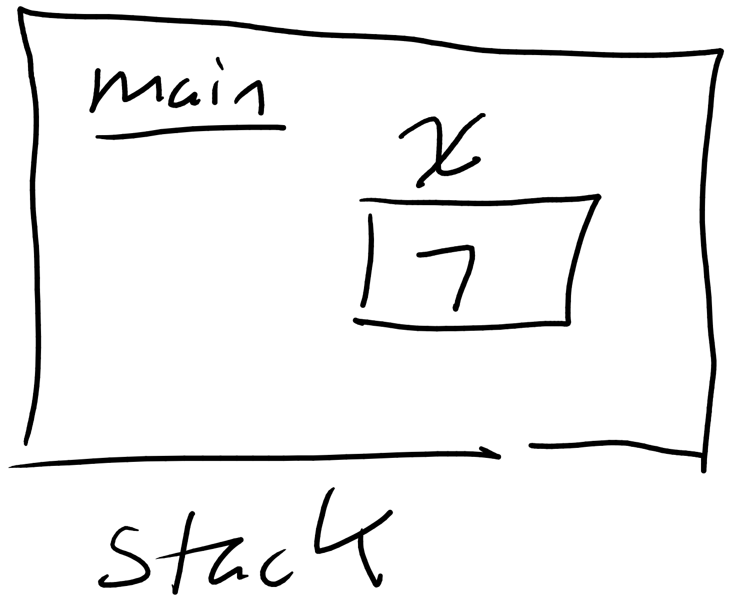
C/memory allocation/comparisons to other languages

$x = 7$ where does the 7 go?

- stack?
- heap?

