

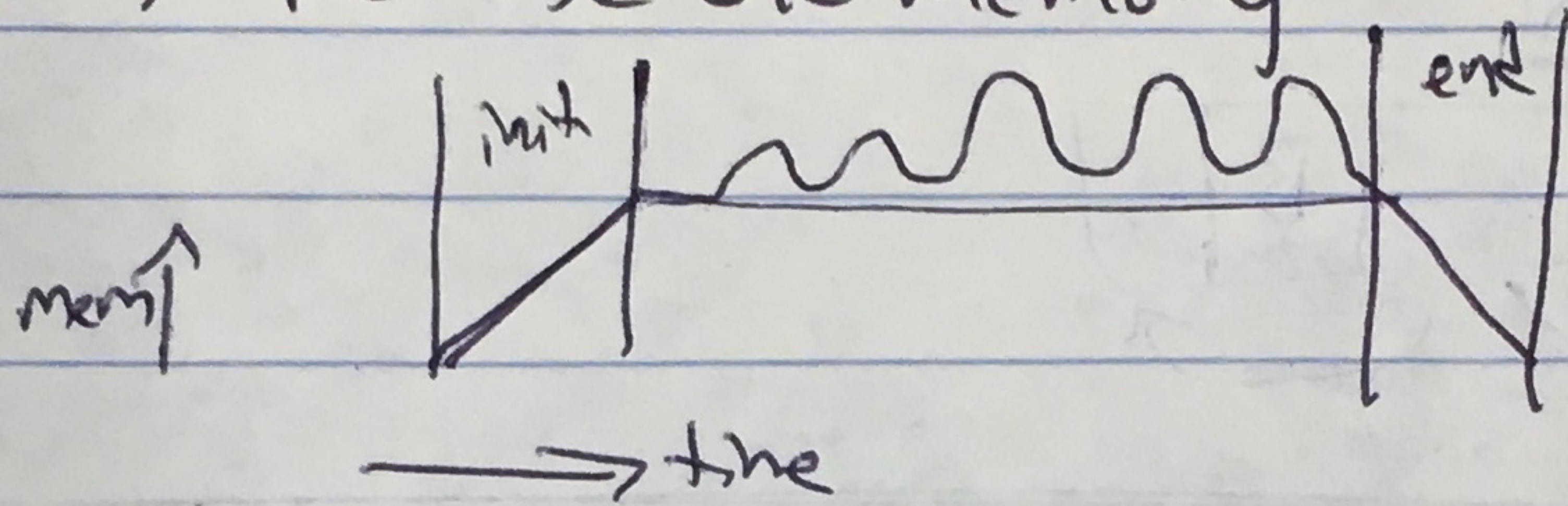
1

(aStore 9 (numV 13)
8 (numV 12)
9 (numV 10))

memory \rightarrow global \rightarrow finite / fixed
 \rightarrow linear

Finite memory

\hookrightarrow re-use old memory



What can be re-used?

- if don't use it in the future \rightarrow free it
- duplicated data (!) \leftarrow hash collision
- could re-calculate \leftarrow cache

free(p) \Rightarrow don't need it \rightarrow humans just know

```
int* f() {
```

```
  int* p = malloc(4);
```

```
  return p;
```

```
}
```

extent of values

\neq

scope of variables

intuition = "you can't get to it"
truth

soundness
don't free
early

completeness
don't free
late

provability

```
int* p = malloc;
```

```
if (f(x) == 0)
```

```
  *p = ...
```

```
  else {
```

```
    error // not mention p, ...
```

```
}
```

\Rightarrow f = "return 0 if x
as a T.M. halts"

