Computer Science 330 Language Implementation Test Solution

Question 1 — 31 Marks

(a) Write JFlex regular expressions to match the following tokens. You may declare support regular expressions if you need them.

(i) A word composed of an upper case alphabetic character, followed by zero or more lower case alphabetic characters. For example, `Ronny` or `Sarah` or `B` but not `Tien-Wei` or `MacDonald` or `B2`.

```
[A-Z][a-z]*
```

(ii) A binary integer, with a prefix of `0b` or `0B`, followed by one or more binary digits. For example, `0b1010111` or `0B1010111` but not `0b123456789` or `0b`.

```
0[Bb][01]+
```

(iii) A floating point number, with an integer part, fractional part, and an exponent with an explicit sign for the exponent. For example, `123.456e+23` or `123.456e-23` but not `1.23e5` or `-1.23e-5` or `123e-5` or `3.14159`.

```
[0-9]+"."[0-9]+[Ee][\+\-][0-9]+
```

(iv) A decimal number with exactly three digits. The leading digit may be `0`. For example, `012` or `120` but not `1234` or `12`.

```
[0-9]{3}
```

(v) A decimal number, with the first digit nonzero, and at most three digits. For example, `123` or `12` but not `012` or `1234`.

```
[1-9][0-9]{0,2}
```

(vi) A sequence of one or more single decimal digits, with "," characters between each digit. For example, `1,2,3` or `2` but not `1,,2` or `12,34` or `,4`.

```
[0-9](","[0-9])*
```

(vii) A nonzero decimal number, and commas grouping the digits into blocks of three, as is normally done when writing numbers in English. For example, `1,200,001` or `12,345,678` or `12` but not `012,333` or `1234`.

```
[1-9][0-9]{0,2}(","[0-9]{3})*
```

(3 marks each)

Print your login name _____

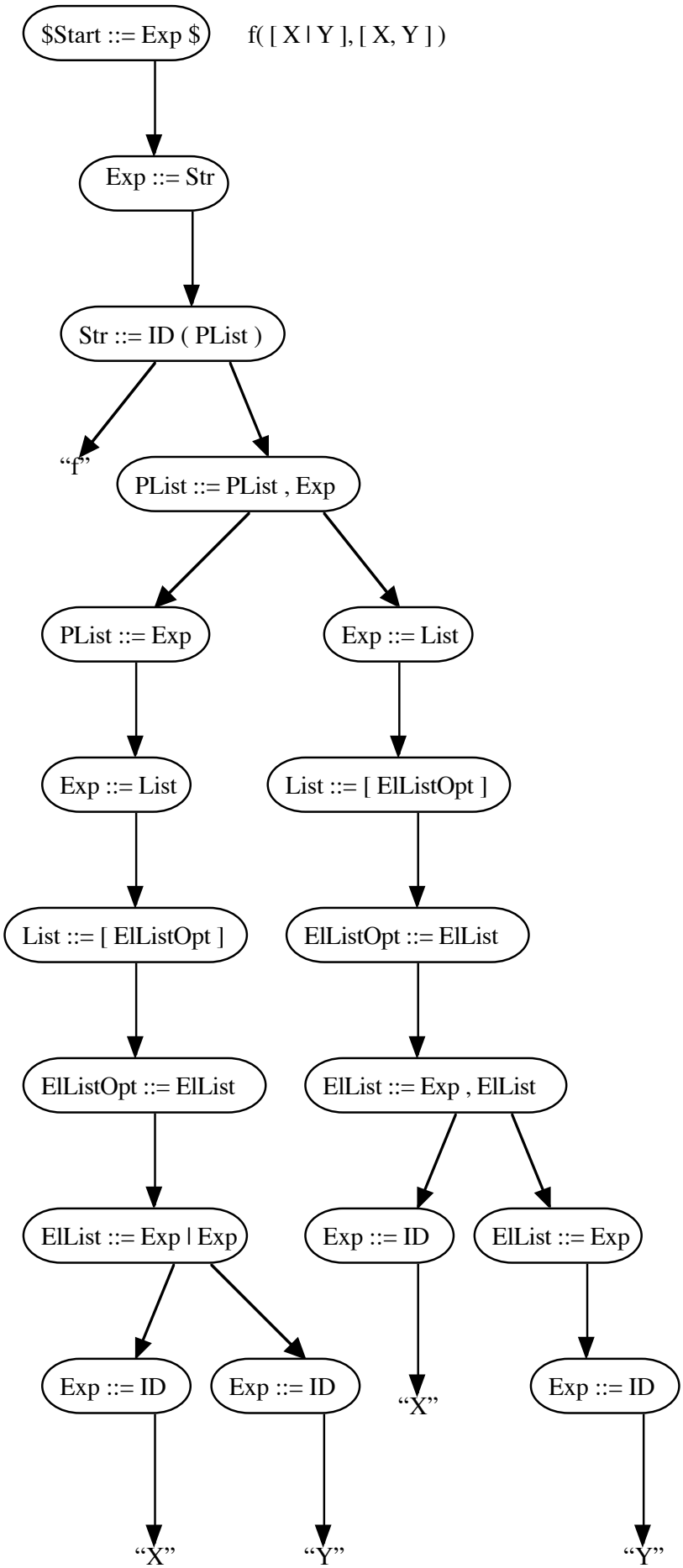
(b) Find and correct at least five different kinds of errors in the following fragment of JFlex code. ("..." just means omitted code). Assume spaces and line breaks are not syntactically important. Assume comments cannot be nested.

