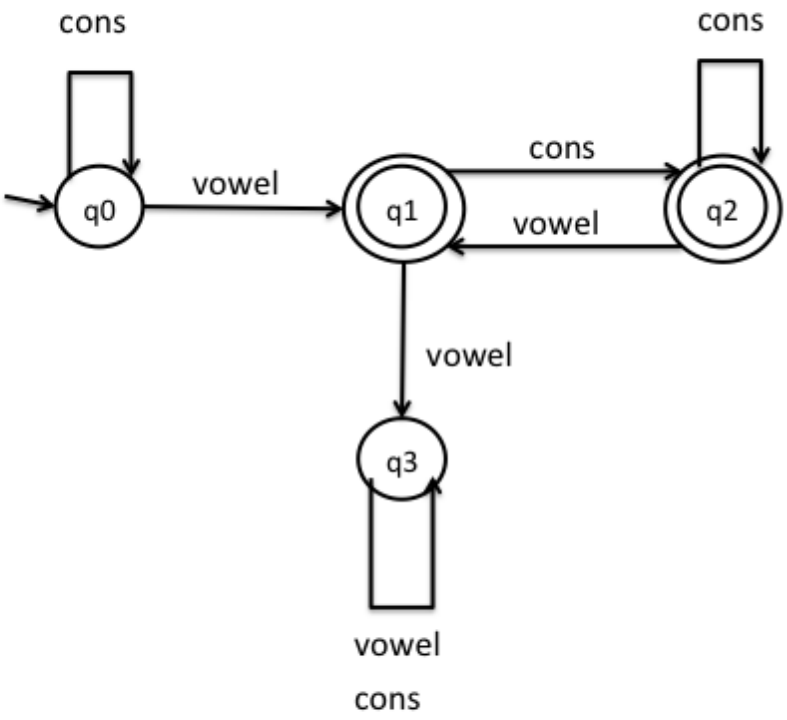
Midterm Exam Solution

CS 536, Spring 2013

Question 1 (12 points)

Part (a) (6 points)

There are many possible ways to do this; here are two:



Part (b) (6 points)

There are many possible ways to do this; here are two:

- cons* vowel (cons+ vowel)* cons*
- cons* vowel (cons vowel | cons+)*

Question 2 (10 points)

```
stmt ::= FOR LPAREN forCond RPAREN LCURLY varDeclList stmtList RCURLY

forCond ::= ID ASSIGN exp COLON exp optFor

optFor ::= COLON exp
        |  /* epsilon */
```

Question 3 (27 points)

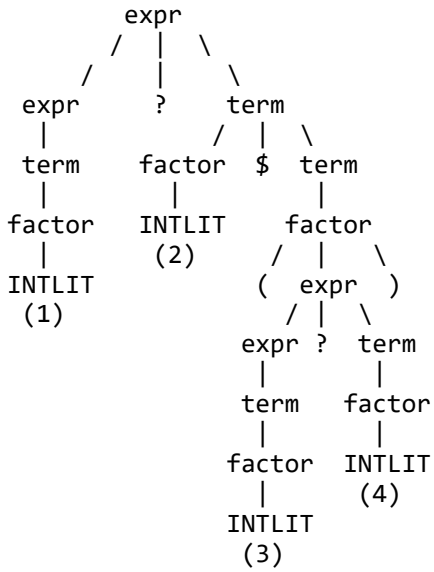
Part (a) (15 points)

```
exp → exp ? term  |  term

term → factor $ term  |  factor

factor → INTLIT  |  ( exp )
```

Part (b) (4 points)



Part (c) (4 points)

```
exp → term exp'

exp' → ? term exp'  |  ε

term → factor $ term  |  factor

factor → INTLIT  |  ( exp )
```

Part (d) (4 points)

```
exp → term exp'

exp' → ? term exp'  |  ε

term → factor term'

term' → $ term  |  ε

factor → INTLIT  |  ( exp )
```

Question 4 (15 points)

Grammar rule	Translation rule
program → varDeclList stmtList	`program.trans = stmtList.trans`
varDeclList → varDeclList varDecl	
varDeclList → varDecl	
varDecl → type ID ;	

type → INT	
type → BOOL	
stmtList → stmtList stmt	$stmtList_1$.trans = union($stmtList_2$.trans, stmt.trans)
stmtList → stmt	stmtList.trans = stmt.trans
stmt → ID = exp ;	stmt.trans = exp.trans
stmt → IF (exp) { varDeclList stmtList }	stmt.trans = union(exp.trans, stmtList.trans)
exp → ID	exp.trans = new List(ID.val)
exp → INTLIT	exp.trans = new List()
exp → exp + exp	exp_1.trans = union(exp_2.trans, exp_3.trans)
exp → exp * exp	exp_1.trans = union(exp_2.trans, exp_3.trans)
exp → exp == exp	exp_1.trans = union(exp_2.trans, exp_3.trans)

Question 5 (36 points)

Part (a) (24 points)

Non-terminal *X*	FIRST(*X*)	FOLLOW(*X*)
block	LCURLY	EOF, ELSE, ID, IF, RETURN, RCURLY
declList	TYPE, ε	ID, IF, RETURN, RCURLY
stmtList	ID, IF, RETURN, ε	RCURLY
decl	TYPE	TYPE, ID, IF, RETURN, RCURLY
stmt	ID, IF, RETURN	ID, IF, RETURN, RCURLY
exp	ID, INT	SEMI, RPAREN

Part (b) (12 points)

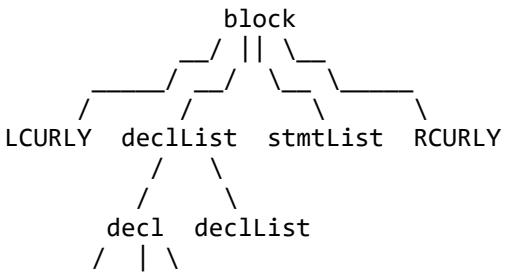
Snapshot 1:

current stack

```
stmtList
 RCURLY
  EOF
```

Snapshot 2:

partial parse tree



```
      /   |  \
   TYPE ID  SEMI
```

Snapshot 3:

input: `LCURLY RCURLY`

current stack

```
stmtList
 RCURLY
  EOF
```