Inheritance Hierarchy

- IO
  - LexArithArray
    - Parser
- Obj
  - Val
    - IntValue
    - FloatValue
  - AssignmentList
    - Assignment
    - MultipleAssingment
  - E
    - SingleTerm
    - AddE
    - SubE
  - Term
    - SinglePrimary
    - MulTerm
    - DivTerm
  - Primary
    - Id
    - Int
    - Floatp
    - Parenthesized
- CountObjects

About Data Structure and Sorting

The objective of this project was to implement the data structure that maintains the *Obj* and all its descendant classes and outputs number of times a *Obj* and its descendant classes was instantiated in descending order. For that, I implemented a class called *CountObjects* which has a method named *getSuperClass* which takes in class name as argument and recursively gets all its superclasses. Whenever new class name is seen, a new entry with value 1 is pushed into hashmap. However, if the class is already present in the hashmap, its value is increased by one.

Since classes were maintained in a hashmap, sorting was not easy to achieve. However, I implemented a function which takes in any HashMap and returns LinkedHashMap in sorted order. It does so by first breaking key and value of input hashmap into two separate ArrayLists. Then, using Collections class, the second ArrayList containing value is sorted in reverse order. Then for each sorted value, its key is found and pushed into LinkedHashMap. This LinkedHashMap is then iterated in the main to get each classes and their number of repetition in sorted order.