CS 103000
Prof. Madeline Blount

Week 12:
POINTERS

attendance link:

https://cs103-week12.glitch.me/

Dall-E 2: cats learning C++ in the forest on '90's technology

THE MYSTIQUE OF POINTERS

tricky but IMPORTANT …

https://cs103-week12.glitch.me/

https://www.youtube.com/watch?v=2ybLD6_2gKM

THE MYSTIQUE OF POINTERS

tricky but IMPORTANT …

https://cs103-week12.glitch.me/

https://www.youtube.com/watch?v=DTxHyVn0ODg

THE MYSTIQUE OF POINTERS

tricky but IMPORTANT …

https://cs103-week12.glitch.me/

https://www.youtube.com/watch?v=t5NszblerYc

THE MYSTIQUE OF POINTERS

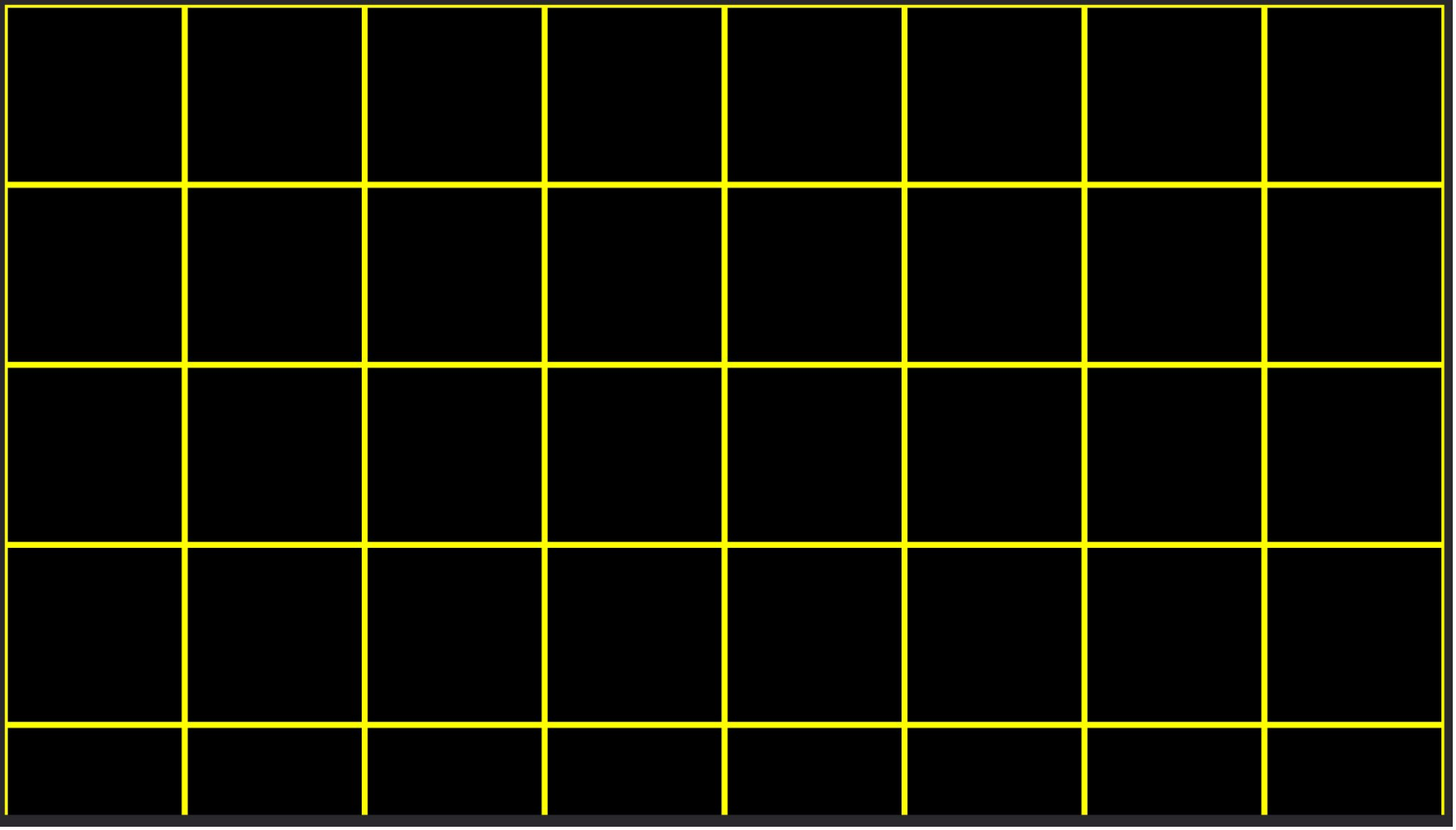
tricky but IMPORTANT …

`https://cs103-week12.glitch.me/`

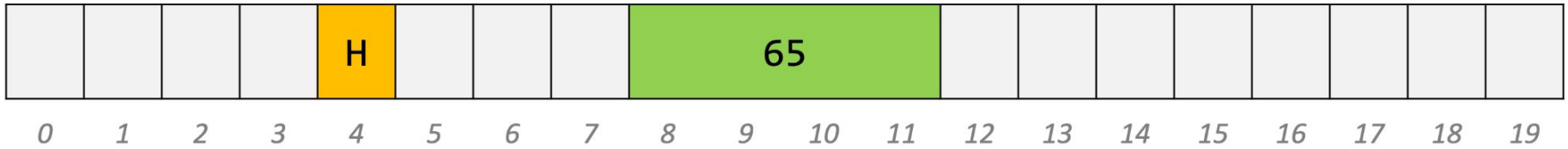
https://www.youtube.com/watch?v=i49_SNt4yfk

POINTERS give you the power to manage and directly manipulate **MEMORY**

MEMORY: for programs, RAM (random access memory)

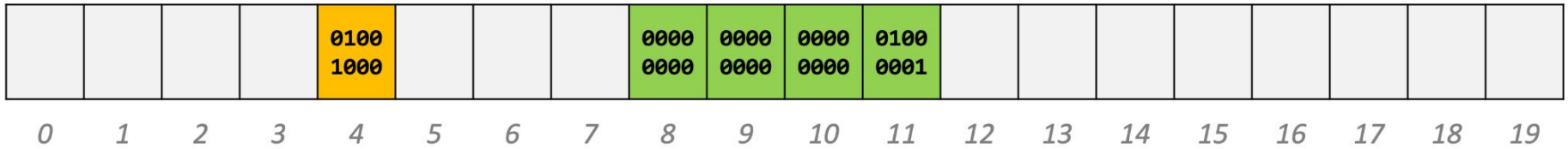


from Harvard CS50




```
char c = 'H';
int speedlimit = 65;
```

from Harvard CS50



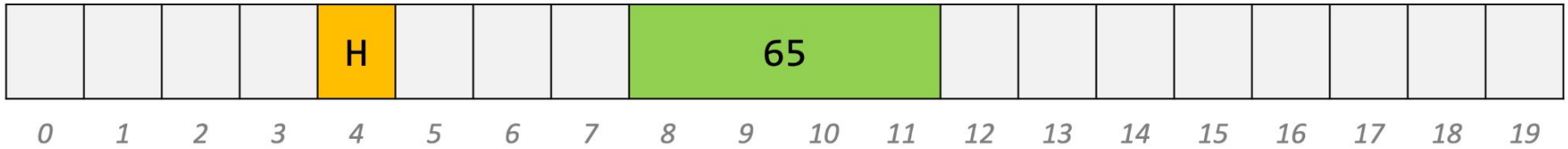

```
char c = 'H';
int speedlimit = 65;
```

from Harvard CS50

Memory Size

The size of a given data type is measured in bytes:

Data Type	Memory Size
`bool`	1 byte
`char`	1 byte
`int`	4 bytes
`float`	4 bytes
`double`	8 bytes
`std::string`	24 bytes




```
char c = 'H';
int speedlimit = 65;
```

