5.3 The Language: Grammar

<program> ::= {<class-decl>}* <expression>

```
a-program (class-decls body)
```

<class-decl> ::= **class** <identifier> **extends** <identifier>
{**field** <identifier>}* {<method-decl>}*

```
a-class-decl
  (clas-name supername field-ids method-decls)
```

<method-decl> ::= **method** <identifier> **(**{<identifier>*(,)**)**
<expression>

```
a-method-decl (method-name ids body)
```

The Language :Grammar

<expression> ::= **new** <identifier> **(**{<expression>*(,)**)**

```
new-object-exp (class-name rands)
```

<expression> ::= **send** <expression> <identifier> **(**{<expression>}*(,)**)**

```
method-app-exp (obj-exp method-name rands)
```

<expression> ::= **super** <identifier> **(**{<expression>}*(,)**)**

```
super-call-exp (method-name rands)
```

The Language : Interpreter

```
(define eval-program
  (lambda (pgm)
    (cases program pgm
      (a-program (c-decls exp)
        (elaborate-class-decls! c-decls)
        (eval-expression exp (empty-env))))))
```

- Why `!` in `elaborate-class-decls!` ??
- Note crucial ordering between last two lines:

c.f. Exercise 3.5:

```
(print-prim () (display (car args)) (newline) 1)
```