Benjamin Scholar **Problem Set - 2**
Email: scholar@case.edu ID: ??
Course: CSDS 337 - Compiler Design Term: Spring 2023
Instructor: Dr. Vipin Chaudhary Due Date: 15th February, 2023

Number of hours delay for this Problem Set: 0
Cumulative number of hours delay so far: 0

I discussed this homework with: Zachary Baldwin, Matthias Portzel

SUBMISSION GUIDELINES: Submit a zip file that includes the written answers and the flex file for Problem 4.

Problem 1 - 5 points
Describe the language denoted by the following regular expression?
$(aa|bb)^*((ab|ba)(aa|bb)^*(ab|ba)(aa|bb)^*)^*$

Solution: Your solutions goes here

The regex can be simplified down to the following: $((aa|bb)^*(ab|ba))^*$

This language describes a string consisting of a sequence of zero or more 'a's and 'b's followed by a single instance of "ab" or "ba". This pattern can repeat 0 or more times.

This language describes a syntax that generates a string containing sequences of odd numbers of "a" and "b" characters.

Problem 2 - 35 points
Write regular definitions for the following languages:

a All strings of lowercase letters that contain the five vowels in reverse order.

b Binary strings that has at least 3 characters, and the third character is 0.

c Binary strings that has number of 0s which is a multiple of 3

d Binary strings that starts and ends with the same character

e Binary strings that has odd length

f Binary strings that starts with 0 and has odd length, or starts with 1 and has even length

g Binary strings whose length is at least 1 and at most 3

Solution: Your solutions go here

a $[\hat{}aeiou]^*u[\hat{}aeio]^*o[\hat{}aei]^*i[\hat{}ae]^*e[\hat{}a]a.^*$
(I am assuming the input is all lowercase letters, problem seems to imply this)

b (0|1)(0|1)0(0|1)*

c 1*((0(1*)){3})*

d (0(0|1)*0)|(1(0|1)*1)

e ((0|1)(0|1))*(0|1)

f (0((0|1)(0|1))*)|(1(0|1)((0|1)(0|1))*)

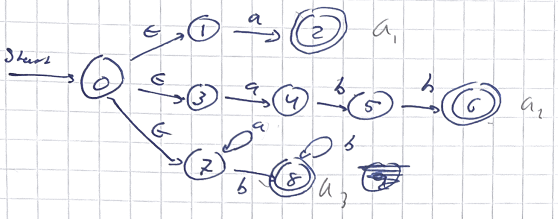
g (0|1){1, 3}

Problem 3 - 10 points
Provide transition diagram as an NFA to recognize the language represented by $a|abb|a^*b^+$. Convert this NFA to a DFA and show all steps.

Solution:

$a|abb|a^*b^+$

NFA



	NFA State	DFA State	a	b
ε-Closure(s0)	{0, 1, 3, 7}	A	B	C
	{2, 4, 7}	B	D	E
	{8}	C		C
	{7}	D	D	C
	{5, 8}	E		F
	{68}	F		C

{2, 4, 7}

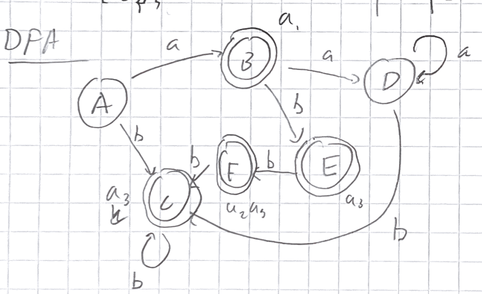
move({0, 1, 3, 7}, a) = {[illegible]}

ε-closure{2, 4, 7} =

move({0, 1, 3, 7}, b) = {8}

ε-Closure({8}) = {8}

DFA



Problem 4 - 50 points
Write a flex program which does the following:

- reads multiple input files
- for each file:

  - it prints the number of characters, number words and number of lines
  - it replaces more than one contiguous space by a single space
  - it prints the number of single line C comments
  - it prints the number of multiple line C comments
  - it prints the number of occurrences of each of these keywords: for, do, and $while$

- all the above counts are printed for each file in order and a cumulative number for all the files is also printed at the end
- the entire output is printed to a file named "problem4output"
- the output should clearly indicate what each of the count indicates
- the flex file should be named "problem4lex.l"

Assumptions and Comments:

I assumed that the words within the input files would contain alphanumeric characters as well as underscores. Limiting the input to these symbols simplified the problem significantly. If every character has to be handled the same way, the regular expressions become quite complicated (lookaheads would be required for comments, etc.) Plus, flex does not make it easy to make a regex that matches all characters besides whitespace + newline (the . operator only matches everything but newline).

I found that there are *a lot* of potential edge cases for this problem, so I tried to keep it simple.