C `int x=7;`

```
int main() {
      int x=7;
 }
```

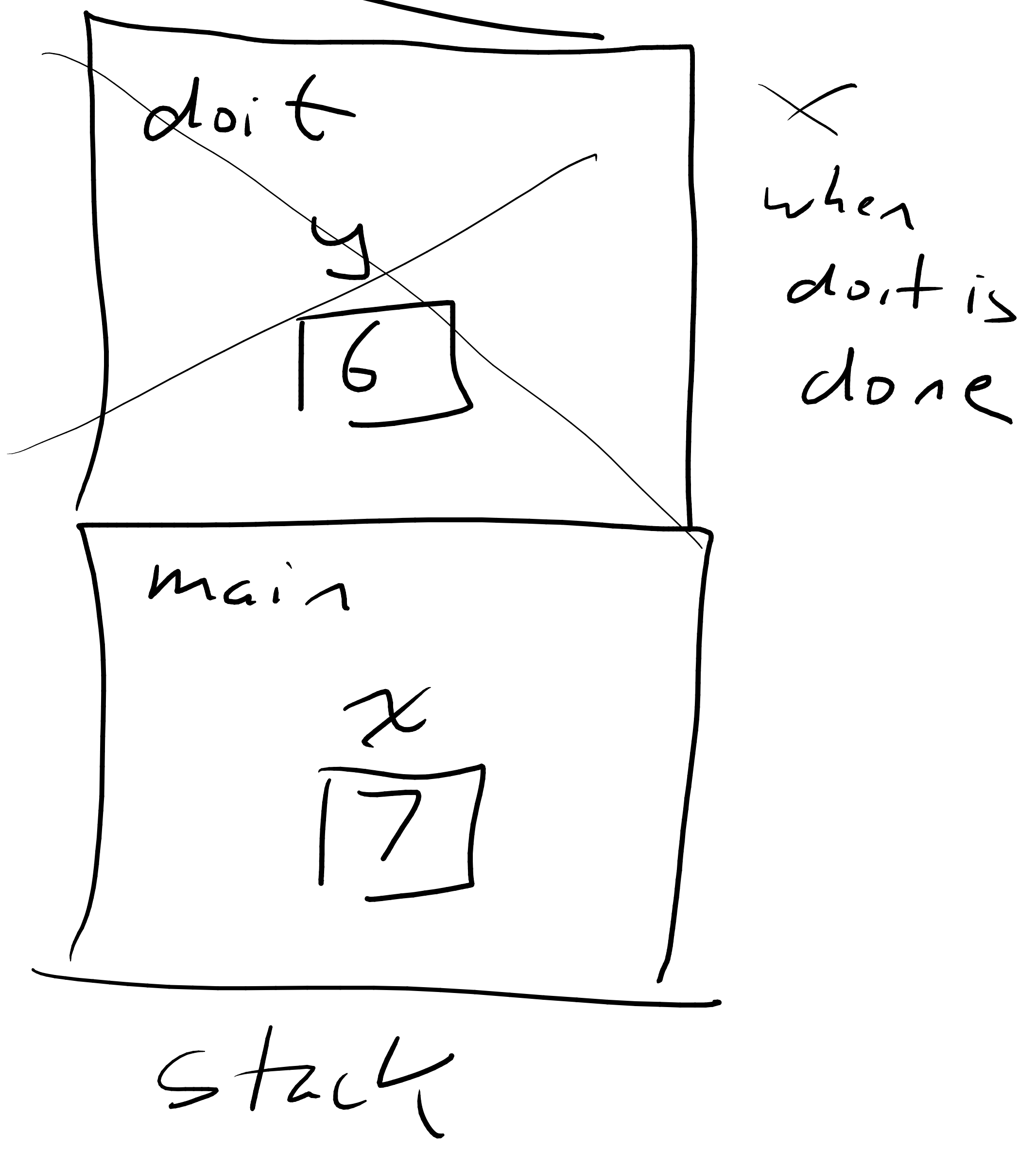


These stack frames come and go as functions come and go

(Olin 30618, 8–9 pm, prefect session)

```
