para testar o compilador digite o seguinte comando após compilar e construir o projeto no netbeans:

java -jar dist/CompiladorAscanio.jar test/teste_x.l

classes utilizadas na implementação:

CompiladorAscanio - classe principal do projeto, contem, o método main, que chama o Lexer, busca os tokens e adiciona-os na tabela de símbolos

Env - classe que contém a tabela de símbolos que é uma hashtable de chave Token e de conteúdo Id, que inicialmente contém somente o lexema do token inserido mesmo.

Id - classe que descreve o conteúdo dos identificadores da tabela de símbolo - inicialmente possui somente o lexema do token.

Token - superclasse padrão que representa um token gerado pelo analisador léxico, contendo um campo tag, que é a tag correspondente a palavra reservada da linguagem.

Tag - classe que específica as tags das palavras reservadas e operadores da linguagem

Word - sub classe de Token para específicar identificadores gerais (statements, variáveis, etc)

FloatConst - sub classe de Token para especificar constantes reais

IntegerConst - sub classe de Token para especificar constantes inteiras

Testes:

os arquivos fontes para serem testados, estão na pasta test e possuem a extensão .l, a saída do compilador para a execução desses testes também foi gravada na pasta teste, como arquivos de texto no formato teste_x.s

Exemplo da saída do teste do programa 1:

```
Sequencia de tokens identificadas:
256, app
285, teste1
285, int
285, a
44, 44
285, b
44, 44
285, c
59, 59
285, int
285, result
257, start
269, read
40, 40
285, a
41, 41
59, 59
269, read
40, 40
285, c
41, 41
59, 59
285, b
287, :=
284, 10.23
43, 43
285, a
43, 43
285, c
59, 59
285, result
287, :=
40, 40
285, a
43, 43
285, b
41, 41
```

```
45, 45
59, 59
270, write
40, 40
285, result
41, 41
59, 59
285, teste1
287, :=
283, 1
258, stop
65535, 65535
Simbolos instalados na tabela de simbolos:
44, 44
1, 1
59, 59
59, 59
40, 40
45, 45
65535, 65535
44, 44
app, app
40, 40
teste1, teste1
59, 59
write, write
int, int
a, a
41, 41
:=, :=
41, 41
59, 59
43, 43
40, 40
43, 43
59, 59
read, read
40, 40
stop, stop
10.23, 10.23
result, result
59, 59
41, 41
b, b
43, 43
41, 41
c, c
start, start
```