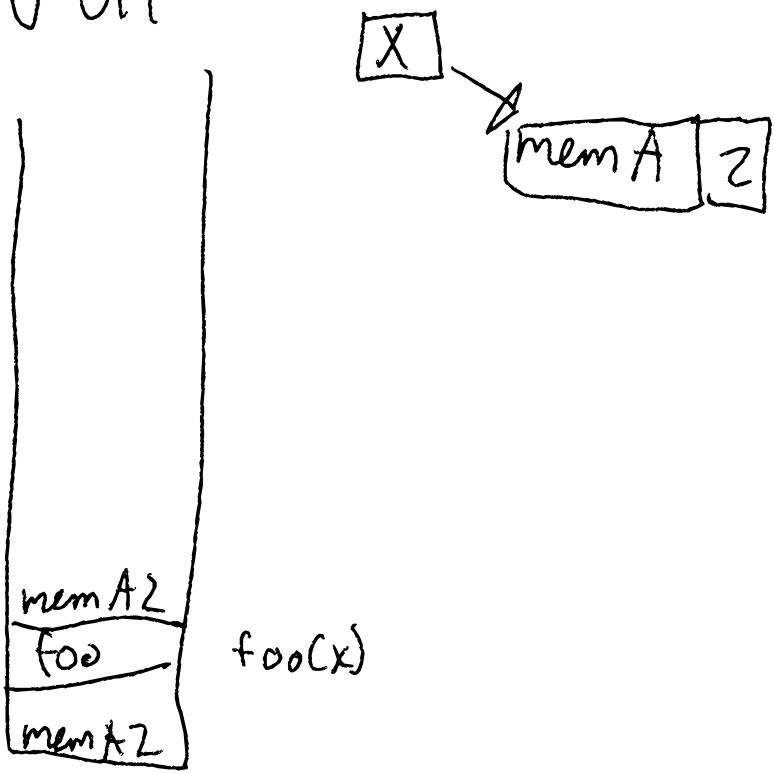
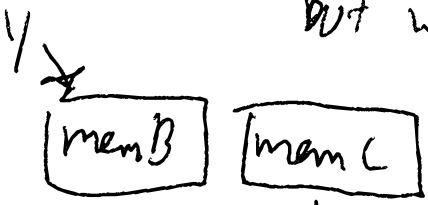
pass by ref vs Val

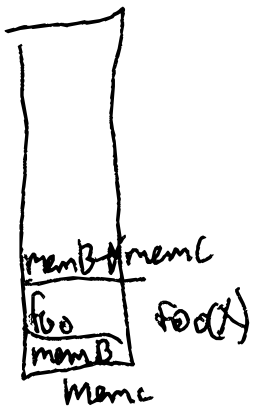
Val



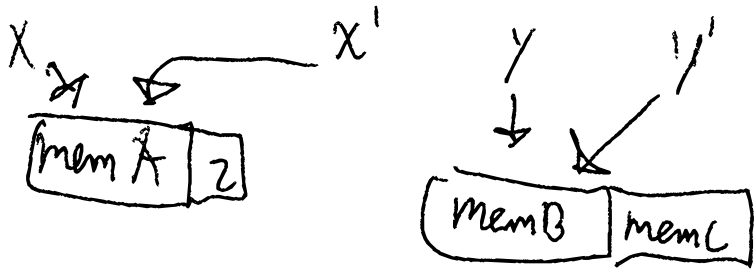
Sharing

pass by val
but w/ classes





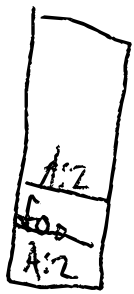
Reference



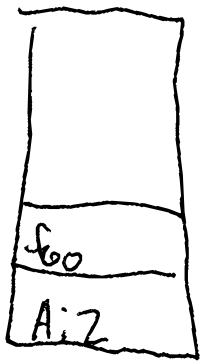
Pass by ref passes the x'

New pass by's

Value/result typically slower.



when updated it updates the org when func is done, not instant.



foo(X,Y,X) foo(a,b,c)
call

Order you copy out matters
typicall L→R

Pass by name wtf

```
f(x,y)
  return x+2y
```

macro → text substitution. before compiling.

f(2,3)

f(b[0]a[2])

Variation of inline funcs.

examples

```
func foo(a,b)
  a = b
  b = b+1
```

X = 1

Y = 2

Foo(x,y)

What is x,y after func

Val

1,2

reference

2,3. its 3,3 if aliased

Val/result

2,3

name

2,3

```
func foo(a,b)
  a = b   x = y   x = 2
  b = b+1 y = y+1 y = 2+1
```

Point
x
y

foo(x)
a.x = 3
a = new point(2,2)

a = Point(1,0)

foo(a)

Value
nothing chages
:
0

Sharing
foo(x) # mema: memb: a
a.x = 3 a = 3,0
a = new point(2,2) new mem addr
Creates and heap storage

mem C: memB: a' # (2,2)
but a dosnt see the
update outside func so ends
up returning 3,0

```
func foo(a, b, c)
    a = c+1
    c = a*b

x = 1
y = 2
foo(x, x, y)
```

Val
1
2
ref
3
6

Val/result
3 or 1 depends on fill
4

name
3
& a and b are replaced w/x

```
func foo(a, b, c)
    a = c+1      x = y+1
    c = a*b      y = x+x
```

Optional / Default fcns.

foo(int x=1)

typically make make diff fcns

foo(int x)

foo(L(1))

Named params

passed in as a hash map.
pass by name.

Parameter Lists.

printf (_ _ _, ___)

to be filled. vars

Not type safe

Multi return

f(2) dont know if passed by ref/val