int doit() {
   int y=6;
   return y;
}

int main() {
   int x=7;
   doit();
}
```



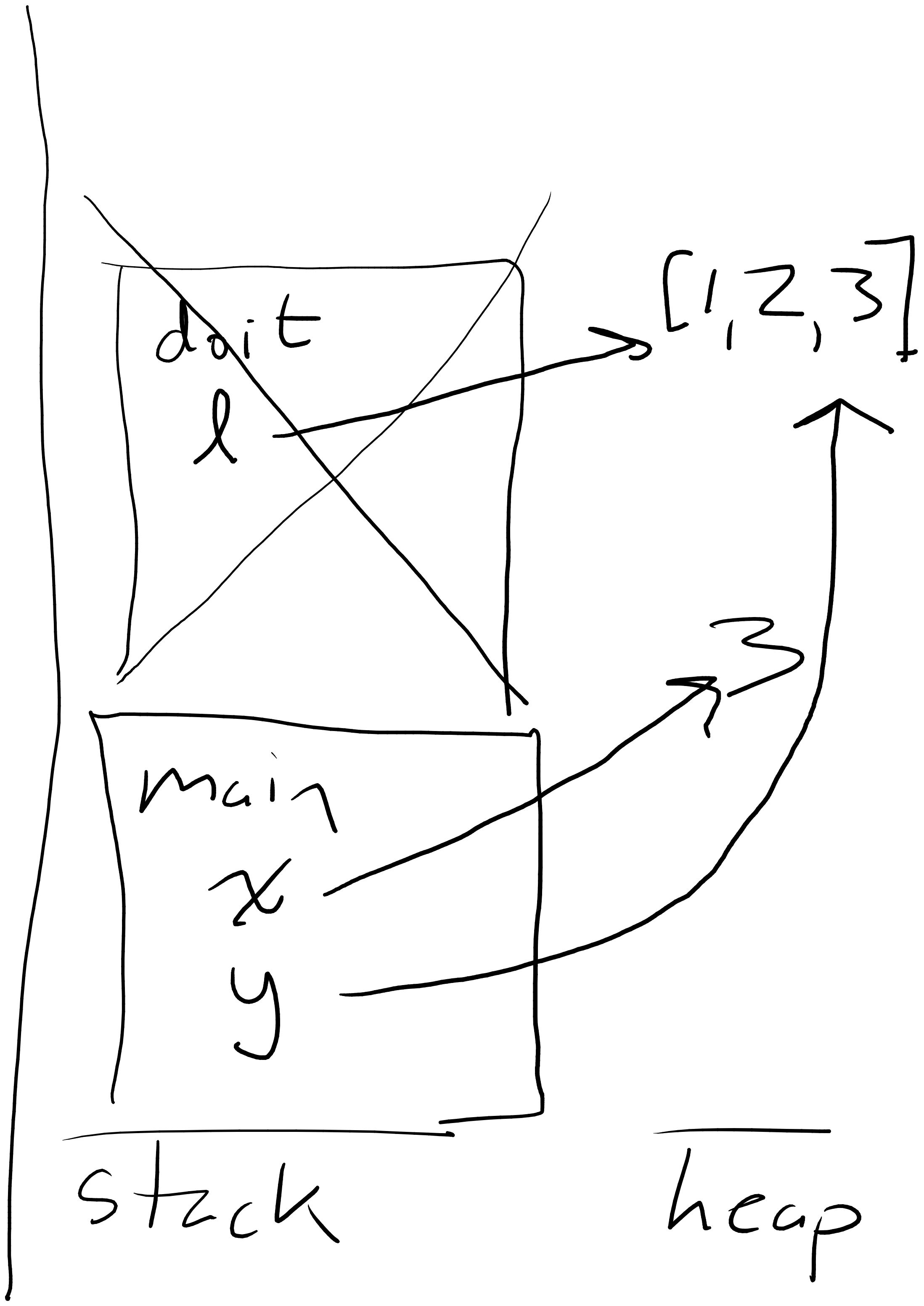
When a function ends, its memory is popped off the stack, and it goes away.

