

Semantics

Assignment Semantics

- We know that the rhs of an assignment statement is an expression. To determine the meaning of an assignment statement, we need to know the meaning of the rhs expression.
- Now, we know how to determine the meaning of an expression. Then, let's figure out the way to determine the meaning of an assignment.
- We know that the meaning function of an expression generate a new value and the meaning function of an assignment generates a new state. (Remember: Assignment is a type of statement.)

$$M : Expression \times State \rightarrow Value$$

$$M : Statement \times State \rightarrow State$$

- This means the value generated by the rhs expression is used to update the value of the lhs variable of the assignment. In the new state, the lhs variable will be associated with the new value generated by the rhs expression. Expressed this mathematically,

$$M(Assignment a, State state) = state \bar{\cup} \{a.target, M(a.source, state)\}$$

Here, $\bar{\cup}$ means overriding union.

- If we have a set X which contains three pairs, $\{ \langle a, 1 \rangle, \langle b, 5 \rangle, \langle c, 1 \rangle \}$, and a set Y which contains two pairs, $\{ \langle b, 6 \rangle, \langle d, 0 \rangle \}$, the overriding union of X and Y is:

$$X = \{ \langle a, 1 \rangle, \langle b, 5 \rangle, \langle c, -1 \rangle \}$$

$$Y = \{ \langle b, 6 \rangle, \langle d, 0 \rangle \}$$

$$X \bar{\cup} Y = \{ \langle a, 1 \rangle, \langle b, 6 \rangle, \langle c, -1 \rangle, \langle d, 0 \rangle \}$$

The **overriding union** of X and Y, written $X \bar{\cup} Y$, is the result of **replacing** in X all pairs $\langle x, v \rangle$ whose first member matches a pair $\langle x, w \rangle$ from Y by $\langle x, w \rangle$ and then adding to X any remaining pairs in Y

- Example
 - If $X = \{ \langle a, 1 \rangle \}$ and $Y = \{ \langle b, 2 \rangle \}$, what is the result of overriding union of X and Y?
 $\{ \langle a, 1 \rangle, \langle b, 2 \rangle \}$
 - If $X = \{ \}$ and $Y = \{ \langle c, 3 \rangle \}$?
 $\{ \langle c, 3 \rangle \}$
 - If $X = \{ \langle d, 4 \rangle \}$ and $Y = \{ \langle d, 4 \rangle \}$?
 $\{ \langle d, 4 \rangle \}$
- We are going to expand the simple interpreter we built for the expression and let it be able to determine the meaning of assignment.
- Add: Assignment class, Meaning function for assignment, and let the state can be printed out pretty. Also, a main function to test them.

```

# Assignment has the attribute target and source
class Assignment:
    def __init__(self, target, source):
        self.target = target
        self.source = source

# Maintain a dictionary that maps
# variables to their corresponding values
class State:
    def __init__(self):
        self.state = {}

    def setValue(self, var, value):
        self.state[var.name] = value

    def getValue(self, var):
        return self.state[var.name]

    def __str__(self):
        statestr = '{'
        for var in self.state:
            statestr += '<' + var + ", " + str(self.state[var]) + '>, '
        return statestr[:-2] + '}'

# Meaning function of Assignment
# M(Statement statement, State state)
def M_Assignment(statement, state):
    state.setValue(statement.target, M_Expression(statement.source,
state).value)
    return state

def main2():
    # a = b + 3 x c {<a, 5>, <b, 1>, <c, 3>}
    vara = Variable('a')
    varb = Variable('b')
    varc = Variable('c')
    val3 = Value(3)
    expr = BinaryExpression(varb, '+', BinaryExpression(val3, '*', varc))
    assignment = Assignment(vara, expr)
    state = State()
    state.setValue(vara, 5)
    state.setValue(varb, 1)
    state.setValue(varc, 3)
    newState = M_Assignment(assignment, state)
    print(newState)

if __name__ == "__main__":
    #main1()
    main2()

```

Fun Facts about Assignment Semantics in C and Java

- C treats assignments as expression, while Java doesn't

```

/**
 * Exp.c
 * Ying Li
 * 10/15/2019
 */

#include <stdio.h>

int main () {
    int a = 1;
    int b = 0;

    if ((a = b)) {
        printf("%d == %d \n", a, b);
    } else {
        printf("%d != %d \n", a, b);
    }

    return 0;
}

```

```

/**
 * Exp.java
 * Ying Li
 * 10/15/2019
 */
public class Exp {

    public static void main (String[] args) {
        int a = 1;
        int b = 2;

        if (a = b) { //(a == b)
            System.out.printf("%d == %d \n", a, b);
        } else {
            System.out.printf("%d != %d \n", a, b);
        }
    }
}

```

- What are the outputs of Exp2.c? [a = 2, b = 2, c = 2; a = 0, b = 0, c = 3]

```

/**
 * Exp2.c
 * Ying Li
 */

#include <stdio.h>

int main () {
    int a = 1;
    int b = 0;
    int c = 2;

    a = b = c;
    printf("a = %d, b = %d, c = %d \n", a, b, c);

    a = 1;
    b = 0;
    c = 2;

    c = 3 + (a = b);
    printf("a = %d, b = %d, c = %d \n", a, b, c);

    return 0;
}

```

- In assignments `a = b = c` evaluates `c`, assigns that to `b`, then assigns that to `a`.
- In expressions `c = 3 + (a = b)` assigns the value of `b` to `a` and then adds 3 to the value and assigns it to `c`.