ECOTE - Project tests

Semester: 2022L

Author: Jakub Tomkiewicz

Subject: Constructing DFA using syntax tree for given regular expressions

General overview

The aim of this project is to write a program that will create directly deterministic finite automata using syntax free procedure for regular expression given by the user. The user will be able to enter input strings and check if the inputs can be generated by the regular expression typed earlier.

Test case #1

Correct regular expression a* and two input strings: correct aaa and incorrect b.

Input:
```
What would you like to do?
0 - Convert regex to DFA
1 - exit
Input: 0
Input regex: a*
```

Output:
```
SYNTAX TREE WITHOUT FUNCTIONS:
CAT
|
|___STAR___a

SYNTAX TREE WITH FUNCTIONS:
CAT first_pos(1,2) last_pos(2)
|
|___STAR first_pos(1) last_pos(1)___a first_pos(1) last_pos(1)

TRANSITION TABLE:
      STATE     a
BEGIN 1       | 1 | FINAL
```

Input:
```
What would you like to do?
0 - Input string to check, if it can be generated by regex
1 - exit
Input: 0
Input string: aaa
```

Output:
```
aaa can be generated by regex
```
Input:
```
What would you like to do?
0 - Input string to check, if it can be generated by regex
1 - exit
Input: 0
Input string: b
```

Output:
```
b cannot be generated by regex
```

Input:
```
What would you like to do?
0 - Input string to check, if it can be generated by regex
1 - exit
Input: 1
```

Test case #2

Correct regular expression a|b|c and two input strings: incorrect dd and correct b.

Input:
```
What would you like to do?
0 - Convert regex to DFA
1 - exit
Input: 0
Input regex: a|b|c
```

Output:
```
SYNTAX TREE WITHOUT FUNCTIONS:
CAT
|
|___OR
|   |
|   |___OR
|   |   |
|   |   |___c

SYNTAX TREE WITH FUNCTIONS:
CAT first_pos(1,2,3) last_pos(4)
|
|___OR first_pos(1,2,3) last_pos(1,2,3)
|   |
|   |___OR first_pos(2,3) last_pos(2,3)
|   |   |
|   |   |___c first_pos(3) last_pos(3)

TRANSITION TABLE:
      STATE     a   b   c
BEGIN 1       | 2 | 2 | 2 |
      2       |   |   |   | FINAL
```

Input:
```
What would you like to do?
0 - Input string to check, if it can be generated by regex
1 - exit
Input: 0
Input string: dd
```
Output:
```
dd cannot be generated by regex
```

Input:
```
What would you like to do?
0 - Input string to check, if it can be generated by regex
1 - exit
Input: 0
Input string: b
```

Output:
```
b can be generated by regex
```

Input:
```
What would you like to do?
0 - Input string to check, if it can be generated by regex
1 - exit
Input: 1
```

Test case #3

Correct but tricky regular expression (((a)*)*)* and no input strings.

Input:
```
What would you like to do?
0 - Convert regex to DFA
1 - exit
Input: 0
Input regex: (((a)*)*)*
```

Output:
```
SYNTAX TREE WITHOUT FUNCTIONS:
CAT
|
|___STAR___STAR___STAR___a

SYNTAX TREE WITH FUNCTIONS:
CAT first_pos(1,2) last_pos(2)
|
|___STAR first_pos(1) last_pos(1)___STAR first_pos(1) last_pos(1)___STAR
first_pos(1) last_pos(1)___a first_pos(1) last_pos(1)

TRANSITION TABLE:
      STATE     a
BEGIN 1       | 1 | FINAL
```

Input:
```
What would you like to do?
0 - Input string to check, if it can be generated by regex
1 - exit
Input: 1
```

Test case #4

Incorrect regular expression (((a)*)* with too many opening parentheses.

Input:
```
What would you like to do?
0 - Convert regex to DFA
1 - exit
Input: 0
Input regex: (((a)*)*
```

Output:
```
Traceback (most recent call last):
  File "/Users/jtomkiewicz/Desktop/ECOTE/main.py", line 35, in <module>
    main()
  File "/Users/jtomkiewicz/Desktop/ECOTE/main.py", line 9, in main
    alphabet, regex = menu.read_regex()
  File "/Users/jtomkiewicz/Desktop/ECOTE/menu.py", line 32, in 
read_regex
    is_regex_correct(regex)
  File "/Users/jtomkiewicz/Desktop/ECOTE/menu.py", line 62, in 
is_regex_correct
    raise Exception(
Exception: Given regex contain different number of closing and opening 
parentheses!
```

Test case #5

Correct regular expression (&b|%c)* that contain characters & and % that are being ignored. No input strings.

Input:
```
What would you like to do?
0 - Convert regex to DFA
1 - exit
Input: 0
Input regex: (&b|%c)*
```

Output:
```
SYNTAX TREE WITHOUT FUNCTIONS:
CAT
|
|___STAR___OR
|       |
|       |___c

SYNTAX TREE WITH FUNCTIONS:
CAT first_pos(1,2,3) last_pos(3)
|
|___STAR first_pos(1,2) last_pos(1,2)___OR first_pos(1,2) last_pos(1,2)
|       |
|       |___c first_pos(2) last_pos(2)

TRANSITION TABLE:
      STATE     b   c
BEGIN 1       | 1 | 1 | FINAL
```

Input:
```
What would you like to do?
0 - Input string to check, if it can be generated by regex
1 - exit
Input: 1
```

Test case #6

Exiting application at the beginning.

Input:
```
What would you like to do?
0 - Convert regex to DFA
1 - exit
Input: 1
```

Test case #7

Correct regular expression a(b|c|d)* with four parameters and no input string.

Input:
```
What would you like to do?
0 - Convert regex to DFA
1 - exit
Input: 0
Input regex: a(b|c|d)*
```

Output:

```
SYNTAX TREE WITHOUT FUNCTIONS:
CAT
|
|___CAT
|   |
|   |___STAR___OR
|   |       |
|   |       |___OR
|   |       |   |
|   |       |   |___d

SYNTAX TREE WITH FUNCTIONS:
CAT first_pos(1,2,3,4) last_pos(5)
|
|___CAT first_pos(1,2,3,4) last_pos(1)
|   |
|   |___STAR first_pos(2,3,4) last_pos(2,3,4)___OR first_pos(2,3,4)
last_pos(2,3,4)
|   |       |
|   |       |___OR first_pos(3,4) last_pos(3,4)
|   |       |   |
|   |       |   |___d first_pos(4) last_pos(4)

TRANSITION TABLE:
      STATE     a   b   c   d
BEGIN 1       | 2 | 1 | 1 | 1 |
      2       |   |   |   |   | FINAL
```

Input:

```
What would you like to do?
0 - Input string to check, if it can be generated by regex
1 - exit
Input: 1
```

Summary

Of the seven test cases, six performed flawlessly, and their inputs were correct. The transition table is right in the last, seventh test case, but the tree was cut when printing.