

Comp 311
Principles of Programming Languages
Lecture 18
Implementations of Assignment and Mutation

Corky Cartwright
October 6, 2010

Parameter Passing Examples

Recall:

```
procedure Sum(int x, int y, int n) {  
    // actual x must occur free in actual y  
    int sum = 0;  
    for (x = 0; x < n, x++) sum = sum + y;  
    return sum;  
}
```

```
int j, sum = 0;  
int[10] a;  
for (int i = 0; i < 10; i++) a[i] = i; //  
    initialize a  
sum = Sum(j, a[j], 10));           // compute the sum  
print(j, sum)                       // print the  
    result
```

Parameter-Passing Scenarios

- call-by-value: in **Sum**, the local variables **x**, **y**, **n** are new boxes with contents **0**, **0**, **10**. Hence, the loop repeatedly adds **0** to sum which is initially **0**. Hence, the returned result is **0** and the program prints **0 0**
- call-by-name: in **Sum**, the local variables **x**, **y**, **n** are bound to suspensions with bodies **j**, **a[j]**, **temp** closed over the calling environment where **temp** is a variable generated by the compiler/interpreter holding the value **10**. In the body of **Sum**, the loop repeatedly evaluates the suspensions for **x**, **y**, **n**. **x** always evaluates to the box corresponding the variable **j**, but **y** evaluates to the box for array element **a[j]** which depends on **j**. Hence, the loop sums the integers between **0** and **9** and increments variable **j** on each iteration. The program prints **10 45**
- call-by-reference: in **Sum**, the local variables **x**, **y**, **n** are bound to the boxes corresponding to the variable **j**, the array element **a[0]**, and a dummy box created to hold the actual parameter **10**. (We are assuming that the language boxes constants passed by reference instead of declaring the program syntactically incorrect.) In the body of **Sum**, the loop repeatedly adds **0** to sum, incrementing **j** by **1** on each iteration. Hence, the program prints **10 0**
- call-by-value-result: in **Sum**, the local variables **x**, **y**, **n** are bound to the boxes containing copies of the value of **j**, the value of **a[0]**, and **10** (We are assuming that the language boxes constants passed by value-result instead of declaring the program syntactically incorrect.) In the body of **Sum**, the loop repeatedly adds **0** to **sum**, incrementing the local variable **x** on each iteration. On exit, the values of **x**, **y**, **n** are copied into the variables (boxes) **j**, **a[0]**, dummy box created to hold the value **10**. Hence, the program prints **10 0**
just like call-by-reference. The code in this program does not depend on whether the local variables are copies or originals.
- call-by-result: the parameter passing convention does not make sense in this example because it fails to initialize the formal parameters **x**, **y**, **n**. The program is ill-formed for this parameter passing method.