Lecture 18 Worksheet

Question 1

Try to break the following Jack statements into parse trees.

(we haven't covered the syntax yet so we want you take some educated guesses. You'll be able to check whether you're right later.)

```
if (x < 0)
{
        let x = 0 + y
}
```

```
if ((x > 3) & (y < 4))
{
        let y = 3
} else
{
        let x = 4
}
```

PTO for Question 2

Lecture 18 Worksheet

Question 2

Now you have the following syntax definition. Check your answers to question 1 above and, if they are wrong correct them.

Statements:	
statements:	statement*
statement:	letStatement | ifStatement | whileStatement | doStatement | returnStatement
letStatement:	'**let**' varName ('**[**' expression '**]**')? '=' expression '**;**'
ifStatement:	'**if**' '**(**' expression '**)**' '**{**' statements '**}**' ('**else**' '**{**' statements '**}**')?
whileStatement:	'**while**' '**(**' expression '**)**' '**{**' statements '**}**'
doStatement:	'**do**' subroutineCall '**;**'
ReturnStatement	'**return**' expression? '**;**'
Expressions:	
expression:	term (op term)*
term:	integerConstant | stringConstant | keywordConstant | varName | varName '**[**' expression '**]**' | subroutineCall | '**(**' expression '**)**' | unaryOp term
subroutineCall:	subroutineName '**(**' expressionList '**)**' | (className | varName) '**.**' subroutineName '**(**' expressionList '**)**'
expressionList:	(expression (',' expression)*)?
op:	'+' | '-' | '*' | '/' | '&' | '|' | '<' | '>' | '='
unaryOp:	'-' | '~'
KeywordConstant:	'**true**' | '**false**' | '**null**' | '**this**'

Question 3

Construct an XML representation of the parse tree for the following Jack ifStatement:

```
if (x < 0)
{
        let x = 0 + y
}
```