from Harvard CS50


1
2
3
4


1
2
3
4
203
204
301
302
303
304

POINTER = MEMORY ADDRESS

POINTER = MEMORY ADDRESS

POINTER = integer that stores a memory address

POINTER = MEMORY ADDRESS

POINTER = variable that holds an integer which is a memory address

POINTER = MEMORY ADDRESS

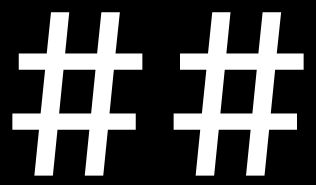
0x16d9d7470

hexadecimal

0 1 2 3 4 5 6 7 8 9 a b c d e f

from Harvard CS50

16 1



16 1

02

from Harvard CS50

16 1

09

from Harvard CS50

16 1

from Harvard CS50

11111111

from Harvard CS50

FF

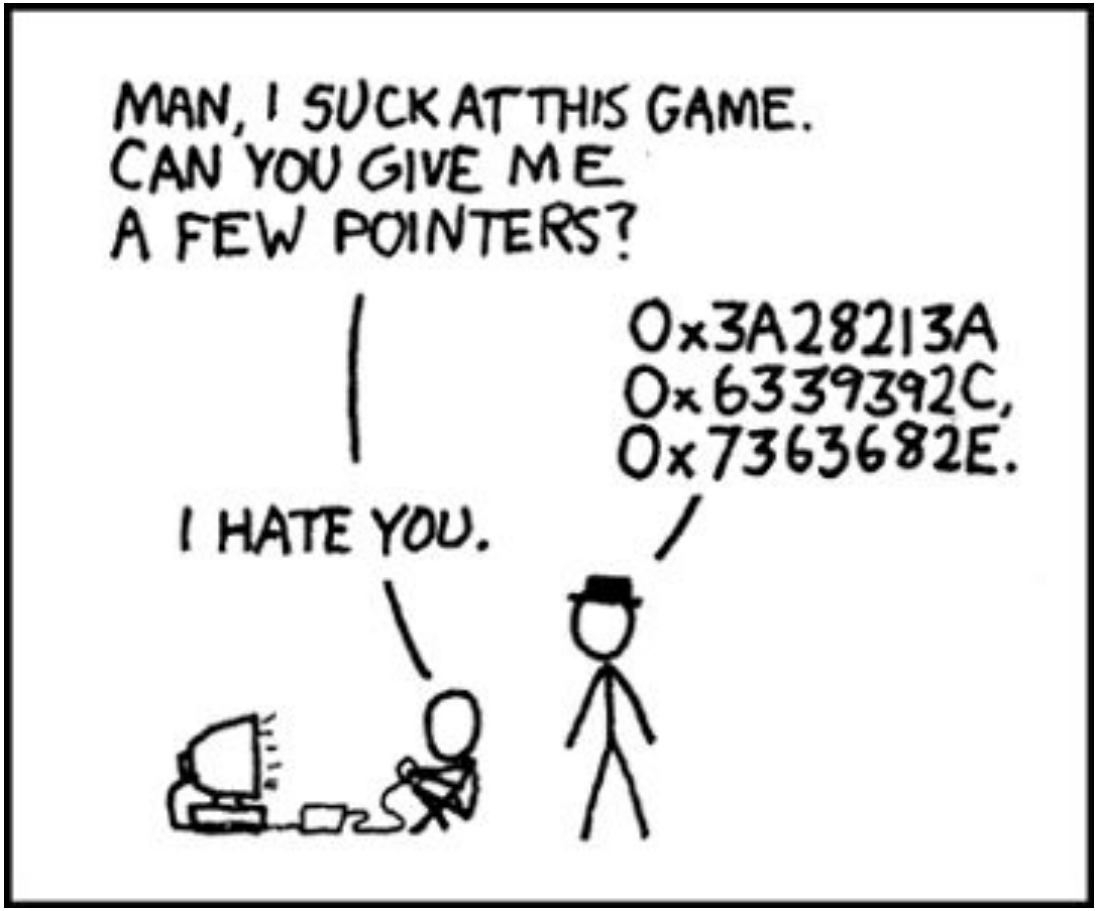
from Harvard CS50

POINTER = MEMORY ADDRESS

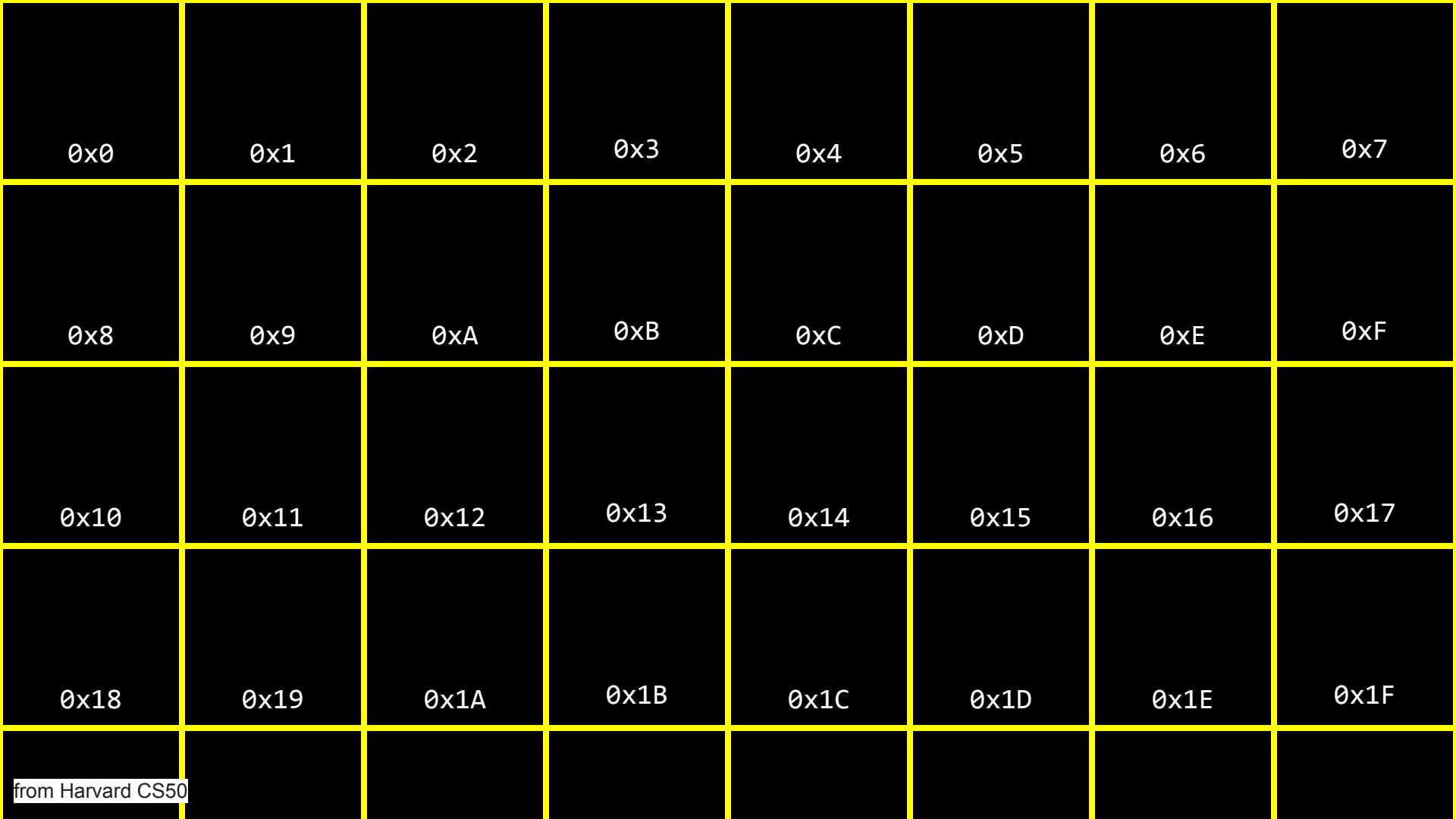
0x16d9d7470

6134002800

10110110110011101011010001110000

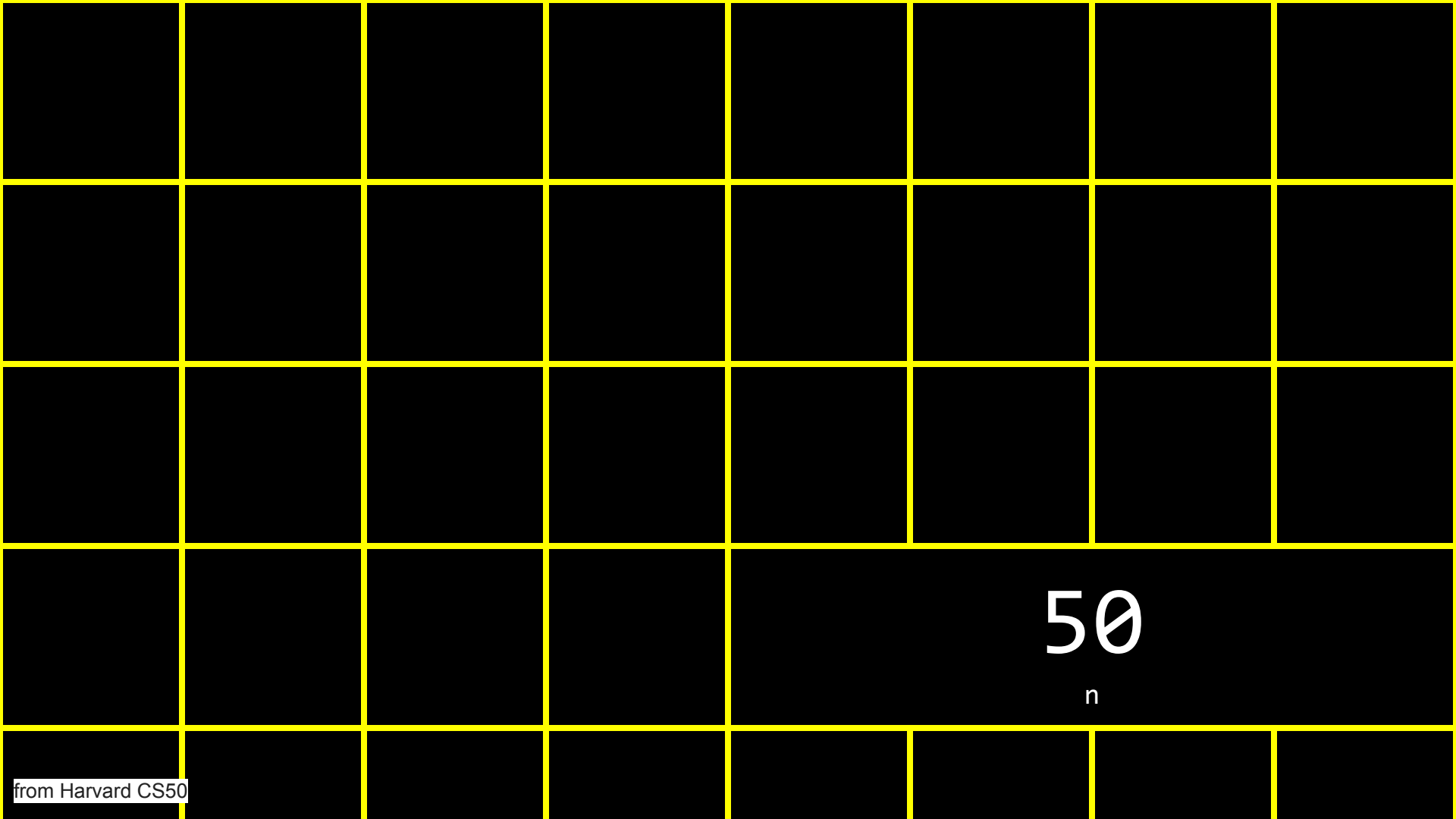


https://xkcd.com/138

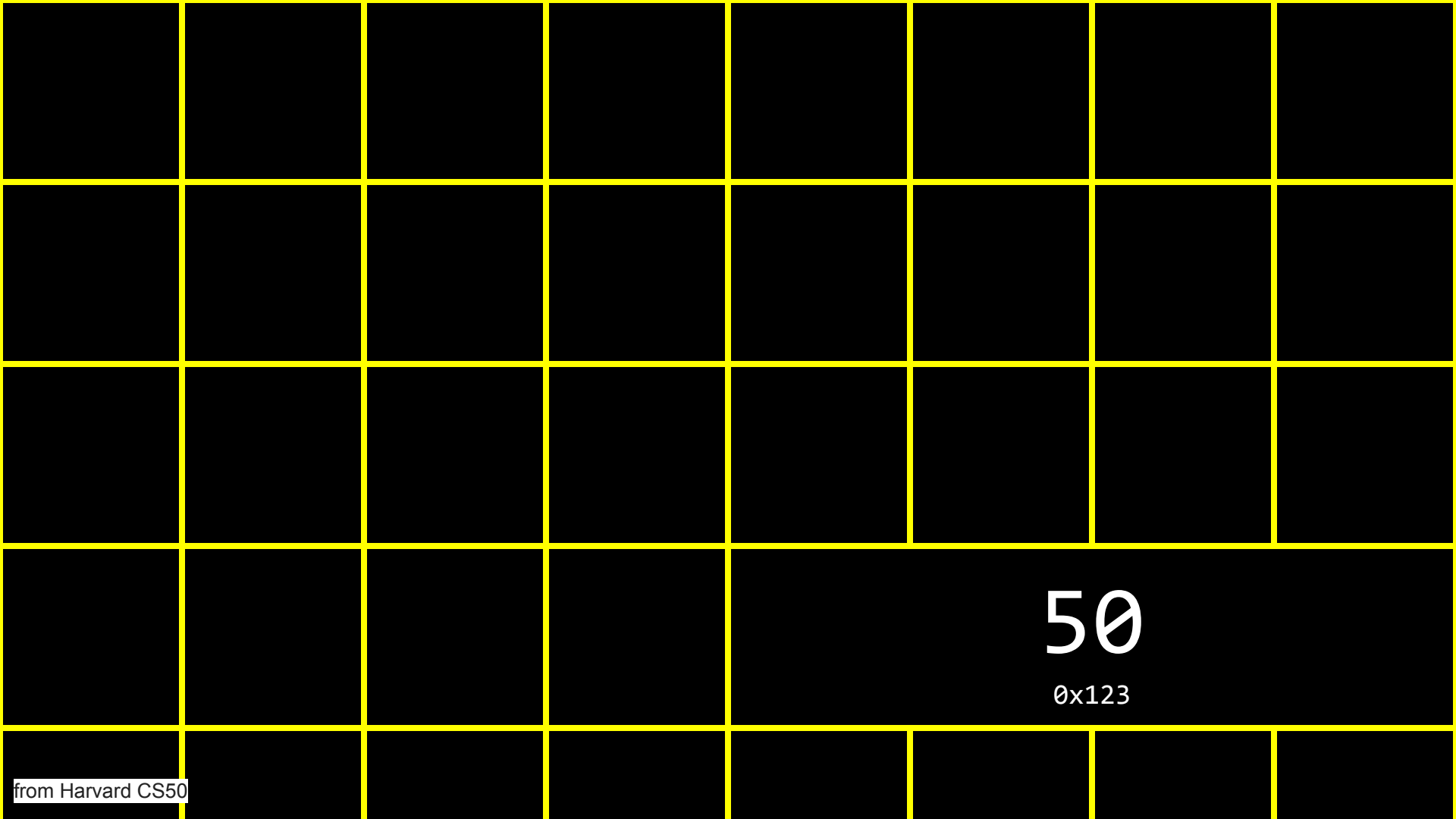


from Harvard CS50

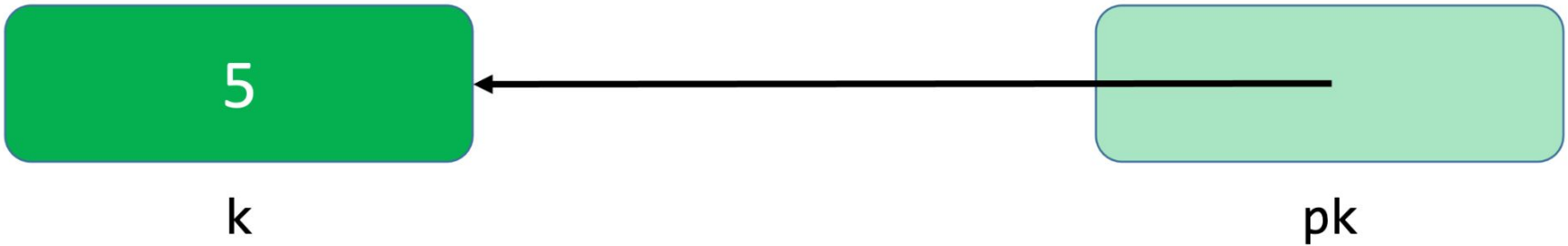
```
int n = 50;
```



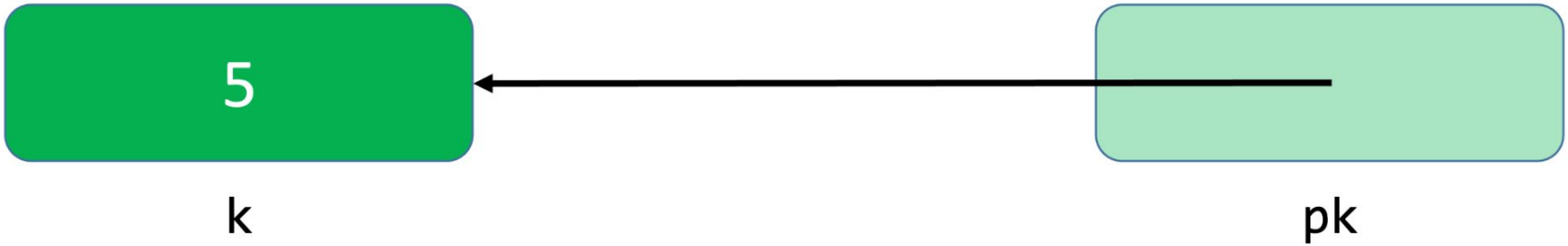
from Harvard CS50



