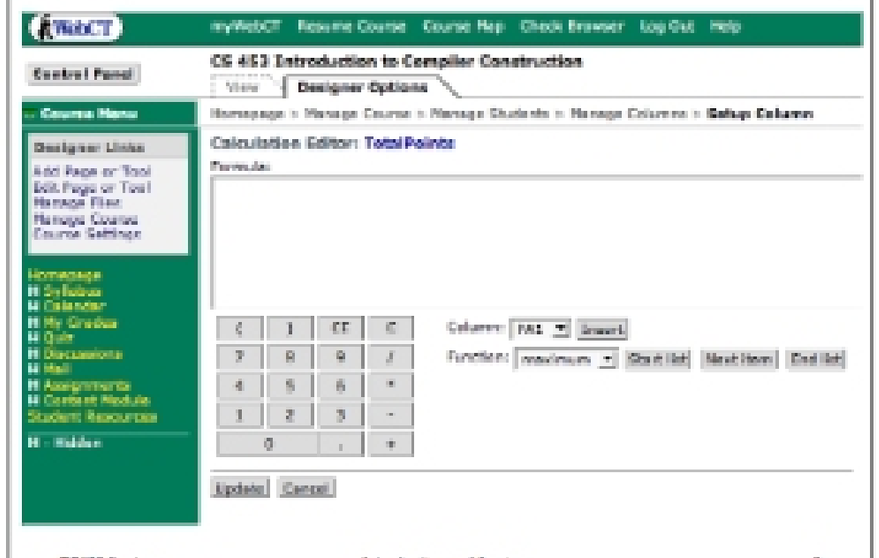
Plan for Today

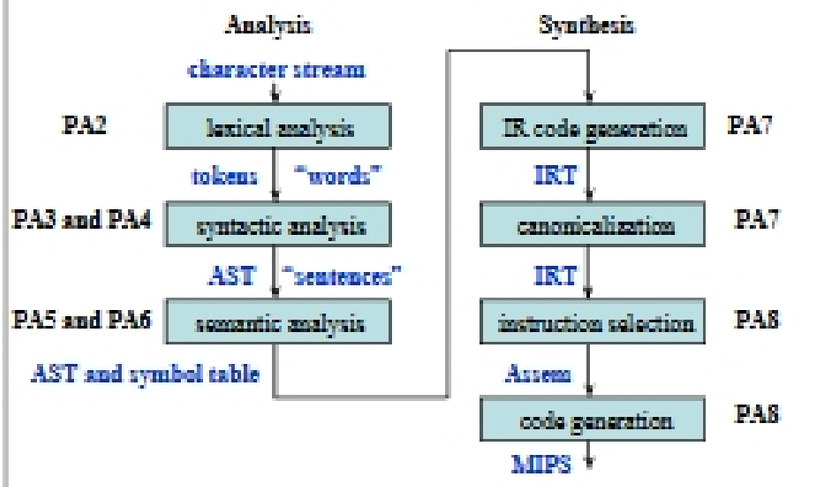
Motivation
- Why study compilers?

Programming Assignment Overview
- The compiler we will be building

Lexing/Scanning
- Representing tokens with regular expressions
- Using DFAs to recognize a string of tokens



Structure of the MiniJava Compiler



Specifying Tokens with SableCC

Theory meets practice:
- Regular expressions, formal languages, grammars, parsing…

SableCC example input file:

```
Package minijava;

Helpers
  all = [0..0xFFFF];
  cr  = 13;

  digit  = ['0'..'9'];
  letter = ['a'..'z'] | ['A'..'Z'];
  underscore  = '_';

  not_star       = [all - '*'];
  not_star_slash = [not_star - '/'];

  c_comment = '/*' not_star* ('*'
(not_star_slash not_star*)?)* '*/';
```

```
Tokens
  plus  = '+';

  lf    = 'LF';

  id = letter (letter | digit | underscore)*;

  blank   = (' ' | eol | tab)+;
  comment = c_comment | line_comment;

Ignored Tokens
  blank,
  comment;
```

Example DFA for Recognizing a String of Tokens

```
digit  = ['0'..'9'];
letter = ['a'..'z'] | ['A'..'Z'];
not_f = [.letter - ['f'+'F']];
not_o = [.letter - ['o'+'O']];
not_r = [.letter - ['r'+'R']];
Tokens
  for    = 'for';
  id = letter (letter | digit | underscore)*;
```