Defining Language TWO

- Extend Language ONE with:
  - Variables
  - An ML-style `let` expression for defining them
TWO: Syntax

A subset of ML expressions
This grammar is unambiguous
A sample Language TWO expression:

```
let val y = 3 in y*y end
```

What does the parse tree for the above expression look like?

TWO:

<exp>* ::= <exp> + <mulexp> | <mulexp>
<mulexp> ::= <mulexp> * <rootexp> | <rootexp>
<rootexp> ::= let val <variable> = <exp> in <exp> end
            | (<exp>) | <variable> | <constant>
Two: Abstract Syntax

Additional abstract syntax nodes for language TWO:

1. \texttt{var(X)} dereferences a variable X
2. \texttt{let(X,E1,E2)} binds the variable X to expression E1 in the context of expression E2.

Example: the TWO program

\begin{verbatim}
let val y = 3 in y*y end
\end{verbatim}

will result in the AST

\begin{verbatim}
let(y, const(3), times(var(y), var(y)))
\end{verbatim}
From Parse Tree to Prolog AST

- Consider: 2 * let x = 5 in 1+x end
  - Parse tree?
  - AST?
  - Prolog AST?
In order to provide semantics we need to remember the values assigned to variables -- binding environments, contexts.

In our case, for the Prolog based semantics, we let the terms bind(X,K) represent the binding of variable X to value K. A context is simply a list of these binding terms:

\[
\text{[bind(y,3),bind(q,20),bind(z,5)]}
\]

Given this binding structure, we can write a predicate, lookup/3, that returns a variable binding for a particular Var

\[
\text{lookup(Var, [bind(Var,Value) | _ ], Value).}
\text{lookup(Var, [ _ |Rest], Value) :- lookup(Var,Rest,Value).}
\]

Finds the most recent binding of variable \texttt{Var} if there is one.
two: Prolog Interpreter

val2(plus(X,Y),C,Value) :-
    val2(X,C,XValue),
    val2(Y,C,YValue),
    Value is XValue + YValue.

val2(times(X,Y),C,Value) :-
    val2(X,C,XValue),
    val2(Y,C,YValue),
    Value is XValue * YValue.

val2(const(X),_,X).

val2(var(X),C,Value) :-
    lookup(X,C,Value).

val2(let(X,Exp1,Exp2),C,Value) :-
    val2(Exp1,C,XValue),
    val2(Exp2,[bind(X,XValue)|C],Value).

val2 / 3 - interpretation predicate, first argument: AST; second argument: context; third argument: semantic value.
Examples

let val y = 3 in y*y end

?- val2(let(y,const(3),times(var(y),var(y))),[ ],X).
X = 9
Yes

let val y = 3 in
  let val x = y*y in
    x*x
  end
end

let val y = 1 in
  let val y = 2 in
    y
  end
end
Exercises

- Use the semantics of TWO to show the following:
  - Assume that the context $C = [\text{bind}(y,3)]$ then the semantic value of ‘$2\times y$’ is 6
  - The semantic value of ‘$2 \times \text{let} \ x = 3 \ \text{in} \ x \times x \ \text{end}$’ is 18
  - The semantic value of ‘$\text{let} \ x = 1 \ \text{in} \ \text{let} \ y = x + 1 \ \text{in} \ y \ \text{end} \ \text{end}$’ is 2
Exercises

Use the semantics to compute the meaning of the following expressions in TWO (use the rules given in the notes, the book has many typos):

1) let val y = 3 in 2*y end

2) let val y = 1 in
   let val y = 2 in
   y
   end
   end

Note: first construct an abstract syntax tree, then give the representation in Prolog notation, and then show the computation in our semantics.