2

malloc

ideal: $O(1)$

free

ideal: $O(1)$

cons

- trust people

- m is how much memory

- n is # of malloc

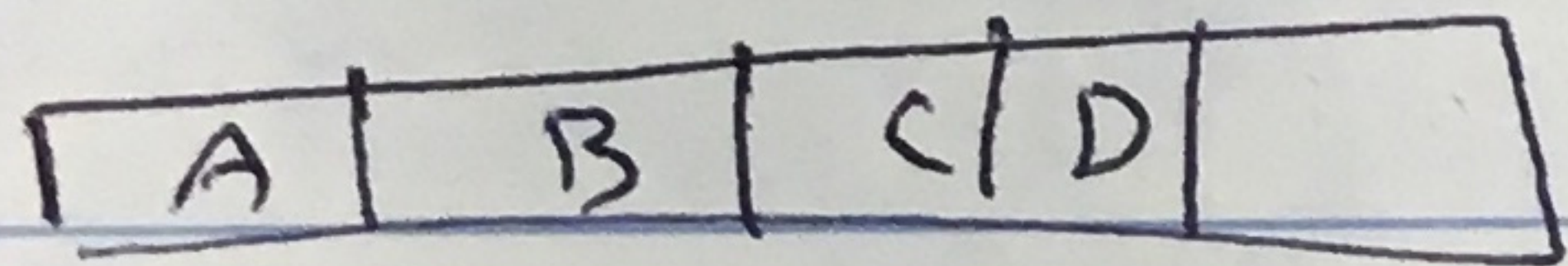
memory head

ideal: $O(1)$

tree: $O(1)$

~~$O(1)$~~

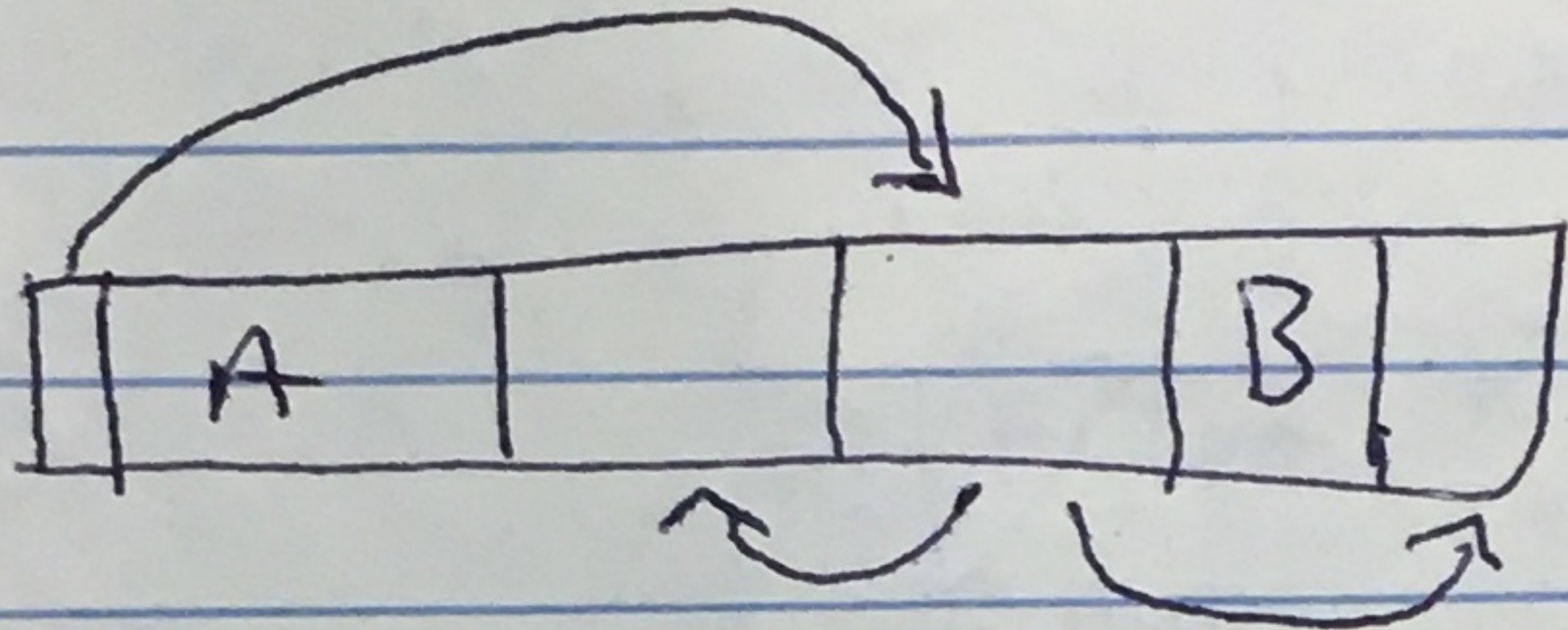
return (free_ptr + size)