from Harvard CS50

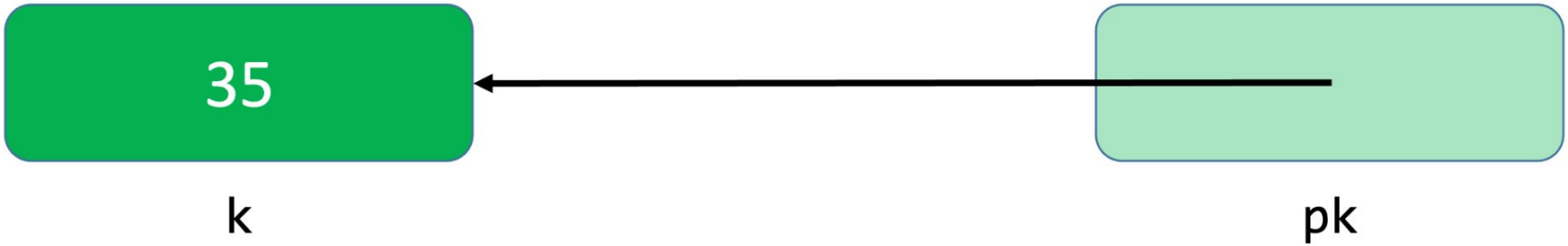


from Harvard CS50



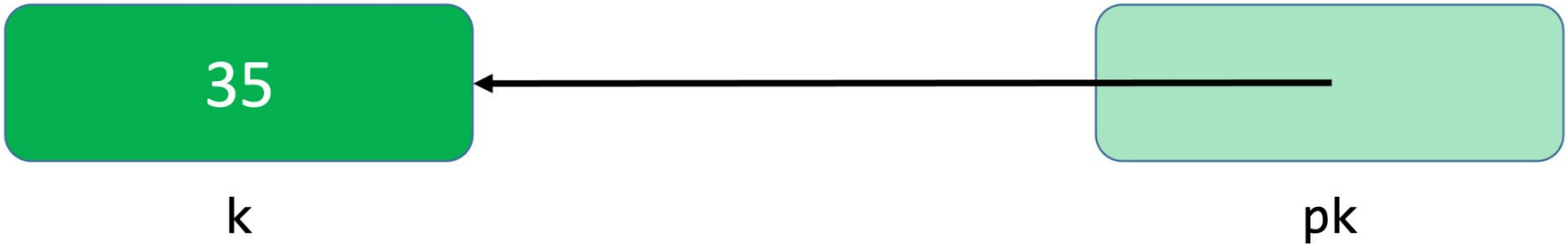

```
*pk = 35;
```

from Harvard CS50



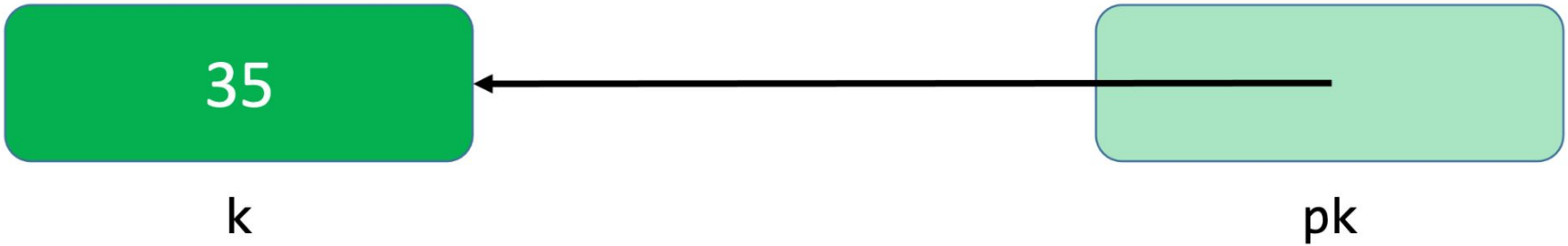

```
*pk = 35;
```

from Harvard CS50



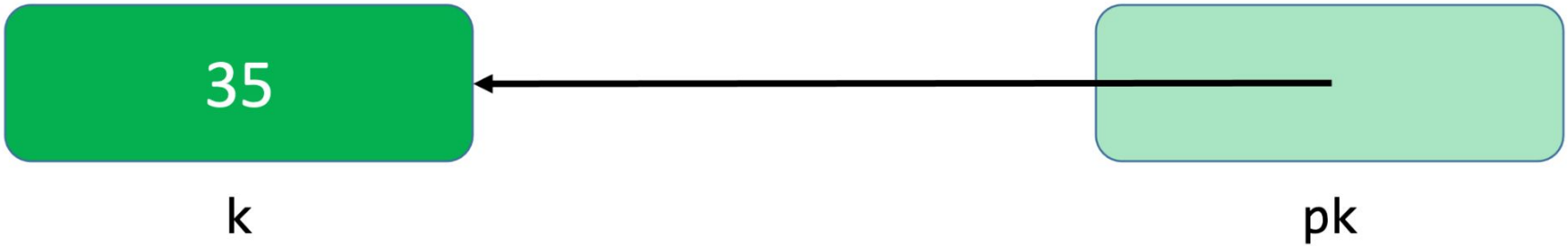

```
int m;
```

from Harvard CS50



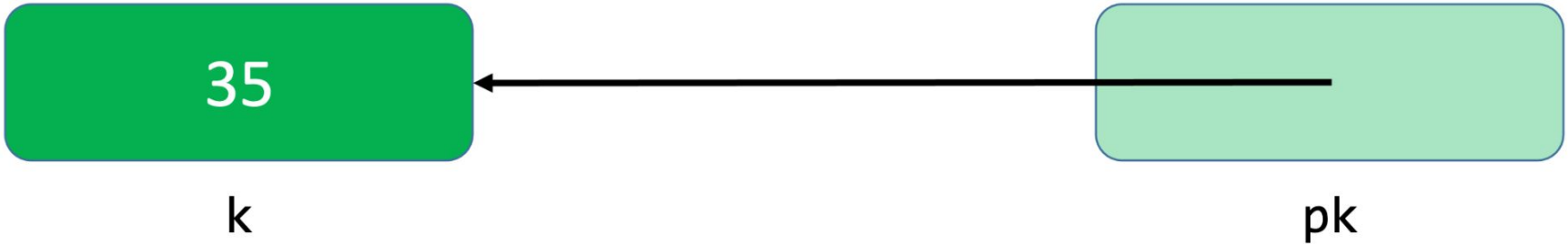

```
int m;
m = 4;
```

from Harvard CS50




```
int m;
m = 4;
```

from Harvard CS50

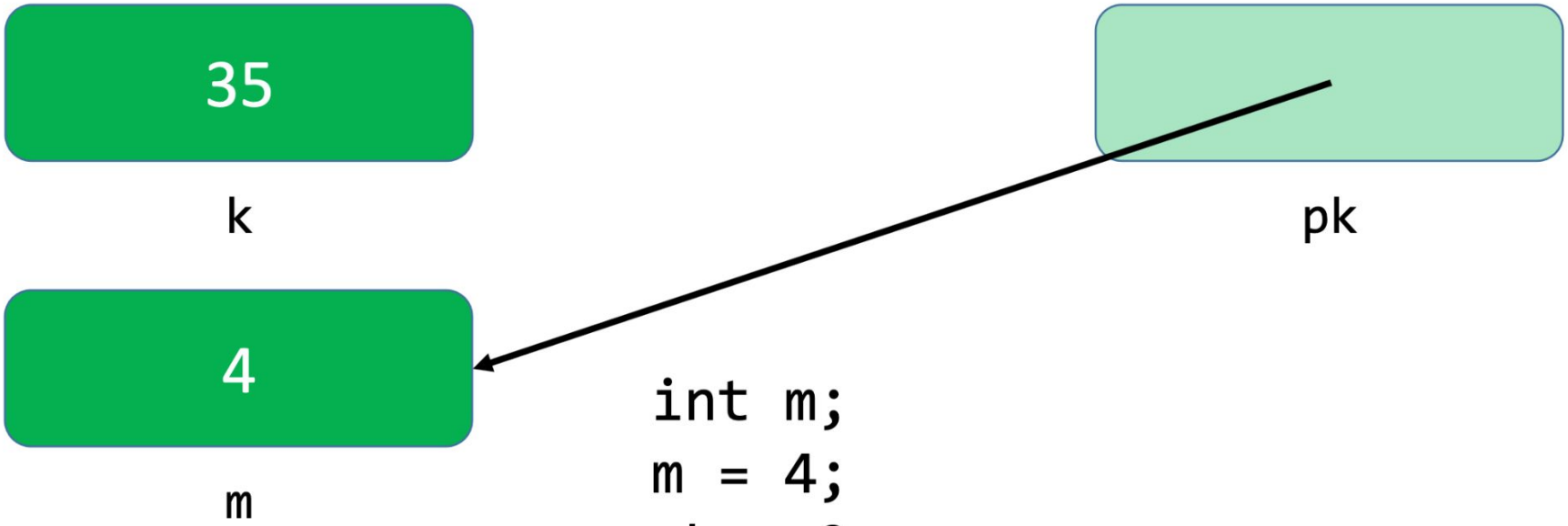


