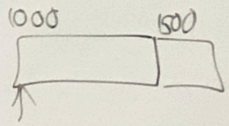
Where to put arrays?

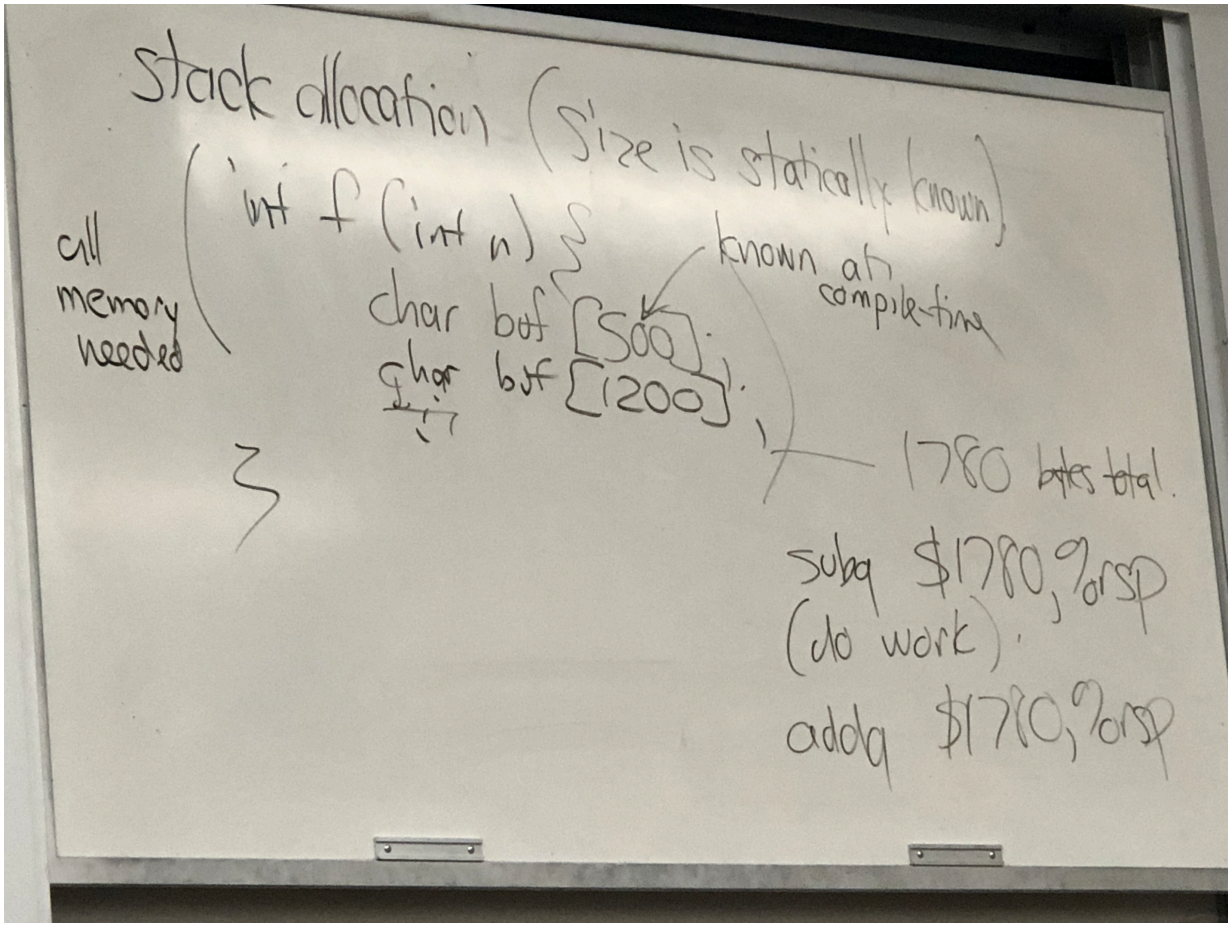
Array allocation strategies.

