Class closure and Instance Closure:

Class closure:

1) Super class

2) Methods

3) Class fields and values (optional)

4) Instance fields names

5) Constructors (optional)

Instance closure:

1) Runtime type (true type)

2) Instance field values (for all fields of all polymorphic types)

Methods:

When you call a method:

a.m(x, y)

Similar to the denotational semantics for C-style functions of interpreter part 3.

(Java -style OO function call)

0) We get the expression to the left of the dot, evaluate it in the current state + the current class we are executing this code in to get the object we are calling the function on

1) Lookup m in the ~~state~~ the closure of the runtime type of the object (from the object closure) and get its closure

2) Call the function from the closure that creates the state for executing the body of the method.

3) Place a layer on that state, for each formal parameter (from the closure) and each actual parameter (from the function call),

- evaluate the actual parameter in the current state + current type of the class we are in

- bind it to the formal parameter and place in the top layer of the function state

- bind the object closure of the value to the left of the dot to the name “this” and place that binding in the top layer of the function state.

3.5) Get the current type of the function from using a function stored in the function’s closure.

4) We call Mstate on the function body (from the closure) using the function state + the compile time type (current type) of the function and a new return continuation

public class A { public class B extends A {

int x; int x;

}

void setX(int x) {

this.x = x

}

}

B b = new B()

b.setX(10)

Hint:

When creating a class that extends a class,

When creating the lists of methods and filed names, just append these to the lists already in the super class.

Typically, you search for a method in the runtime type, if not found, go to the super class and look there. But if you append the class’s method list onto the list from the super type, then you only need to search one list.

Fields/Variables:

When you see a name, “x”, you need to first look in the state. If not found, look in the fields of the class for this method. (Look in both the instance fields and the static fields).