Jason Natale

CSC 173 C weeks 3+4 calculator

Context Free Grammar: (Let € be epsilon, lowercase are terminals, uppercase non-terminals)

note My grammar is not LL(1), as I talked to Joyce and was informed it did not need to be

LIST => A ende LISTPRIME

A => B A'

B => C B'

C => lp pos rp C' D | lp neg rp C' D | D

D => lp inc rp D' E | lp dec rp D' E | E

E => lp A rp E' | val E'

LISTPRIME => eoft | A ende LISTPRIME

A' => € | add B A' | subt B A'

B' => € | mult C B' | div C B' | mod C B'

C' => € | lp pos rp C' | lp neg rp C'

D' => € | lp inc rp D' | lp dec rp D'

E' => € | lp inc rp E' | lp dec rp E'

EPS(LIST) = false EPS(A) = false EPS(B) = false EPS(C) = false

EPS(D) = false EPS(E) = false EPS(LISTPRIME) = false EPS(A') = true

EPS(B') = true EPS(C') = true EPS(D') = true EPS(E') = true

First(E') = {lp} First(D') = {lp} First(C') = {lp} First(B') = {mult,div,mod}

First(A') = {add,subt} First(LISTPRIME) = {eoft, lp, val} First(LIST) = {lp, val}

First(A) = {lp, val} First(E) = {lp, val} First(D) = {lp, val} First(C) = {lp, val}

First(B) = {lp, val}

Follow(LIST) = emptyset Follow(LISTPRIME) = emptyset Follow(A) = {ende}

Follow(A') = {ende} Follow(B) = {add, subt, ende} Follow(B') = {add,subt,ende}

Follow(C) = {mult,div,mod, add, subt, ende} Follow(C') = {lp, val}

Follow(D) = {mult, div, mod, add, subt, ende} Follow(D') = {lp, val}

Follow(E) = {mult, div, mod, add, subt, ende}

Predict(LISTPRIME => eoft) = {eoft}

Predict(LISTPRIME => A ende LISTPRIME) = {lp, val}

Predict(A => B A') = {lp, val}

Predict(A' => add B A') = {add}

Predict(A' => subt B A') = {subt}

Predict(B => C B') = {lp, val}

Predict(B' => mult C B') = {mult}

Predict(B' => div C B') = {div}

Predict(B' => mod C B') = {mod}

Predict(C => lp pos rp C' D) = {lp}

Predict(C => lp neg rp C' D) = {lp}

Predict(C => D) = {lp, val}

Predict(C' => lp pos rp C') = {lp}

Predict(C' => lp neg rp C') = {lp}

Predict(D => lp inc rp D' E) = {lp}

Predict(D => lp dec rp D' E) = {lp}

Predict(D => E) = {lp, val}

Predict(D' => lp inc rp D') = {lp}

Predict(D' => lp dec rp D') = {lp}

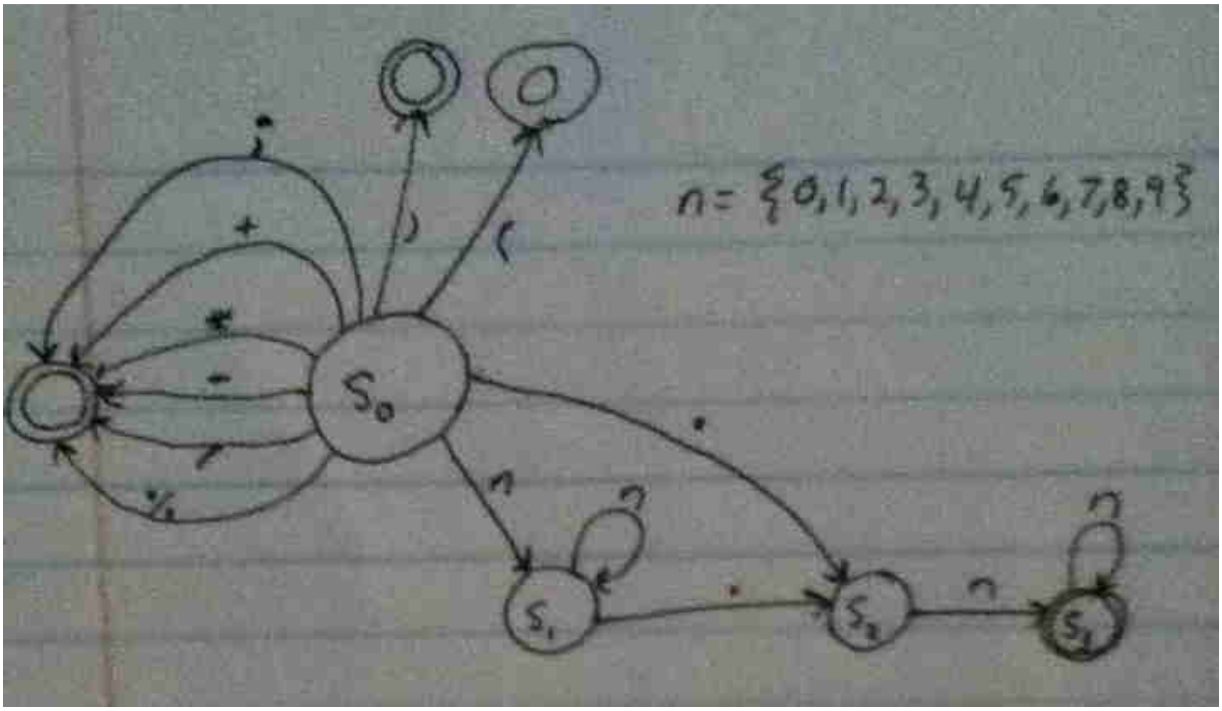
Predict(E => lp A rp E') = {lp}

Predict(E => val E') = {val}

Predict(E' => lp inc rp E') = {lp}

Predict(E' => lp dec rp E') = {lp}

<u>DFA</u>



note The double circle represents state "fin" in my scanner.c and scanner.h files*