- Static allocation
  (before program runs)
  + no run overhead
  + no crashes due storage exhaustion.
  + compiler can optimize more.
  − hassle to change sizes — recompile
  − overallocation is common.
  − hard to have temp arrays



For next time

Dybvig §§1 – 3.4
(Scheme)

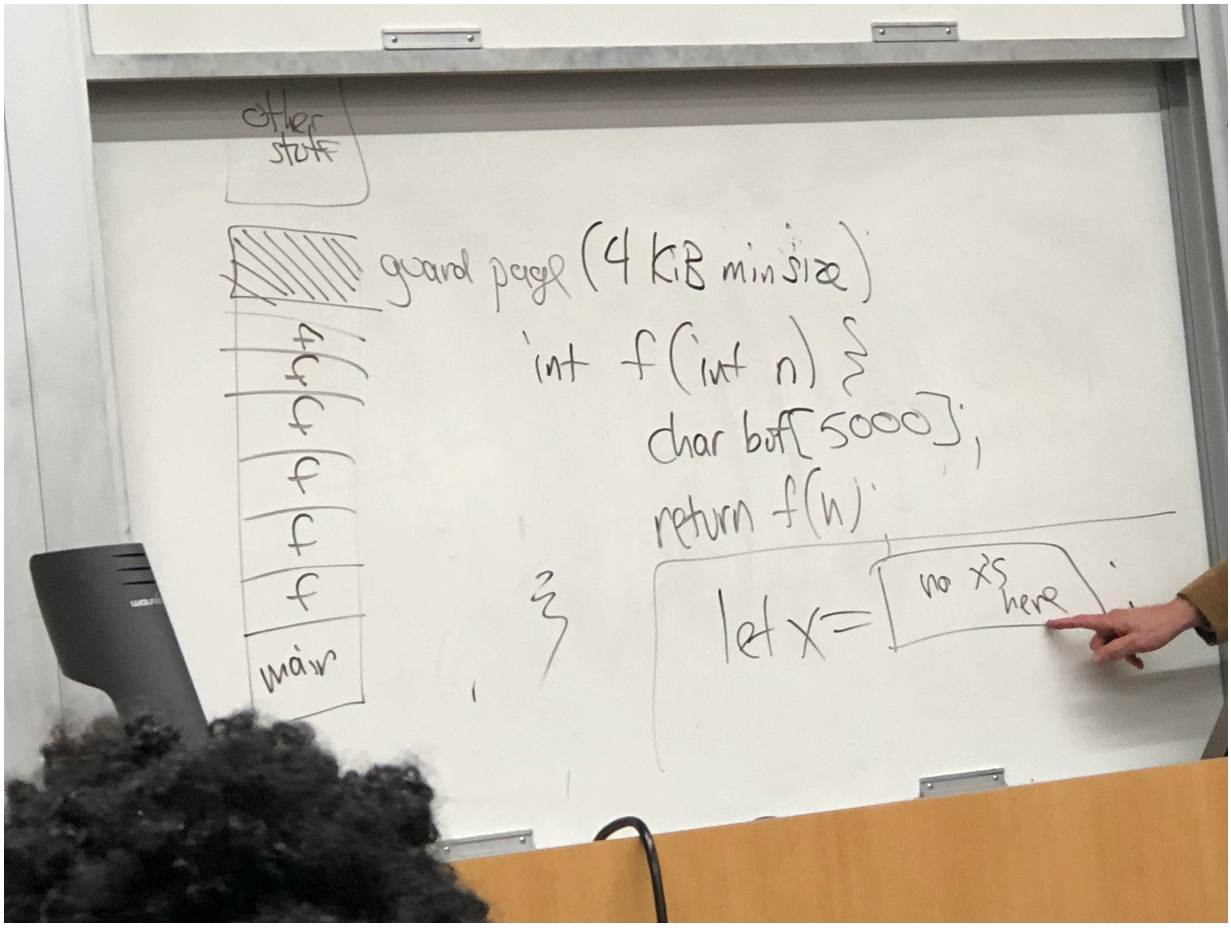


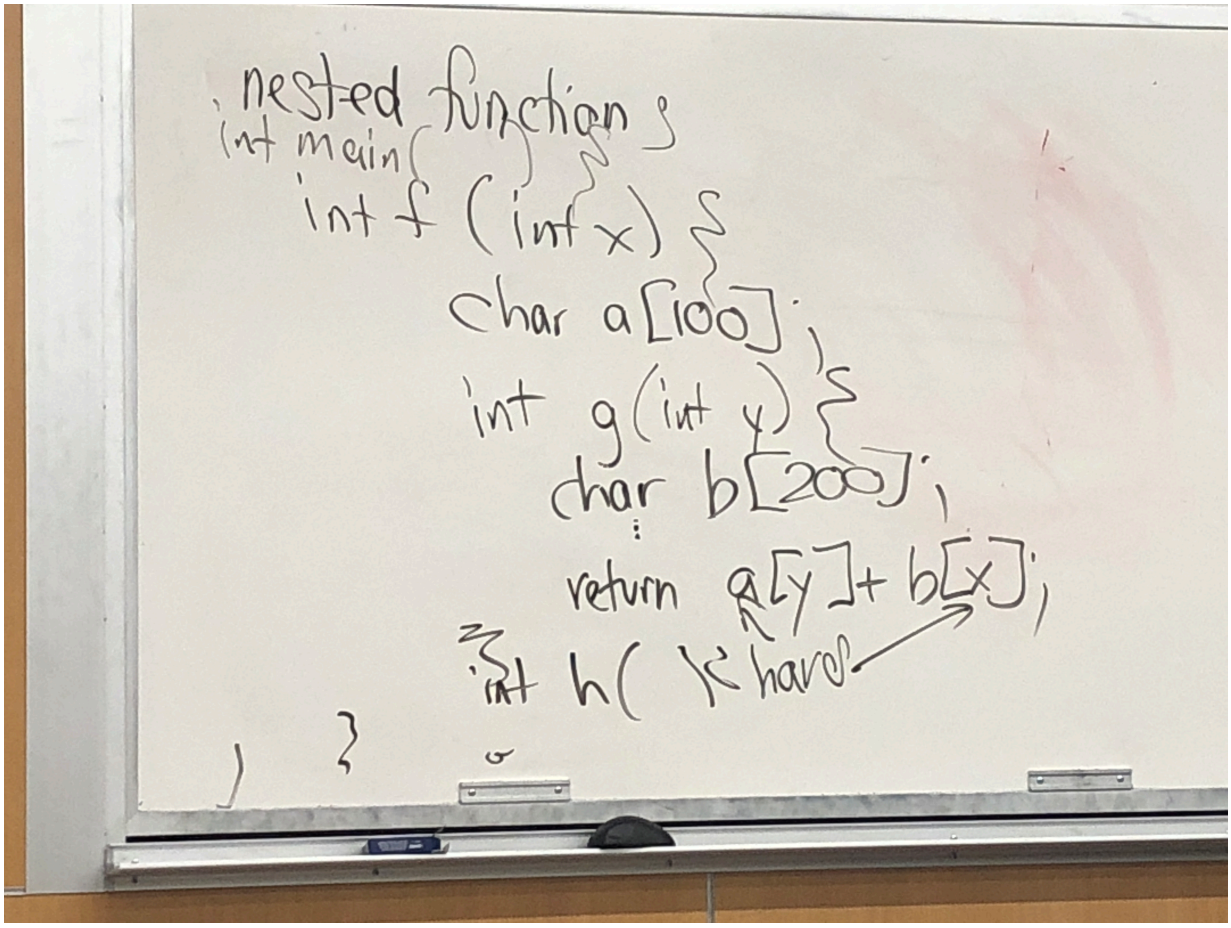
Array allocation is really fast if there's fixed stack allocation with static arrays