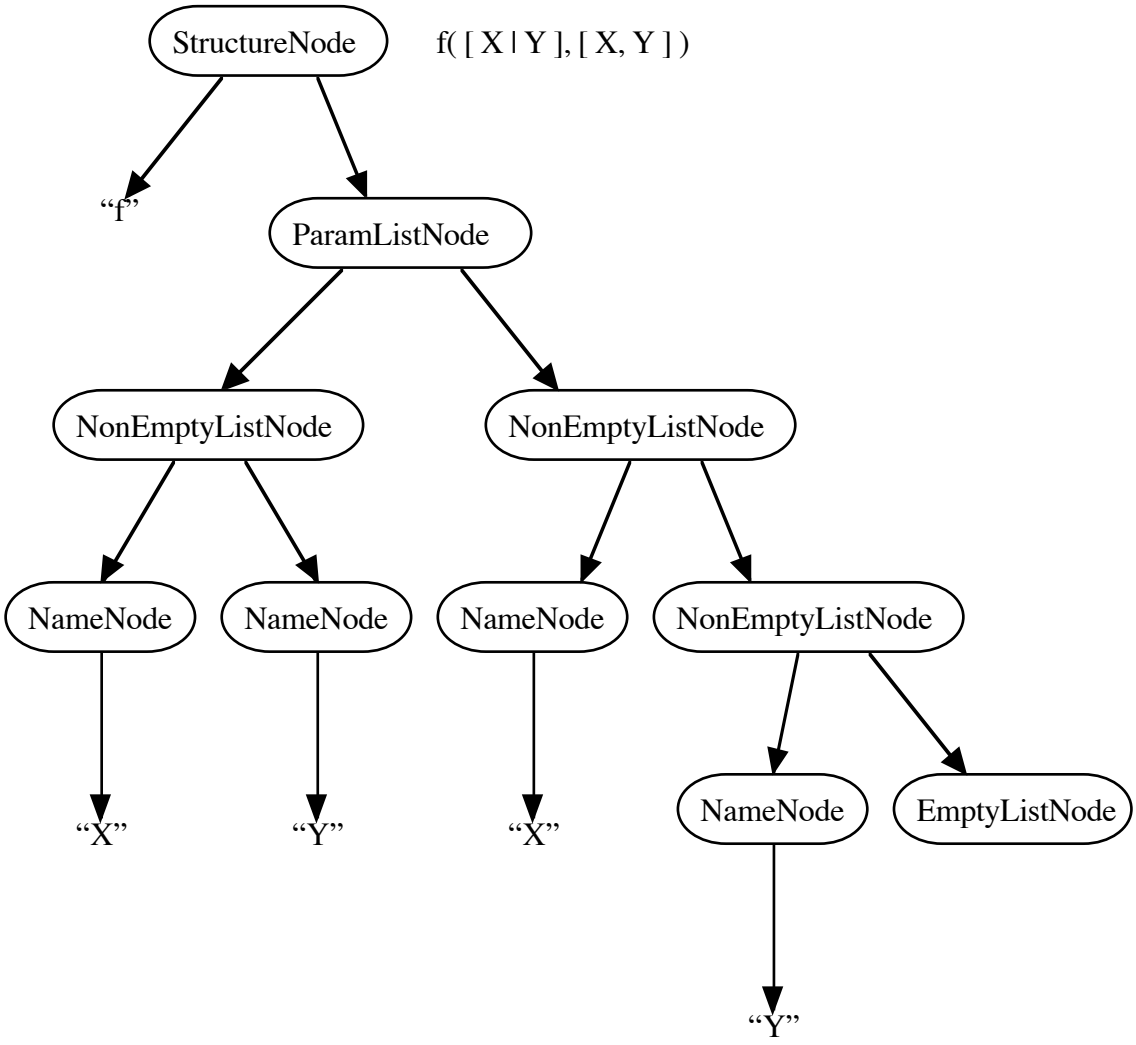
```
...
newline     =     \r|\n|\r\n
space       =     [\ |\t]<<<<<<<<< should not have "|"
ident       =     [A-Za-z][A-Za-z0-9]+ <<<<<<<<< should be "*"
%state NORMAL, COMMENT
%init{
      yybegin( COMMENT ); <<<<<<< NORMAL, not COMMENT
%init}

%%
<NORMAL> {
      {ident}     { return token( sym.IDENT ); } <<<<<< move below keywords
      if          { return token( sym.IF ); }
      else        { return token( sym.ELSE ); }
      ...
      /*          { yybegin( COMMENT ); } <<<<<< quote "/*"
      .           { return token( sym.error ); } <<<<<< move after {newline}
      {space}     { }
      {newline}   { linecount++; }
      }
<COMMENT> {
      */          { yyend( COMMENT ); } <<<<<< quote "*/", yybegin( NORMAL );
      {newline}   { linecount++; }
      .           { return token( sym.error ); } <<<<<< should have no action
      }
<<EOF>>           { return token( sym.EOF ); }
```

