# Week 13 - Day 1 (Call-by-value)

## Contents

## Week 13 - Day 1 (Call-by-value)

Apr 12, 2016

### Navigate using audio

## CALL-BY-VALUE

- CALL-BY-VALUE
  - every modern language
    - except c++
      - Audio 0:02:30

```
int swap(int a, int b) {
    int temp=a;
    a = b;
    b = temp;
}
```

... but in c

```
int x = 5
int y = 13;
// x still = 5
swap(x,y);
// x equals the old value of y now
swap(&x,&y);
int *z = &y;
// now x = old value of y, z still equals address of y
// Audio 0:07:30
swap(&x,z);
// you can change what z points to, but you can't change z itself
```

- Call by value means you can't change the value of a variable by passing it to a function

## CALL-BY-REFERENCE

```
int swap(int &a, int &b) {
    int temp=a;
    a = b;
    b = temp;
}
```

... now

```
int x = 5;
int y = 13;
// now this swaps
swap(x, y);
```

Audio 0:10:10

## We can do it in SCAM too

### Call-by-reference

```
; We capture the calling environment with "#"
(define (swap # $a $b)
    ; Audio 0:11:43
    (define temp (# $a))
    ; Set $a in the calling environment to whatever $b is in the calling environment
    (set $a (# $b) #)
    ; Audio 0:12:24
    ; Set $b to whatever temp is in the calling environment
    (set $b (temp #))
)
```

Audio 0:13:40

You can avoid needing a complicated macro pattern with call-by-reference

Usually just syntactic sugar for passing pointers around

Audio 0:15:50

## Call-by-value-result (Ada)

## Ada

- Ada had 500 keywords
- It also had a call-by-value-result

```
function f(a,b) {
    a = 0;
    // right before we return from the function, we update to be call-by-reference
    // Audio 0:19:03
    return expr;
}
```

Got distracted…

Sorry

## CS 403 - 001 Spring 2016

- CS 403 - 001 Spring 2016
- jmbeach1@crimson.ua.edu

- jmbeach

Website for notes and study material for CS 403 (Programming Languages) at The University of Alabama Spring 2016