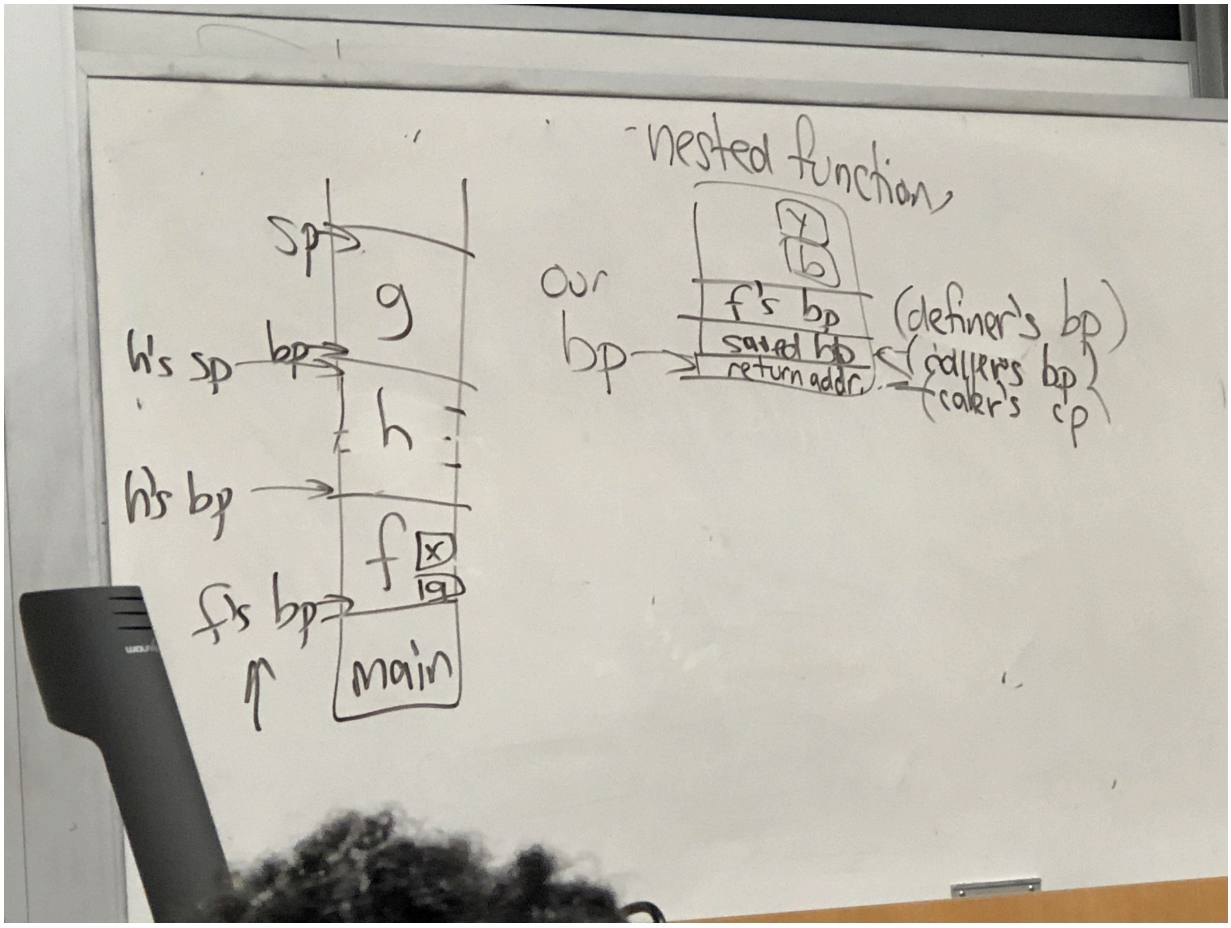
- used by C/C++

stack overflow problem

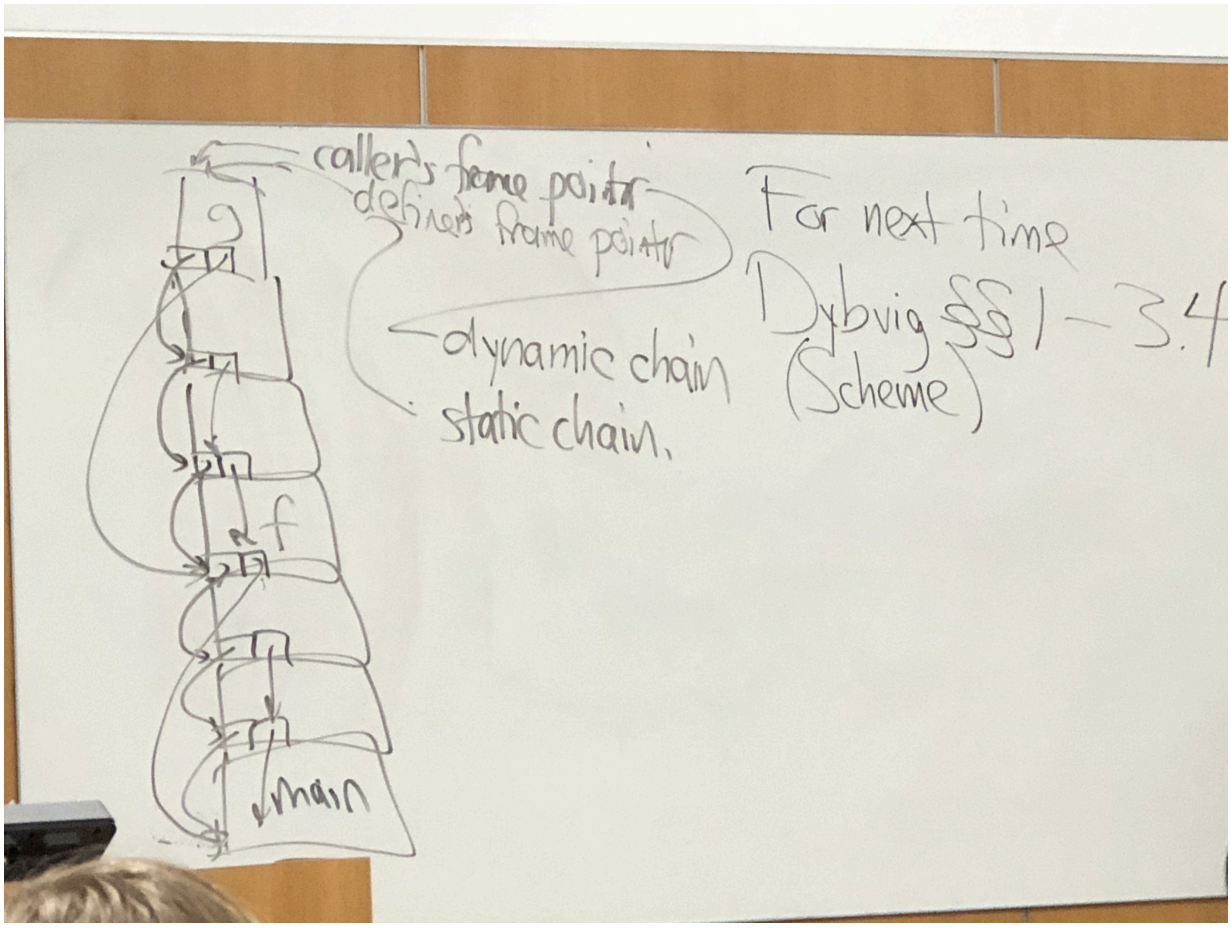
We see that with char buf[5000] we can jump straight over the guard page (which is 4 KiB min size)


other stuff
guard page (4 KiB min size)
int f(int n) {
char buff[5000];
return f(n);
}
f
f
f
f
main
let x = [no x's here]


nested functions
int main(
int f (int x) {
char a[100];
int g(int y) {
char b[200];
return a[y] + b[x];
int h(x hard
}
}

