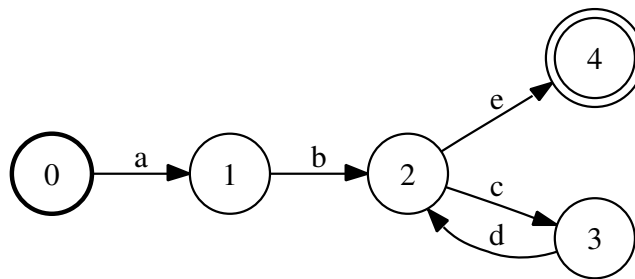# CMPT 379 - Summer 2016 - Midterm

(1) **(6pts) Lexical Analysis:**

   a. Consider the following DFA:

     Provide a regular expression for the regular language generated by this DFA.

   b. You are given the following ordered list of token definitions:

     TOKEN_A $(ab)^*a$

     TOKEN_B $(ab)^*a(ca)^*$

     TOKEN_C $bab(bab)^*$

     TOKEN_D $a^*ba(ba)^*$

     Provide the tokenized output for the input string *abacababab* using the greedy longest match lexical analysis method.

   c. The following CFG generates a regular language. Provide a regular expression that generates the same language as this CFG.

$$
\begin{aligned}
S &\rightarrow AB \\
A &\rightarrow c \mid \epsilon \\
B &\rightarrow cbB \mid ca
\end{aligned}
$$

(2) **(7pts)** Consider the following context-free grammar G.

$$
\begin{aligned}
S' &\rightarrow S & (1) \\
S &\rightarrow aSb & (2) \\
S &\rightarrow \epsilon & (3)
\end{aligned}
$$

   a. Provide FIRST(S).

   b. Provide the LR(0) itemset automaton for G.

   c. Is G LR(0)? If G is not LR(0), provide the shift/reduce or reduce/conflicts to justify your answer.

   d. Is G SLR(1)? Justify your answer using FOLLOW(S).

The missing non-terminals are **D1**, **L**, and **G**.

- `D1` recognizes a single non-zero leading digit and evaluates to that digit's value.
- `L` recognizes a units digit less than 5 (round down → 0).
- `G` recognizes a units digit greater than or equal to 5 (round up → 1).

```
D1 : "1" { $$ = 1; }
   | "2" { $$ = 2; }
   | "3" { $$ = 3; }
   | "4" { $$ = 4; }
   | "5" { $$ = 5; }
   | "6" { $$ = 6; }
   | "7" { $$ = 7; }
   | "8" { $$ = 8; }
   | "9" { $$ = 9; }
   ;

L : "0" { $$ = 0; }
  | "1" { $$ = 0; }
  | "2" { $$ = 0; }
  | "3" { $$ = 0; }
  | "4" { $$ = 0; }
  ;

G : "5" { $$ = 1; }
  | "6" { $$ = 1; }
  | "7" { $$ = 1; }
  | "8" { $$ = 1; }
  | "9" { $$ = 1; }
  ;
```

Explanation of test cases

- A single digit `0` uses `E : "0"` → 0.
- A single non-zero digit uses `E : D1` → its value (e.g. `5`→5, `9`→9).
- `02` fails: `E : "0"` allows no following digit → syntax error.
- A two-digit number `D1 D2`: `D1` is the tens digit, `D2` is 0 if units < 5 (via `L`) or 1 if units ≥ 5 (via `G`), giving rounding to the nearest ten:
  - `10` → 1·10 = 10
  - `22` → 2·10 = 20
  - `78` → 7·10 + 10 = 80
  - `35` → 3·10 + 10 = 40
  - `96` → 9·10 + 10 = 100
- `100` has three digits → no matching rule → syntax error.