4

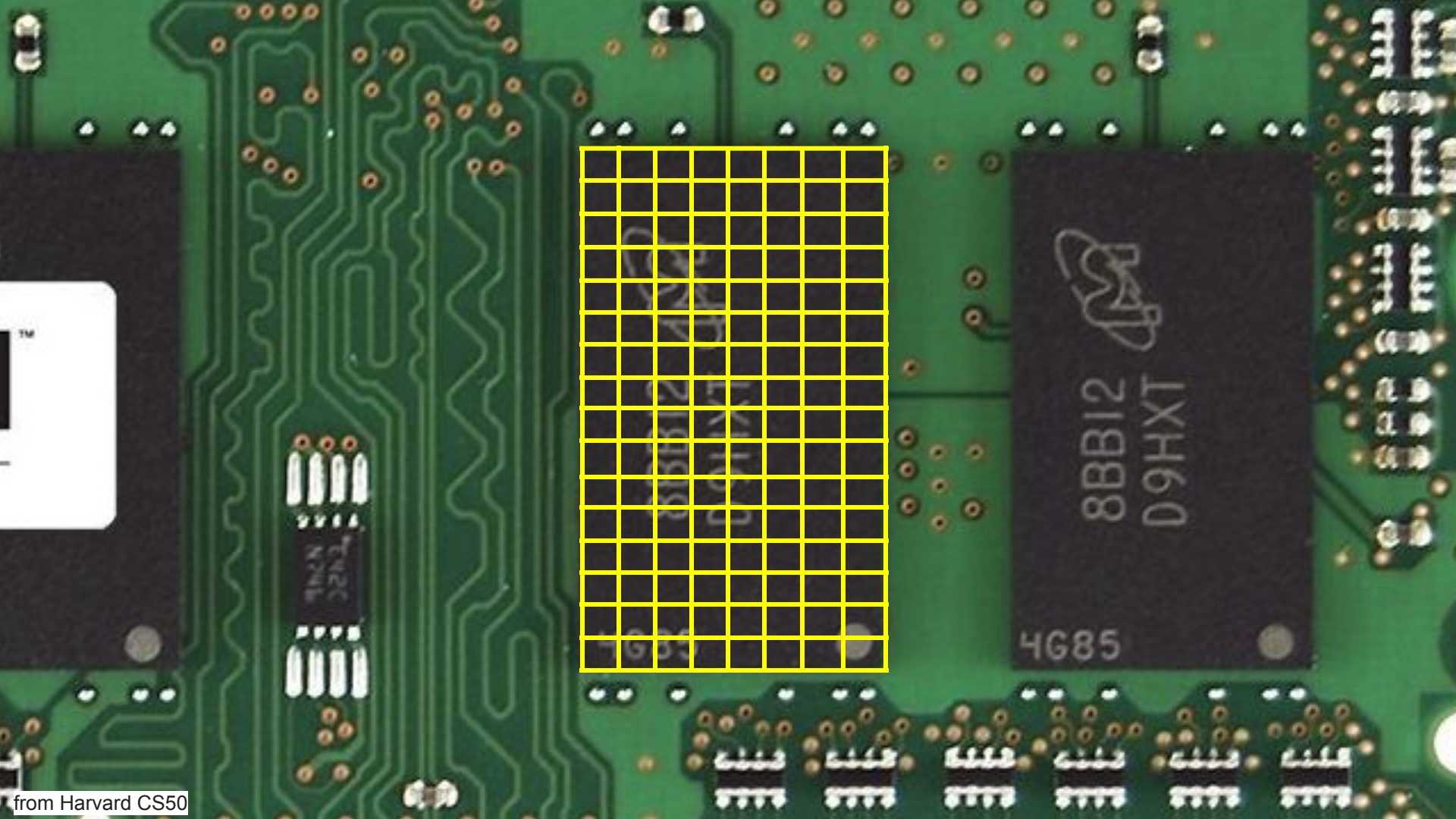
m

```
int m;
m = 4;
pk = &m;
```

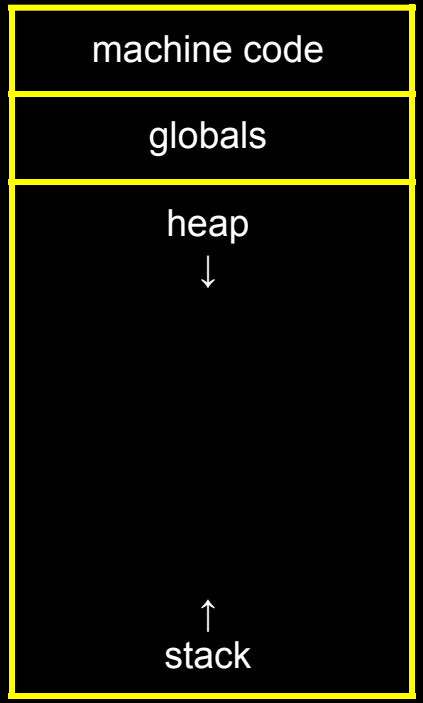
from Harvard CS50



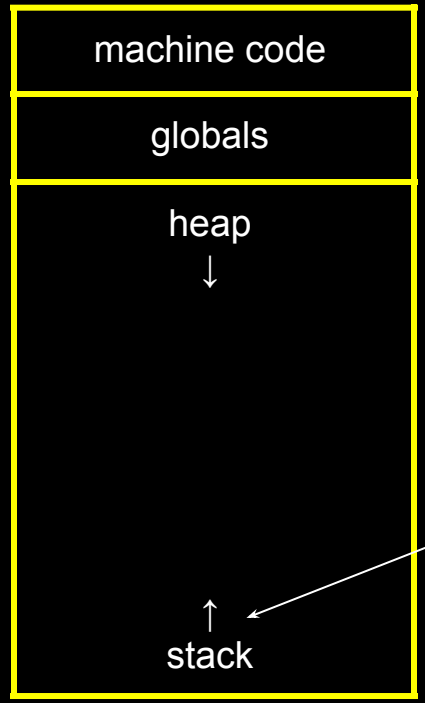
from Harvard CS50



from Harvard CS50



from Harvard CS50



new keyword

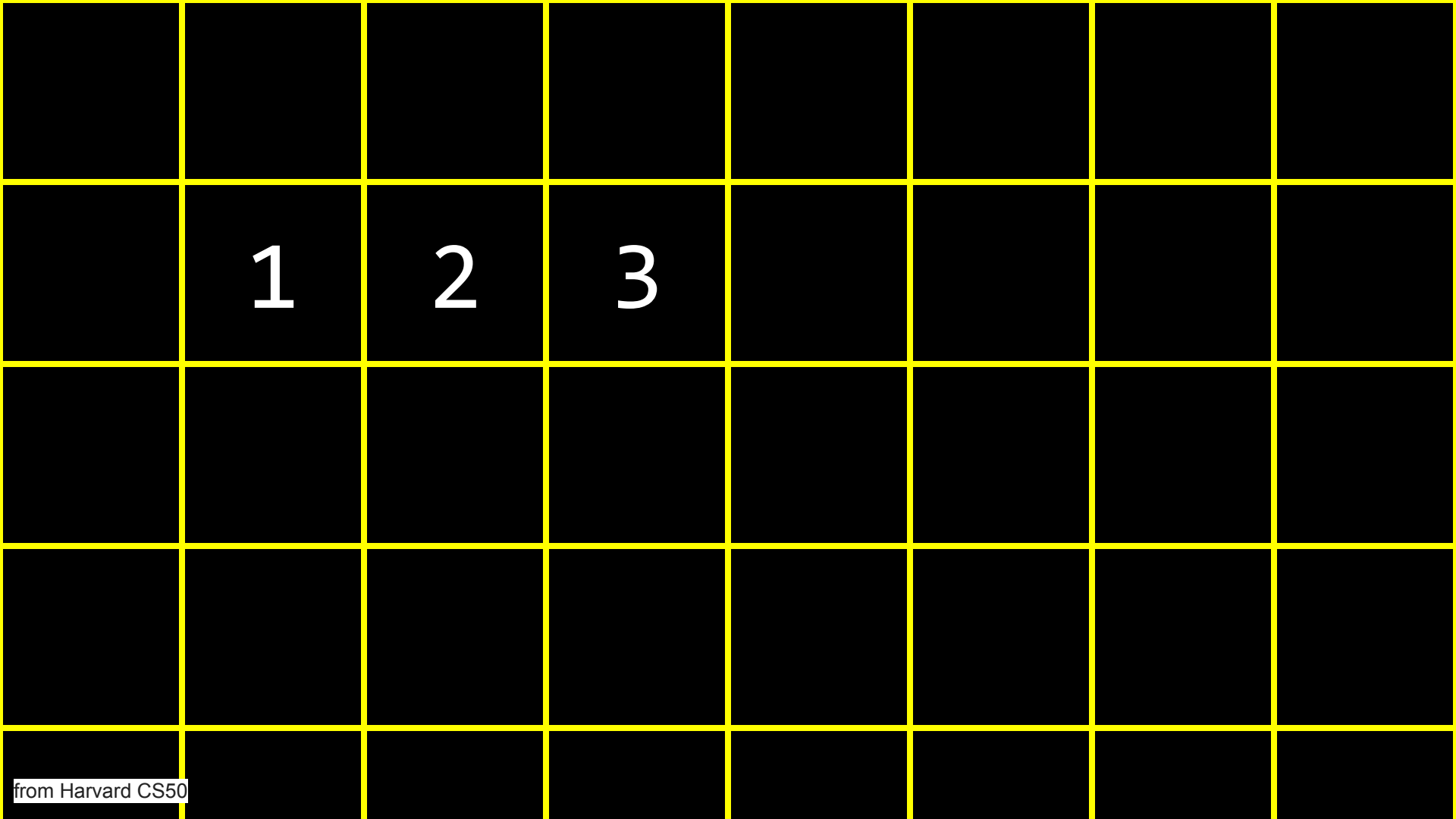
control: we get to allocate (and must free with **delete**)

local scope

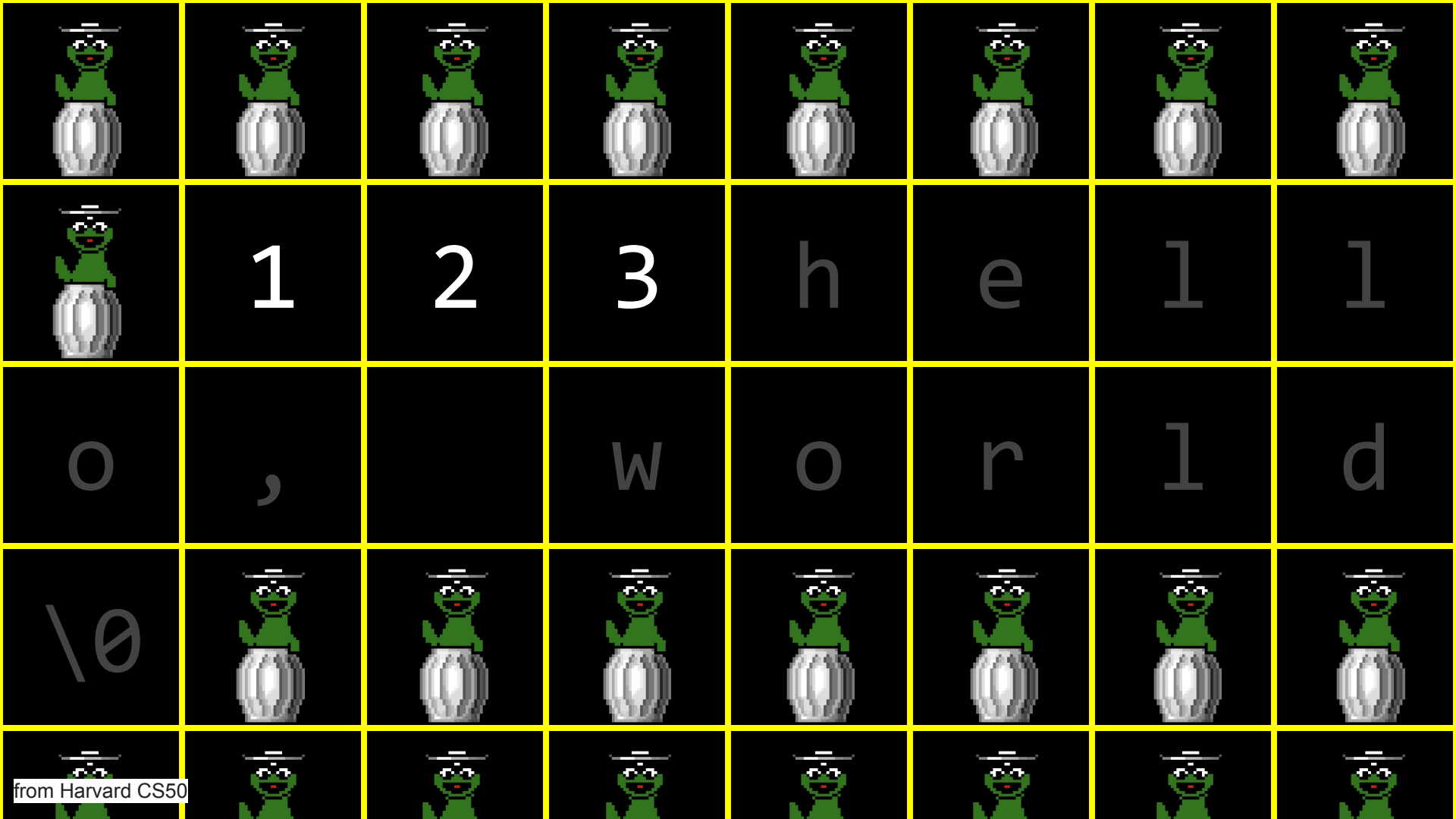
control: clears by itself, after every function

