6.035 Info Session 2: Semantic Checker

Spring 2016

Team Repositories Created

Java	Scala	Haskell
team1		
6035-team		
ddsz	g9north	
waffles	OG Kushdale	
hats	Human suffering	disfunctionalprogramming
hard-disk_and_hard-diks	ALittleByte	Lambda Cabal
Carthage	BusyBeavers	The Unexpected Characters
BAM!	Team Bacon	
Gnu Kids on the Blockchain		
Espresso Yourself		

Timeline

Project	Points	Due
Scanner-Parser		
Semantic Checker		March 1 (ungraded feedback)
Code Generator	25	March 18
Data-flow Analysis		
Optimizer	45	

Components

Parser: from project 1

Abstract syntax tree

High-level IR: simplifies subsequent analysis
design carefully

Symbol tables: hold meta-information

Semantic checks

Building AST in Parser

```
program: (s:stmt)* {#program = #([PROGRAM,"program"], #program);};

stmt: ID ASSIGN^ expr SEMICOLON!;

expr: INT (PLUS^ INT)*;
```

{}: Insert code (imaginary token)

^: Make node a root node

!: Exclude node from AST

*: Left recursion (left associativity)

ANTLR References

http://www.antlr2.org/doc/trees.html

http://www.antlr2.org/doc/sor.html

http://www.antlr2.org/javadoc/antlr/collections/AST.html

http://www.antlr2.org/doc/err.html