Object-Oriented Design
Project 8
Caleb Bitting, Erik Cohen, Matt Cerrato, Ian Ellmer

**Design Overview**

In this project, we were tasked with implementing the Parser for our Bantam Java Compiler. We have all the given methods filled out, yet some methods are more complete and correct than others. We started by addressing comments from section 7 making sure that our Scanner works properly for the Parser. Our coding method was not the best. We opted for a fill in all methods and test later approach which lead to broken messy code with bugs that are hard to locate. We conf

**Class Structure**

The class structure of our parser was what was given to us (a series of AST nodes and a main Parser class)

**Elegant Design**

To reduce the amount of "boiler plate" code we included a helper method that provides some readability to handling errors throughout the Parser file. These two methods are "registerAndThrow()" and "ensureTokenType()".

**Shortcomings**

Our Parser does not work for statements with multiple references for example, "string.get(2).hello();" would crash the compiler even though it should make syntactic sense. In testing our BantamExample.java file, we are unsure of how to get our program running. Due to scheduling conflicts not enough hours were spent on this project and it is incomplete as is.

Our code is not exactly where we would like it to be, but it currently parses our BantamExample.java.

**Division of Labor**

- Caleb worked on Parser.java and refactored for elegance. Caleb also reviewed the project 7 bug fixes.
- Ian wrote several methods in Parser.java
- Matt wrote several methods in Parser.java and debugging
- Erik worked on bug fixing project 7 and worked on methods/debugging Parser.java