from Harvard CS50


```
x = new int;
```



from Harvard CS50



from Harvard CS50

1	2	3

from Harvard CS50

1	2	3

1			

from Harvard CS50

1	2	3

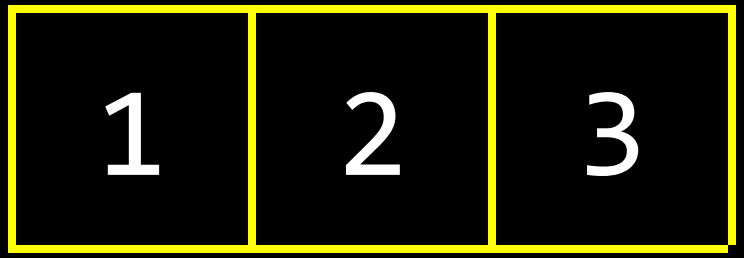
1	2		

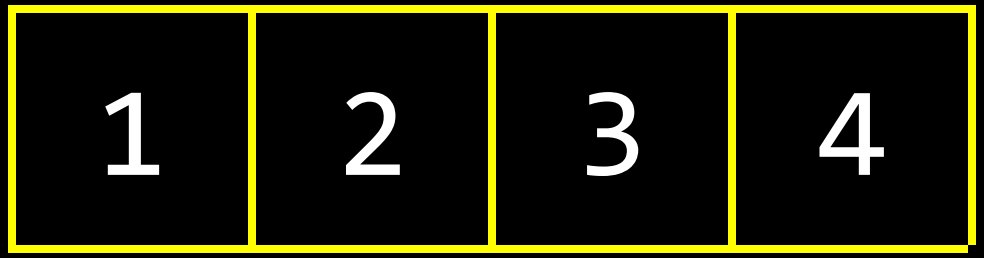
from Harvard CS50

1	2	3

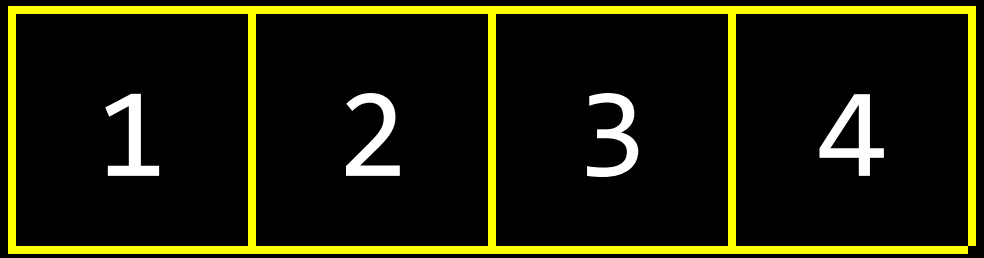