(10 marks)

Question 2 — 49 marks

Stack									Input	Action	
$0									ID f		Shift ID 4
$0	ID 4								(		Shift (14
$0	ID 4	(14							[		Shift [5
$0	ID 4	(14	[5						ID X		Shift ID 4
$0	ID 4	(14	[5	ID 4					|	Reduce E ::= ID	Shift E 6
$0	ID 4	(14	[5	E 6							Shift | 11
$0	ID 4	(14	[5	E 6	| 11				ID Y		Shift ID 4
$0	ID 4	(14	[5	E 6	| 11	ID 4			]	Reduce E ::= ID	Shift E 12
$0	ID 4	(14	[5	E 6	| 11	E 12				Reduce EL ::= E | E	Shift EL 7
$0	ID 4	(14	[5	EL 7						Reduce ELO ::= EL	Shift ELO 8
$0	ID 4	(14	[5	ELO 8							Shift] 9
$0	ID 4	(14	[5	ELO 8	] 9				,	Reduce L ::= [ELO]	Shift L 2
$0	ID 4	(14	L 2							Reduce E ::= L	Shift E 16
$0	ID 4	(14	E 16							Reduce PL ::= E	Shift PL 15
$0	ID 4	(14	PL 15								Shift , 17
$0	ID 4	(14	PL 15	, 17					[		Shift [5
$0	ID 4	(14	PL 15	, 17	[5				ID X		Shift ID 4
$0	ID 4	(14	PL 15	, 17	[5	ID 4			,	Reduce E ::= ID	Shift E 6
$0	ID 4	(14	PL 15	, 17	[5	E 6					Shift , 10
$0	ID 4	(14	PL 15	, 17	[5	E 6	, 10		ID Y		Shift ID 4
$0	ID 4	(14	PL 15	, 17	[5	E 6	, 10	ID 4	]	Reduce E ::= ID	Shift E 6
$0	ID 4	(14	PL 15	, 17	[5	E 6	, 10	E 6		Reduce EL ::= E	Shift EL 13
$0	ID 4	(14	PL 15	, 17	[5	E 6	, 10	EL 13		Reduce EL ::= E , EL	Shift EL 7
$0	ID 4	(14	PL 15	, 17	[5	EL 7				Reduce ELO ::= EL	Shift ELO 8
$0	ID 4	(14	PL 15	, 17	[5	ELO 8					Shift] 9
$0	ID 4	(14	PL 15	, 17	[5	ELO 8	] 9		)	Reduce L ::= [ELO]	Shift L 2
$0	ID 4	(14	PL 15	, 17	L 2					Reduce E ::= L	Shift E 19
$0	ID 4	(14	PL 15	, 17	E 19					Reduce PL ::= PL , E	Shift PL 15
$0	ID 4	(14	PL 15								Shift) 18
$0	ID 4	(14	PL 15	) 18					$	Reduce S ::= ID (PL)	Shift S 1
$0	S 1									Reduce E ::= S	Shift E 3
$0	E 3										Shift $ 20
$0	E 3	$ 20							$	Reduce $St ::= E $	Shift $St -1
$0	$St -1									Accept	


