando e executando o projeto:

No terminal, execute os seguintes passos:



## Organização geral do código:

O código foi dividido entre os arquivos main.cpp e tag.cpp. No arquivo main, acontece a distribuição de tarefas da aplicação relacionada a interação do usuário com o terminal. Já na classe Tag, ocorre a execução das tarefas de listagem, criação, validação, importação e exportação das tags.



### Funcionamento geral do código:

#### Cadastramento da TAG

Ao inserir uma tag, seguindo os padrões estipulados no trabalho de "nome: descrição", é realizada a verificação do nome, para que não exista duas tags com o mesmo nome. Além disso, é validado se existe um espaço antes dos :, fazendo com que a tag seja invalidada. Caso os dois testes sejam aprovados, a descrição é adicionada em um processo de validação para saber se a expressão regular é válida.

A validação, ocorre com auxílio de uma pilha. Sendo assim, o sistema avalia caracter a caracter tendo ações diferentes dependendo do que foi encontrado

- Fecho de Kleene (\*) Ao receber o fecho, o sistema desempilha o topo da pilha, e realiza a concatenação do que foi desempilhado com o fecho. No entanto, caso não exista nenhum elemento na pilha o sistema retorna um erro ao usuário mostrando que a TAG está inválida.
- + ou . Ao receber um dos dois elementos, o sistema utiliza duas variáveis auxiliares e inicia o processo de desempilhar os últimos dois elementos da pilha. Caso o tamanho da pilha seja menor que dois elementos, antes de desempilhar, novamente o sistema retorna um erro ao usuário mostrando que a TAG está inválida. Para realizar a junção dos dois auxiliares, é necessário pegar o primeiro elemento, juntar com o + ou. e depois juntar com o segundo elemento, para que depois seja adicionado na pilha.
- \ A contra barra é utilizada como escape, então quando o usuário a utiliza, é necessário analisar o elemento que vem posterior a ela. Sendo assim, o sistema realiza uma acréscimo de 1 no loop para que seja empilhado os dois elementos de forma simultânea.
- Os outros caracteres são apenas adicionados na pilha.

Ao final da execução, é necessário validar o tamanho da pilha, pois caso ela tenha um tamanho superior a 1, deve ser retornado um erro ao usuário de TAG inválida.

- Divisão de Tags em um arquivo informado (:d arquivo)
  No momento, o comando ainda não foi implementado, então caso o usuário tente utilizá-lo, será exibido um aviso.
- Carregar um arquivo com as tags (:c arquivo)
  Ao executar o comando, o sistema busca todas as tags de um arquivo definido pelo usuário no comando.
  - Especificar o caminho das saídas para a divisão (:O arquivo)
    No momento, o comando ainda não foi implementado, então caso o usuário tente utilizá-lo, será exibido um aviso.
  - Realizar a divisão em tags (:p tag)



No momento, o comando ainda não foi implementado, então caso o usuário tente utilizá-lo, será exibido um aviso.

• Listar as definições formais dos autômatos (:a)

No momento, o comando ainda não foi implementado, então caso o usuário tente utilizá-lo, será exibido um aviso.

• Listar as Tags (:I)

Ao executar o comando, todas as tags válidas digitadas anteriormente pelo usuário são exibidas no terminal

- Sair do Programa (:q)
  Comando para finalizar a execução do programa
- Salvar as Tags (:s arquivo)

Ao executar o comando, todas as tags válidas são armazenadas em um arquivo, passado pelo usuário

#### **Testes realizados:**

- INT: 9+4+2+5+4+\* (não será aprovado) Ao tentar remover da pilha dos elementos no primeiro "+", só terá o 9, fazendo com que não seja possível a operação
- VAR: ab+cd+a+dc+ (não será aprovado) A pilha irá terminar com mais de 1 elemento
- VAR2: ae+ea+dd+ca.\* (não será aprovado) A pilha irá terminar com mais de 1
- VAR3: ae+ea+dd+ca.\* (não será aprovado) A pilha irá terminar com mais de 1 elemento
- INT2: 01+2+3+4+5+6+7+8+9+01+2+3+4+5+6+7+8+9+\*.
- AB: a\*b\* (não será aprovado) Ao final a pilha irá conter dois elementos
- PAR: ab.ba.+aa.+bb.+\*