Another place to store data is in the "heap". Memory in the heap is allocated when you ask for it, and deallocated when you are done (either by saying so, or w/ garbage collection.)

Python

```
def doit():
    l = [1, 2, 3]
    return l

def main():
    x = 3
    y = doit()
    print(y)
```



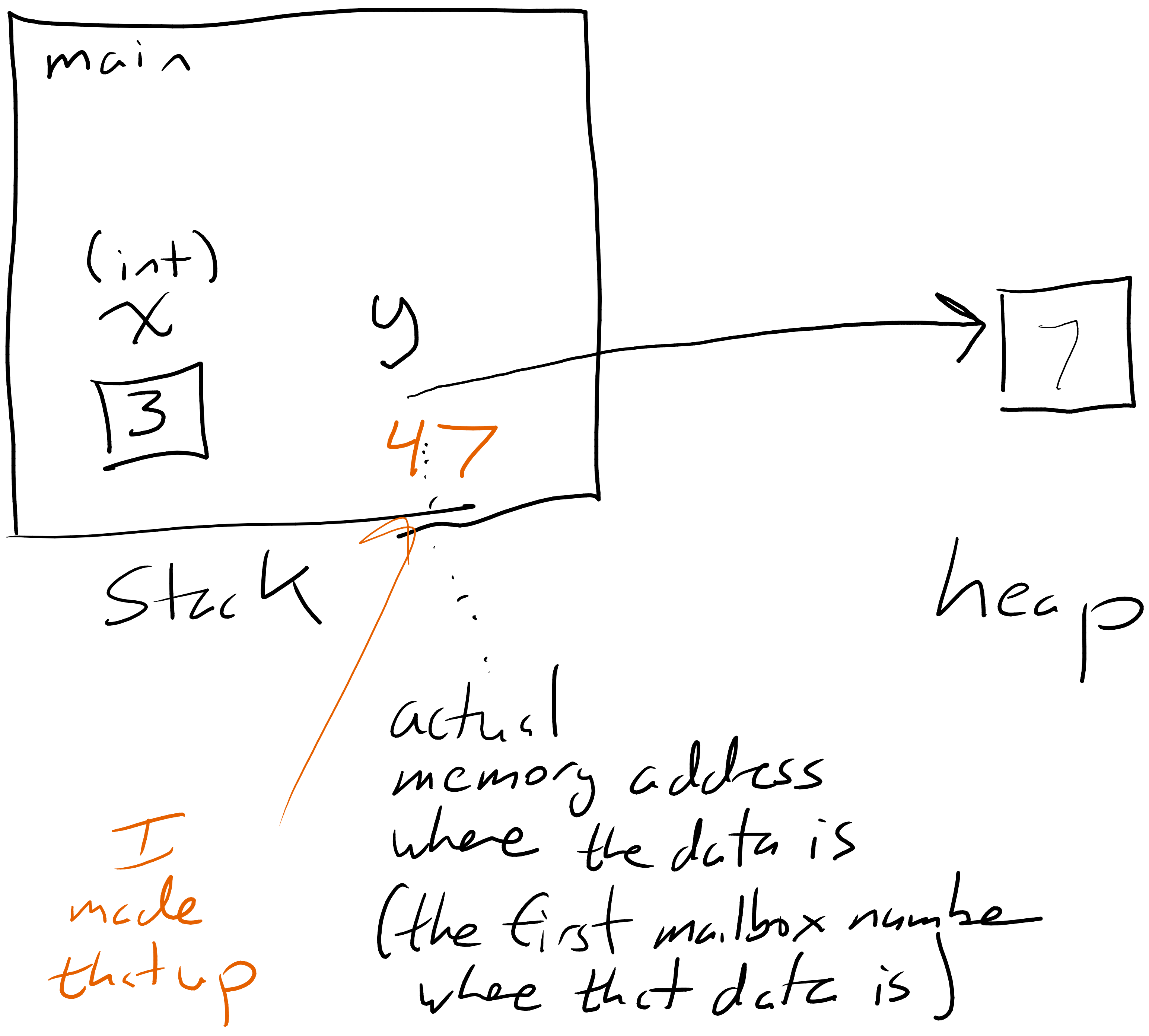
main()