free(B)

$O(\lg m)$ (some $O(\lg n)$) →

malloc + free



s mark pointers

r eference counting

up() if (p->count == MAX)

down() if max do nothing

p->count++

if (-p->count == 0)

→

free(p)

posn * m = f(...)

m->up()

g(m)

m->down()

global = m

global->up()

malloc: same $O(\lg n)$

soundness: trusts humans

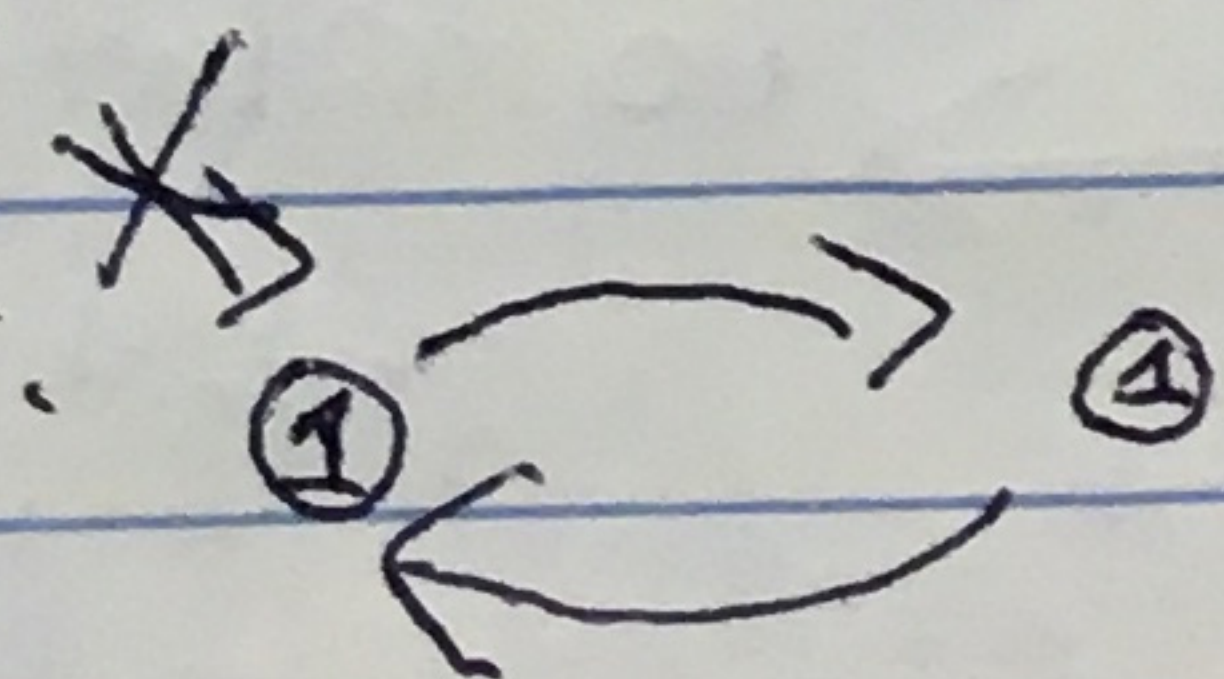
free: same $O(\lg n)$

how completeness:

mem: $(1 + \text{sizeof}(t))$

use: $O(\text{references})$

program



cyclic structures
are never freed