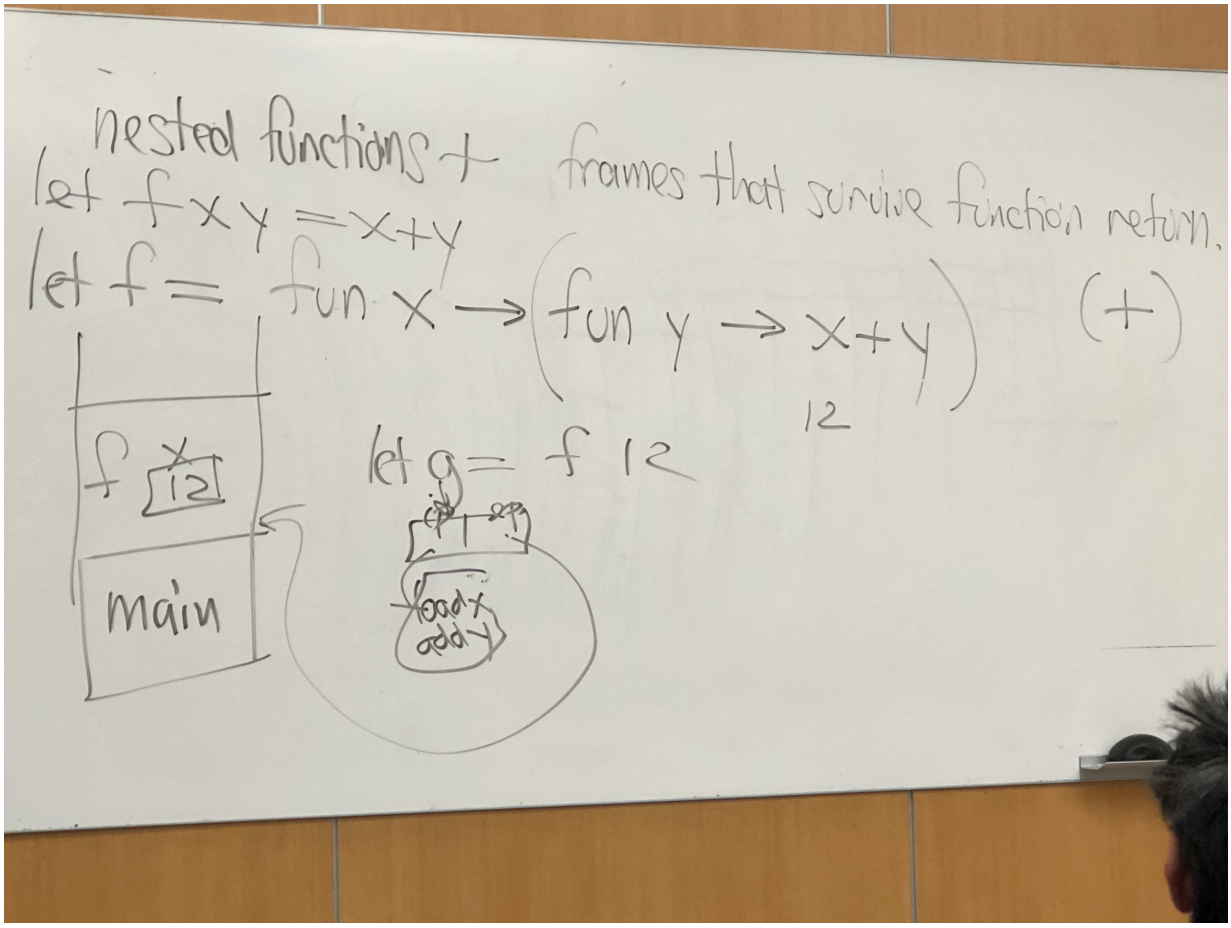


Nested functions need function pointers.



activation records = stack frames
In OCaml, there is no such thing as activation records on STACK

- doesn't use stack.
- Uses heap


nested functions + frames that survive function return.
let f x y = x+y
let f = fun x -> (fun y -> x+y)
(+)
12
f x 12
main
let g = f 12
body
add y

— 2/20/18