CSCI 4430 Homework 1 - Hanson Ma

Problem 1

Part a

`a(a|b)*a`

This language can be separated into three languages concatenated with one another: - `a` : the character "a" - `(a|b)*` : "a" or "b" any number of times in any order - `a` : the character "a"

In completeness the regex described in this problem is "any sequence of 'a' or 'b' as long as it starts and ends with 'a'"

Part b

The language `(b*(e|a))*` (where `e` is the empty character) is "any sequence of 'a' and 'b', and the empty string"

Part c

The language `(a|b)*a(a|b)(a|b)` is "any sequence of 'a' and 'b' as long as the 3rd to last character is an 'a'"

Part d

All strings of 'a' and 'b', but with exactly three 'b's

Part e

Strings of 'a' and 'b' with even numbers of both

Problem 2

Part a

21 in base 2 is `10101`

```
 C -> A 1
C -> B 01
c -> B 101
C -> A 0101
C -> 10101
```

Part b

no

Part c

also no

Problem 3

Part a

We can prove ambiguity by showing that for some generated string, there are two or more different parse trees can be generated

For shorthand let `tn` = θ_n, and `e` = $expr$

First tree:

```
  e
  e    tn e
e tn e tn e
id tn id tn id
```

Second tree:

```
 e
e tn   e
e tn e tn e
id tn id tn id
```

Part b

Following the same shorthand:

```
 Expr -> term1 t1 Expr | term1
term1 -> term2 t2 term1 | term2
term2 -> term3 t3 term2 | term3
term3 -> term4 t4 term3 | term4
...
term_n -> term_n+1 t_n+1 term_n | term_n+1
term_n+1 -> term_f* | term_f
term_f -> id(Expr)
```

Problem 4

Part a

`FOLLOW(Es)` = `{)}` `FOLLOW(E)` = `{$$, atom, ', ), (}` `PREDICT(Es -> e)` = `{)}`

Part b

In order of parse tree depth, and with one substitution at a time:

```
 P
E $$
(E Es) $$
(atom Es) $$
(atom E Es) $$
(atom 'E Es) $$
(atom '(E Es) Es) $$
(atom '(E E Es) Es) $$
(atom '(E E E Es) Es) $$
(atom '(a E E Es) Es) $$
(atom '(a b E Es) Es) $$
(atom '(a b c Es) Es) $$
(atom '(a b c) Es) $$
(atom '(a b c)) $$
(cdr '(a b c)) $$
```

Part c

The active routines are:

```
 - P -> E $$
- E -> (EEs)
- Es -> EEs
- E -> ' E
```

Problem 5

Language	Ambiguity	LL(1)
1	No	No
2	No	Yes
3	Yes	No
4	Yes	No
5	No	Yes