1	2	3	

from Harvard CS50





from Harvard CS50

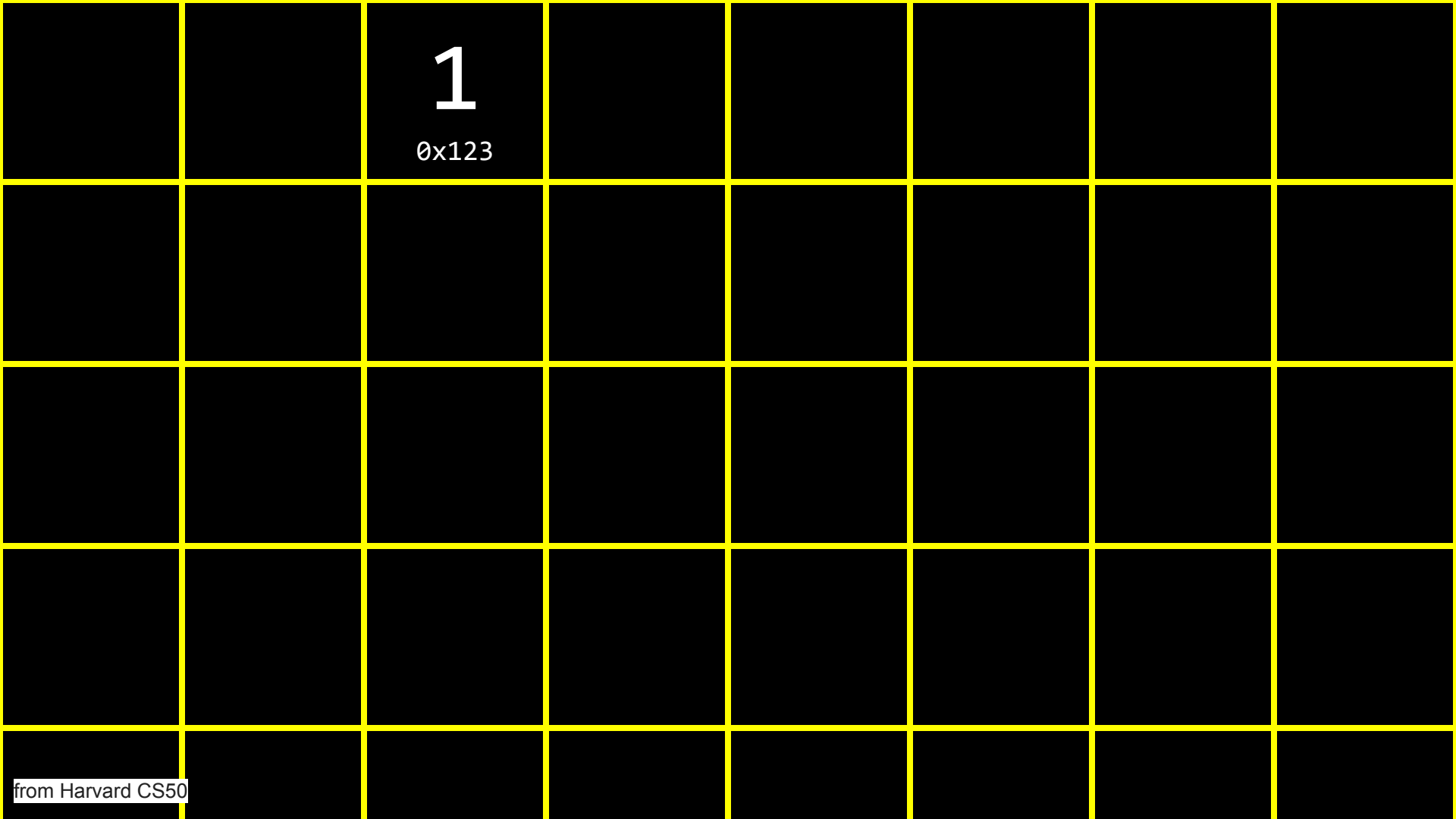


from Harvard CS50

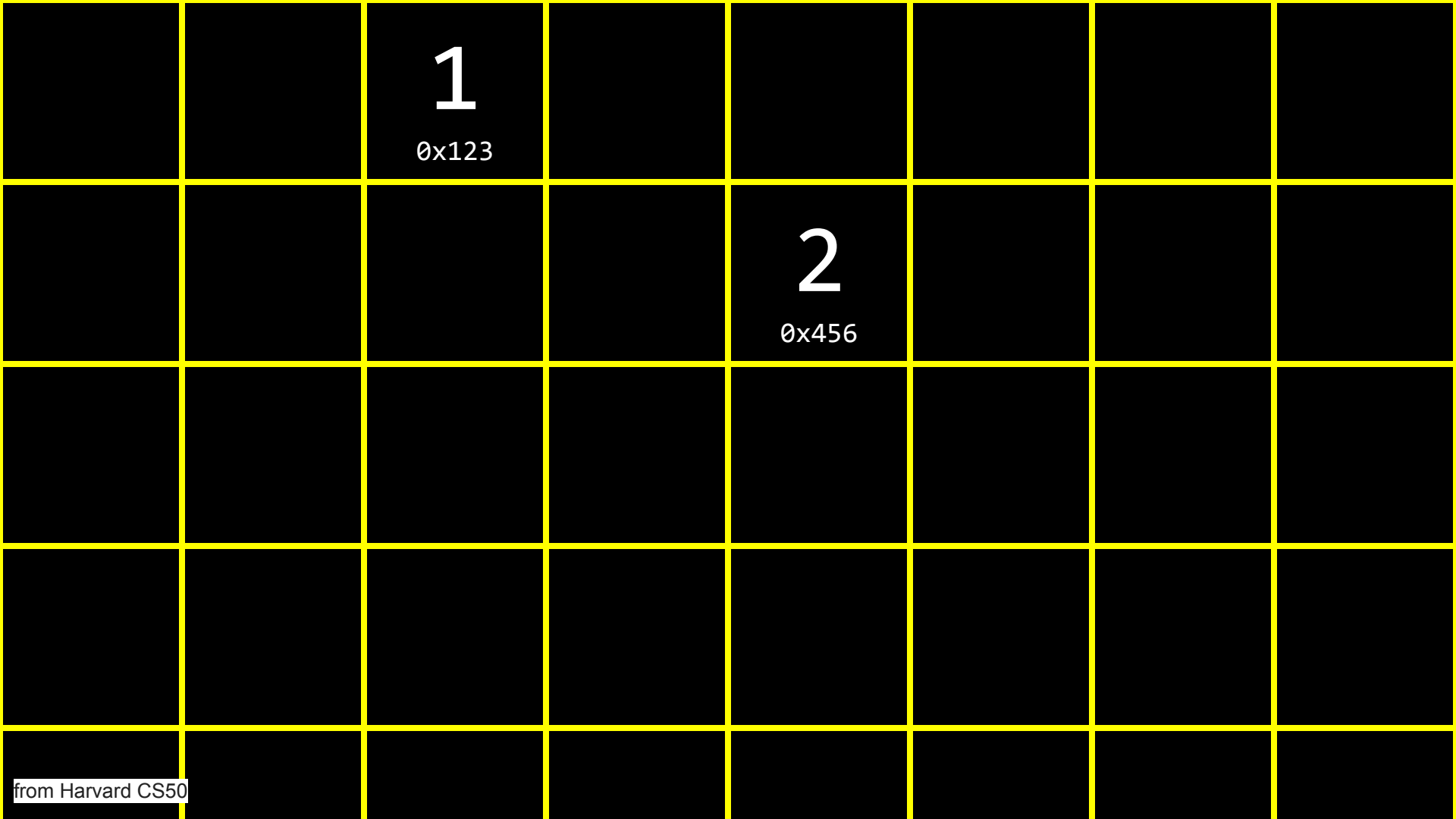
linked lists

from Harvard CS50

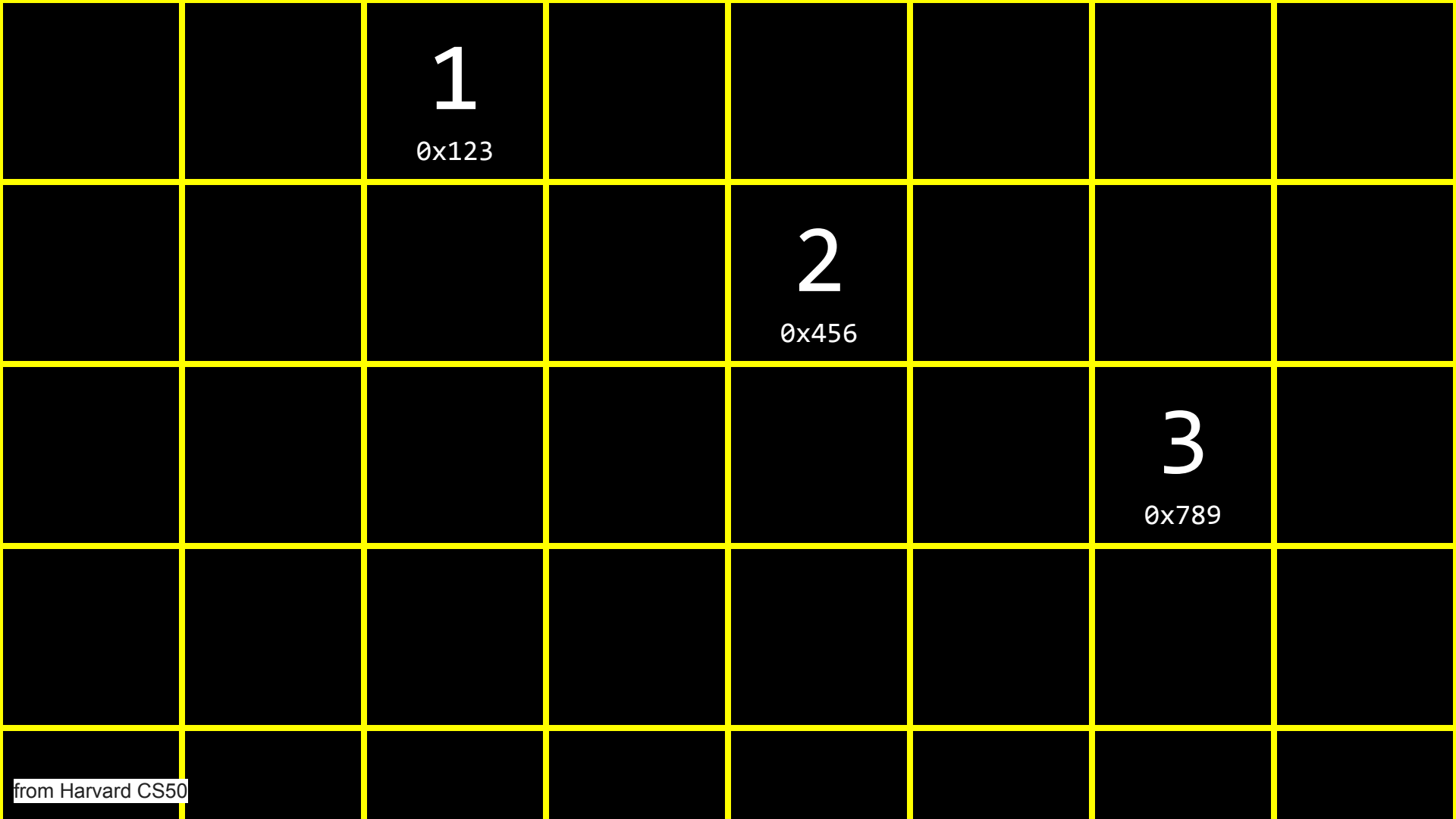
from Harvard CS50



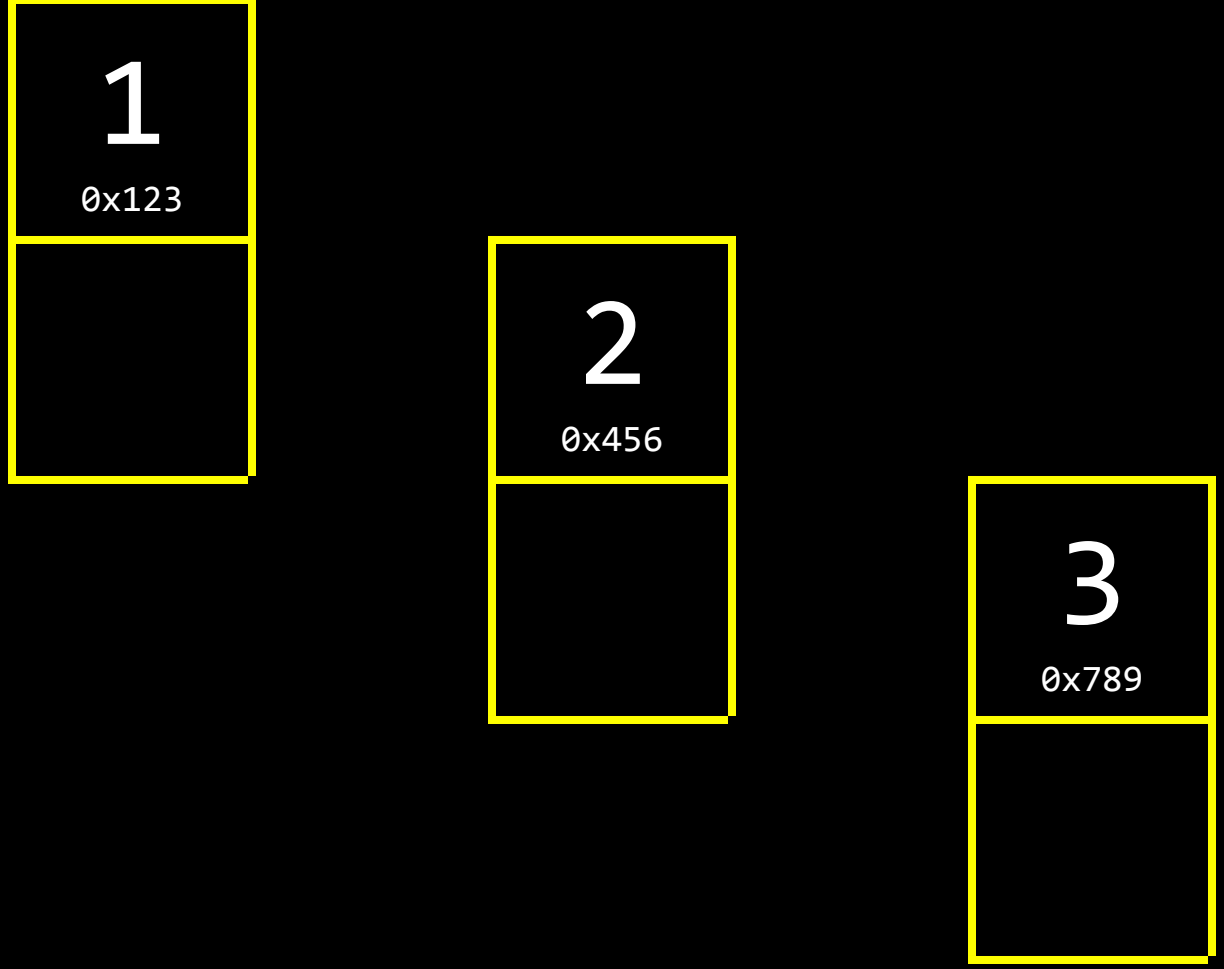