$Start ::= Exp $
f([X | Y], [X, Y])
Exp ::= Str
Str ::= ID (PList)
“f”
PList ::= PList , Exp
PList ::= Exp
Exp ::= List
Exp ::= List
List ::= [ElListOpt]
List ::= [ElListOpt]
ElListOpt ::= ElList
ElListOpt ::= ElList
ElList ::= Exp , ElList
ElList ::= Exp | Exp
Exp ::= ID
ElList ::= Exp
Exp ::= ID
Exp ::= ID
“X”
Exp ::= ID
“X”
“Y”
“Y”


StructureNode
f([X | Y], [X, Y])
“f”
ParamListNode
NonEmptyListNode
NonEmptyListNode
NameNode
NameNode
NameNode
NonEmptyListNode
“X”
“Y”
“X”
NameNode
EmptyListNode
“Y”

Question 3 — 8 marks

```
package node.stmtNode;

import env.*;
import code.*;

import node.exprNode.*;

public class IfThenElseStmtNode extends StmtNode {

      private ExprNode cond;
      private StmtNode thenPart;
      private StmtNode elsePart;

      public IfThenElseStmtNode(
            ExprNode cond,
            StmtNode thenPart,
            StmtNode elsePart ) {
            this.cond = cond;
            this.thenPart = thenPart;
            this.elsePart = elsePart;
            }

      public String toString() {
            return "if " + cond + " then%+%n"
                  + thenPart + "%-%nelse%+%n" + elsePart + "%-";
            }

      public void genDeclCode( Env env ) {
            thenPart.genDeclCode( env );
            elsePart.genDeclCode( env );
            }

      public void genCode() {
            Code.enter();
            Code.labelDefn( "if" );
            cond.evalCode( 0 );
            Code.instrn( "blbc", "$t0", "else" );
            Code.labelDefn( "then" );
            thenPart.genCode();
            Code.instrn( "br", "end" );
            Code.labelDefn( "else" );
            elsePart.genCode();
            Code.labelDefn( "end" );
            Code.exit();
            }

      }
```

Question 4 — 20 marks

```
SwitchStmt::=
            SWITCH LEFT Expr RIGHT LEFTBRACE
            CaseStmtList DefaultStmtOpt RIGHTBRACE;

CaseStmtList::=
            CaseStmt
      |
            CaseStmtList CaseStmt
      ;

CaseStmt::=
            CASE CaseExprList COLON Stmt
      ;

DefaultStmtOpt::=
            /* Empty */
      |
            DEFAULT Stmt
      ;

CaseExprList::=
            CaseExpr
      |
            CaseExprList COMMA CaseExpr
      ;

CaseExpr::=
            SignedInt
      |
            SignedInt DOTDOT SignedInt
      ;

SignedInt::=
            PLUS INTCONST
      |
            MINUS INTCONST
      |
            INTCONST
      ;
```