CSE 12 – Basic Data Structures

Prof. Christine Alvarado

[Slides borrowed/adapted from slides by Cynthia Lee, Rakesh Varna, & Roshni Chandrashekhar]
Announcements

1. HW8
   1. Released Wednesday (at some point… maybe late)
   2. Due Tuesday week 10 (2 week assignment)
   3. Worth 400 points.
   4. It will be LONG. If you wait to start you will not succeed and you will significantly drop your grade in the class!

2. HW8 is PARTNER ENcouraged
   1. You must sign up via the Google form no later than this FRIDAY at 5pm
   2. Once you sign up you are “married” to your partner. Divorces must be done in person by me or Dr. P
Announcements

1. Don’t forget the post-video quiz and the quiz for Friday
Your turn!

"2 + 5 * 3"

\[
\begin{align*}
&S \Rightarrow S + P \\
&S + P \\
&S \Rightarrow S + P * M \\
&S + P * 3 \\
&S \Rightarrow S + \text{const} * 3 \\
&S + \text{const} * 3 \\
&S \Rightarrow P + S * 3 \\
&P + S * 3 \\
&P \Rightarrow M + 5 * 3 \\
&M + 5 * 3 \\
&M \Rightarrow \text{const} + 5 * 3 \\
&\text{const} + 5 * 3 \Rightarrow 2 + 5 * 3
\end{align*}
\]
"2 + 5 * 3"

\[
\begin{align*}
\text{<S>} & := \text{<S>} + \text{<P>} \mid \text{<S>} - \text{<P>} \mid \text{<P>} \\
\text{<P>} & := \text{<P>} * \text{<M>} \mid \text{<P>} / \text{<M>} \mid \text{<M>} \\
\text{<M>} & := \text{<const>} \mid (\text{<S>}) \\
\text{<const>} & := 0 \mid 1 \mid 2 \mid 3 \mid 4
\end{align*}
\]

\[
\begin{align*}
\text{<S>} & \Rightarrow \text{<S>} + \text{<P>} \\
& \Rightarrow \text{<S>} + \text{<P>} * \text{<M>} \\
& \Rightarrow \text{<S>} + \text{<P>} * \text{<const>} \\
& \Rightarrow \text{<S>} + \text{<P>} * 3 \\
& \Rightarrow \text{<S>} + \text{<M>} * 3 \\
& \Rightarrow \text{<S>} + \text{<const>} * 3 \\
& \Rightarrow \text{<S>} + 5 * 3 \\
& \text{...} \\
& \Rightarrow 2 + 5 * 3
\end{align*}
\]
Your turn!

Draw the parse tree for "(2 + 5) * 3"
Your turn!

Draw the parse tree for "(2 + 5) * 3"
Abstract Syntax Trees

```
(S)  
|    
P   +
|    
M   (S)
|    (S) + (P)
|    |    |    |
M   |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    |    |
|    |    |    |    |
const |    |    | 5
```
OK, so how does the computer create the tree?

\[
{\texttt{S}} := {\texttt{P}} + {\texttt{S}} \mid {\texttt{P}} - {\texttt{S}} \mid {\texttt{P}}
\]

\[
{\texttt{P}} := {\texttt{M}} \times {\texttt{P}} \mid {\texttt{M}} / {\texttt{P}} \mid {\texttt{M}}
\]

\[
{\texttt{M}} := \langle \text{const} \rangle \mid ( {\texttt{S}} )
\]

\[
\langle \text{const} \rangle := 0 \mid 1 \mid 2 \mid 3 \mid 4
\]

A slight modification to the grammar (no longer follows proper associativity)

\[
(5 - 3) - 2 \\
5 - (3 - 2)
\]
What is the parse tree for “( 2 + 5 )”?

Recursive Decent Parsing

\[<S> ::= <P> + <S> | <P> - <S> | <P>\]
\[<P> ::= <M> * <P> | <M> / <P> | <M>\]
\[<M> ::= \langle const \rangle | (<S>)\]
\[<\text{const}> ::= 0 | 1 | 2 | 3 | 4\]

tokens: ["(" , "2" , "+" , "5" , ")"]
S: "I start with a P, so ask P to parse"

tokens: ["(" , "2" , "+" , "5" , ")"]
P: "I start with a M, so ask M to parse"

tokens: ["(" , "2" , "+" , "5" , ")"]
M: "I could start with <\text{const}> so let’s try it"

tokens: ["(" , "2" , "+" , "5" , ")"]
const: “I don’t match “)"! Failure!

tokens: ["(" , "2" , "+" , "5" , ")"]
M: “OK, I could also start with “(“.

tokens: ["(" , "2" , "+" , "5" , ")"]
M: “Ah ha! I see a “)”. Now I’ll ask S to parse”

tokens: ["2" , "+" , "5" , ")"]
S: "I start with a P, so ask P to parse"

tokens: ["2" , "+" , "5" , ")"]
P: "I start with a M, so ask M to parse"

tokens: ["2" , "+" , "5" , ")"]
M: "I could start with <\text{const}> so let’s try it"

tokens: ["2" , "+" , "5" , ")"]
const: “Oh good, a 2”

tokens: ["+" , "5" , ")"]
S: “Now that I have seen a P, I could see a +“

tokens: ["5" , ")"]
S: “Oh look. There’s one! I’m a + S”

S: Now I need another S so ask S to parse…
"The only way to be successful with Recursive Descent is to have complete faith in recursion"
-- A HMC CS 60 student

Recursive Descent will test your faith…
Behold! miniMath

To the code...
inumbrage
atonement only in graven speck
soft hearts seek kind images in a memento
gather scarlet note
stendanger slay her relapses

Atonement I, only in graven specks of the arts, seek.
In dim ages, I name men to God, I gather scarlet notes.
Ten dangers lay. She relapses.

At one mention, lying ravens peck soft hearts.
See kind images in a memento.
Go dig at her scar. Let no test end.
Anger slays her, relapses.

- Mike Maguire