Juan Manuel Munoz Perez
juanp@comp.nus.edu.sg
A0134739X
Thursday 22th October 2015

National University of Singapore
CS4212 Assignment 2
AY 2015/2016 Semester 1
Due Date: 22th October 2015 (19:00 Hrs)

How to read the files

To build the executables, launch `sh build.sh`. All the tests will be executed. Two main directories contains the relevant tests (`annotatedtyped` and `ir3codegeneration`).

Part A

In order to produce an easy-to-read code, each component of the static checker has been implemented independently and sequentially. The method `type_mOOL_program` gives explicitly the flow followed.

Important assumption

One important assumption which has been made has been inspired from Python: *"A class A cannot be extended to class B defined after it within the code"*. The following example illustrates it:

Not allowed

```
class Main {...}
class A extends B {...}
class B {...}
```

Allowed

```
class Main {...}
class A {...}
class B extends A {...}
```

Overloading/Overriding

Overloading is first done within the class methods. Then overloading and overriding are checked within the parent's methods, then parent's parent's methods, and so on.

Casting

Upcasting is implicit whereas downcasting is always explicit. The following cases illustrates the assumptions:

Not allowed

```
class A {...}
class B extends A {
...
A a;
B b;
b = new A(); // should be b = (B) new A();
a = (A) new B(); // should be a = new B();
… }
```

Allowed

```
class A {...}
class B extends A {
...
A a;
B b;
b = (B) new A(); // downcasting
a = new B(); // upcasting
... }
```

Shadow policy

This is done within the type checking through the methods

- `get_inherited_environment` builds the environment inherited from super classes;
- `get_class_method_env` builds the environment a method's class using the inherited environment.

Type checking

A method type checking is done by checking that its return type is compatible with its statements return type. Roughly, the methodology followed is:

1. Check that `params` and `localvars` contains variables with well defined types;
2. Type check the statements:
    a. Create the method's scope;
    b. Type check method's statements;
    c. Check compatible types with method's return and stmts return

Part B

In order to generate the IR3 code for `IfStmt3` and `WhileStmt3`, I used the the indications given from slide 49 to 54 in lecture *Intermediate Code Generation* (Translating Control-Flow, Short-Circuit code).