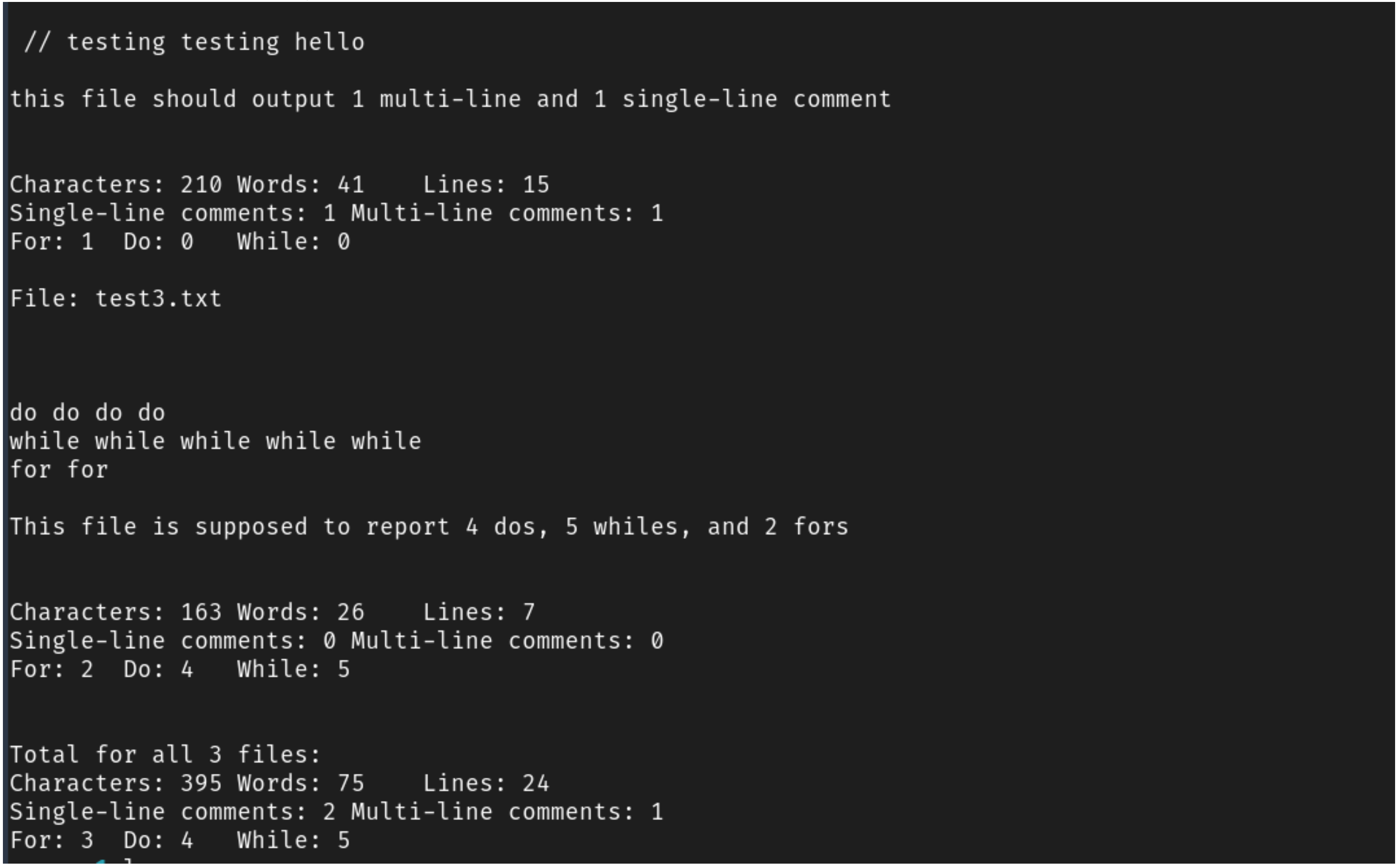
Keywords within comments are still counted.

All input files are modified and output to the same output file under their respective headers.

Comments nested within other comments are not counted.

The word count output by my program does not match up to the word count program, but I found it difficult to match without significant modifications to my flex program. Non alpha-numeric characters are handled as single words. Characters and lines seem to report similarly though.

Sample output for my 3 test files:



```
 // testing testing hello

this file should output 1 multi-line and 1 single-line comment


Characters: 210 Words: 41    Lines: 15
Single-line comments: 1 Multi-line comments: 1
For: 1  Do: 0   While: 0

File: test3.txt



do do do do
while while while while while
for for

This file is supposed to report 4 dos, 5 whiles, and 2 fors


Characters: 163 Words: 26    Lines: 7
Single-line comments: 0 Multi-line comments: 0
For: 2  Do: 4   While: 5


Total for all 3 files:
Characters: 395 Words: 75    Lines: 24
Single-line comments: 2 Multi-line comments: 1
For: 3  Do: 4   While: 5
```