Inaret model like Python, variables are in the stack, but the data is in the heap.
- variables come and go based on the functs they're in but heap data is "forever" except when garbage collected.

In Python,
- varicbles are in stack
- data is in heap

In C
- varicbles are in stack
- data is in stack "by default" but you can choose to put it in the heap
  - typiclly manage it vic pointes

Memory diag for ptrfun.c

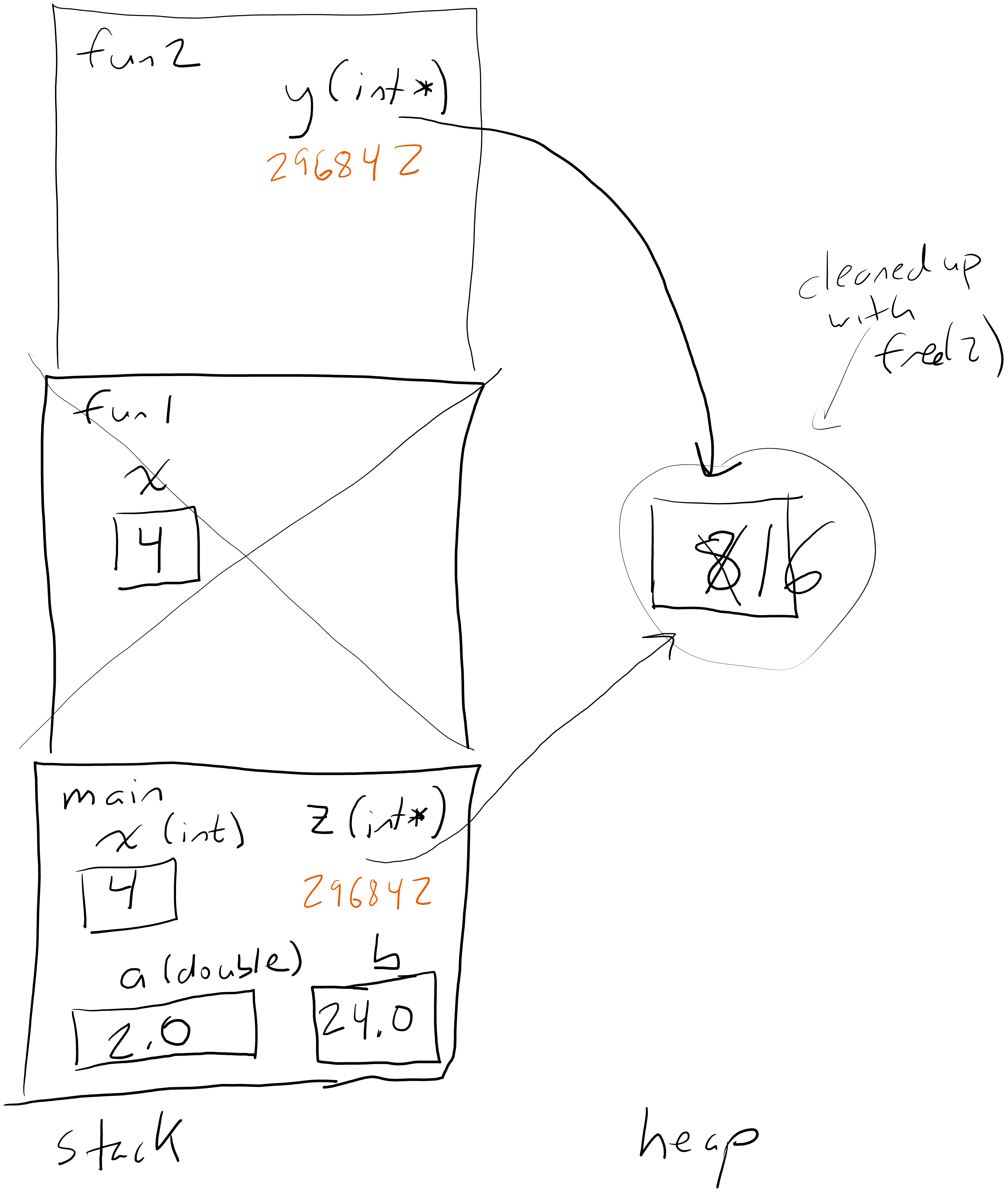

main
(int)
x
3
y
47
7
Stack
heap
actual memory address where the data is (the first mailbox number where that data is)
I made that up

int * y = malloc(sizeof(int));

y is a pointer to an int.

The type of y is int*

stack vs heap example

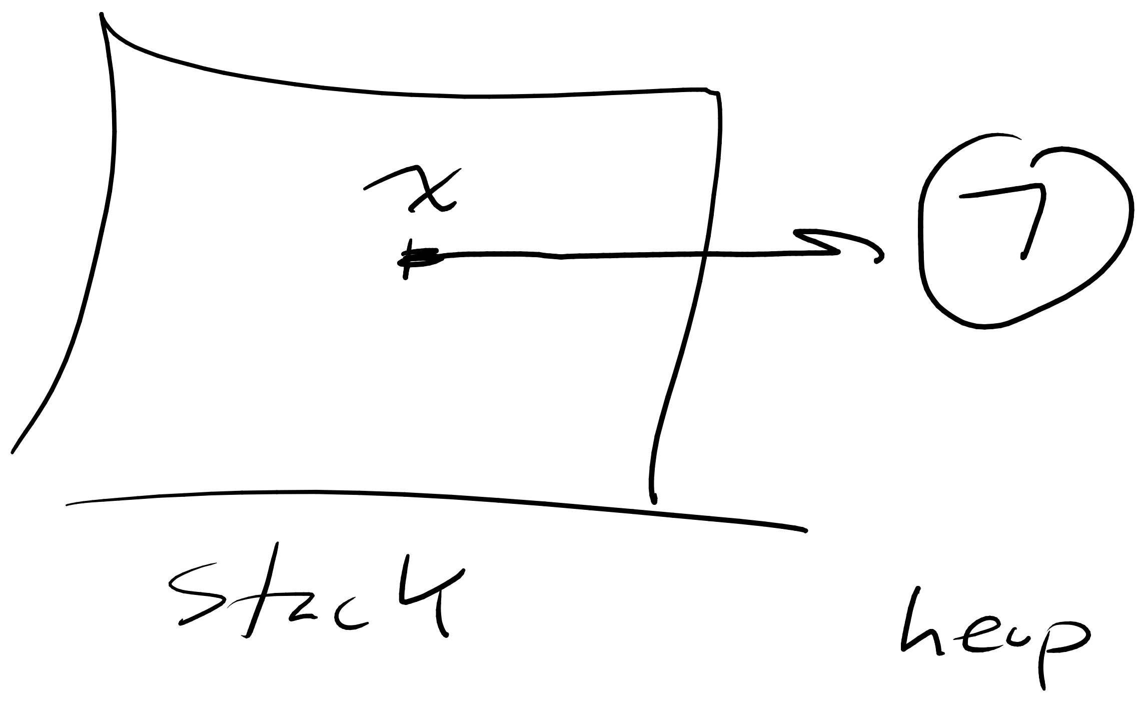


Python

```
def main():
    x = 7

    print(x)

main()
```



print(x)

↖ follow the pointer
the C "*" is implicit

In Python how do I actually print the memory address?

YOU CAN'T