from Harvard CS50



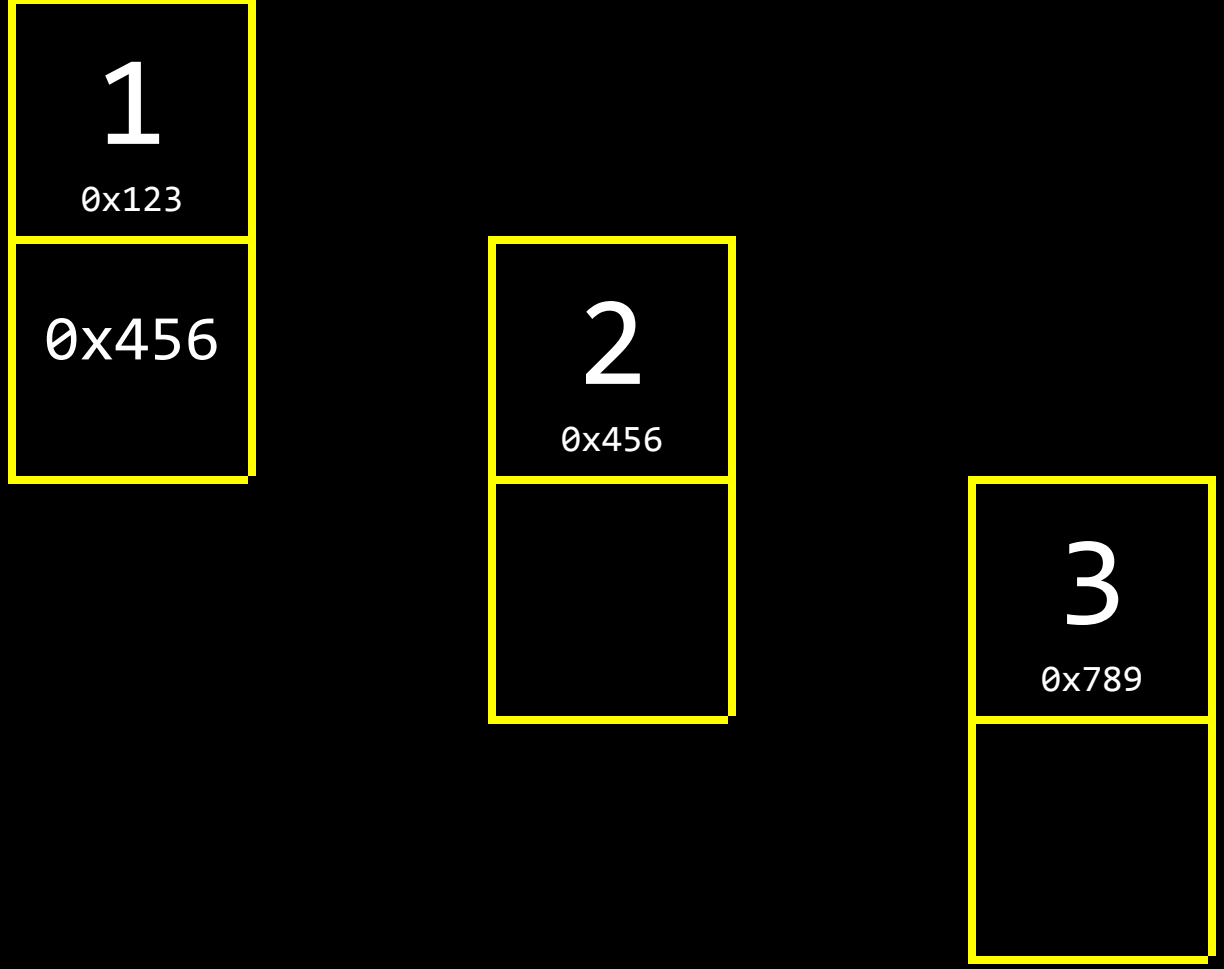
from Harvard CS50



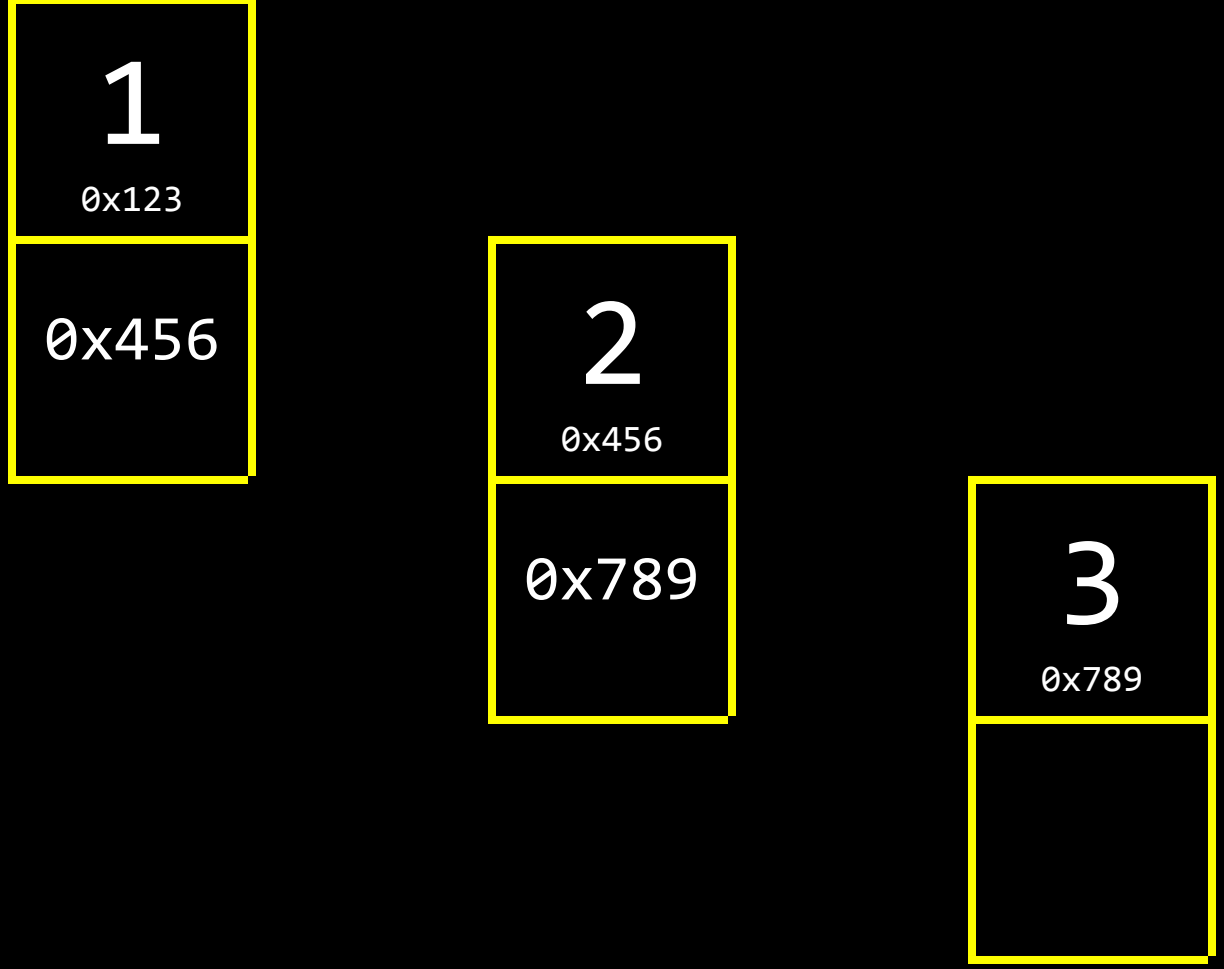
from Harvard CS50



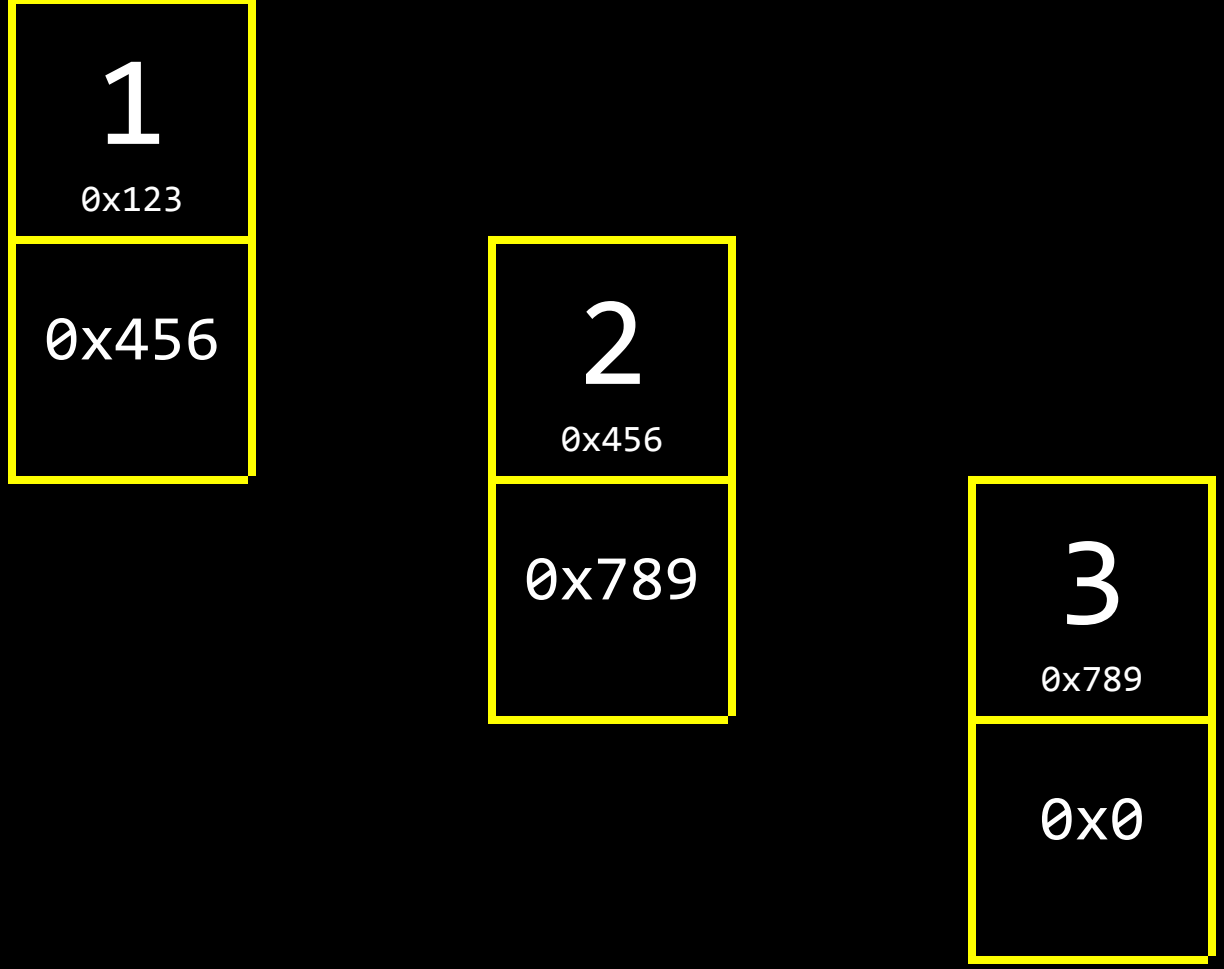
from Harvard CS50



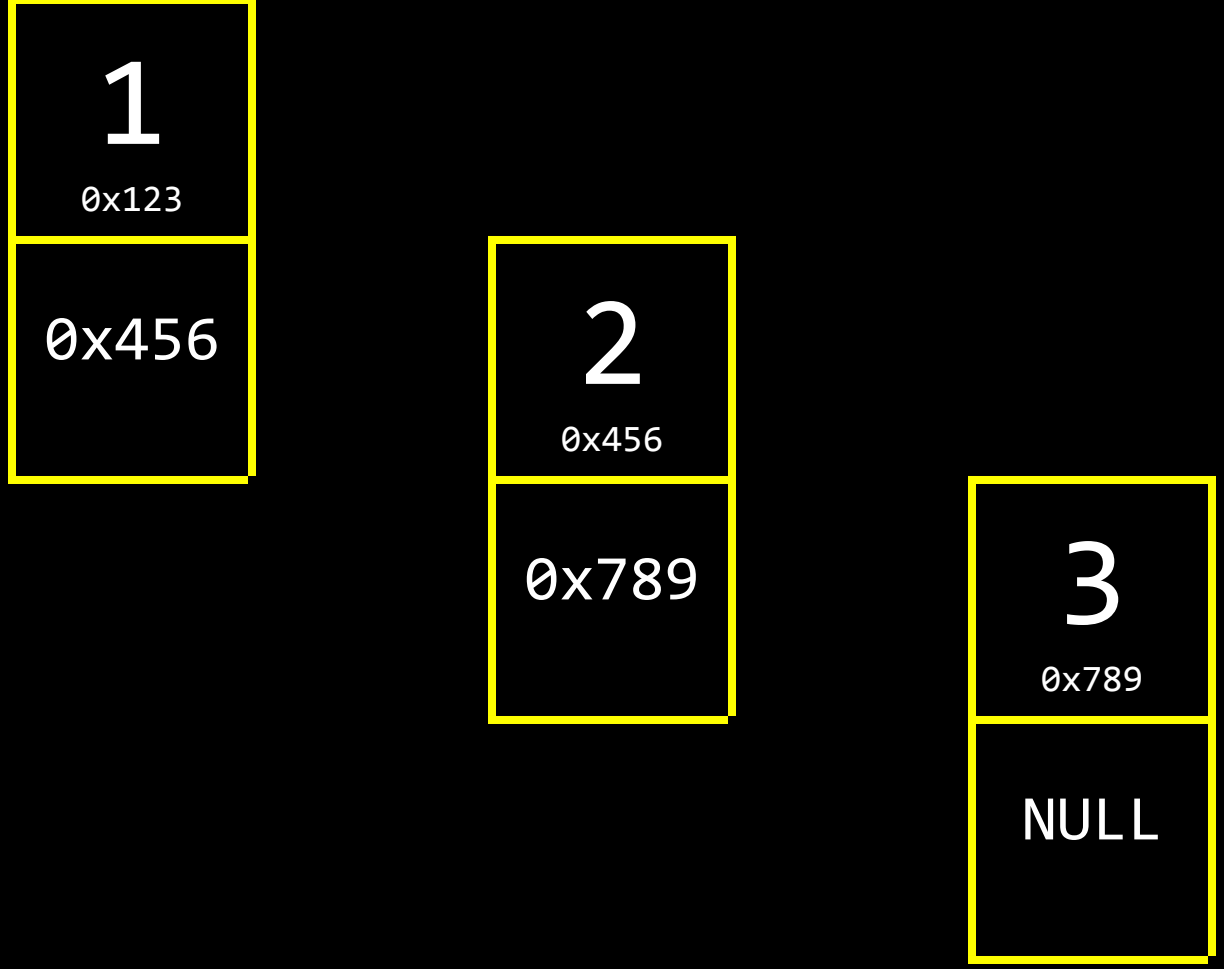
from Harvard CS50



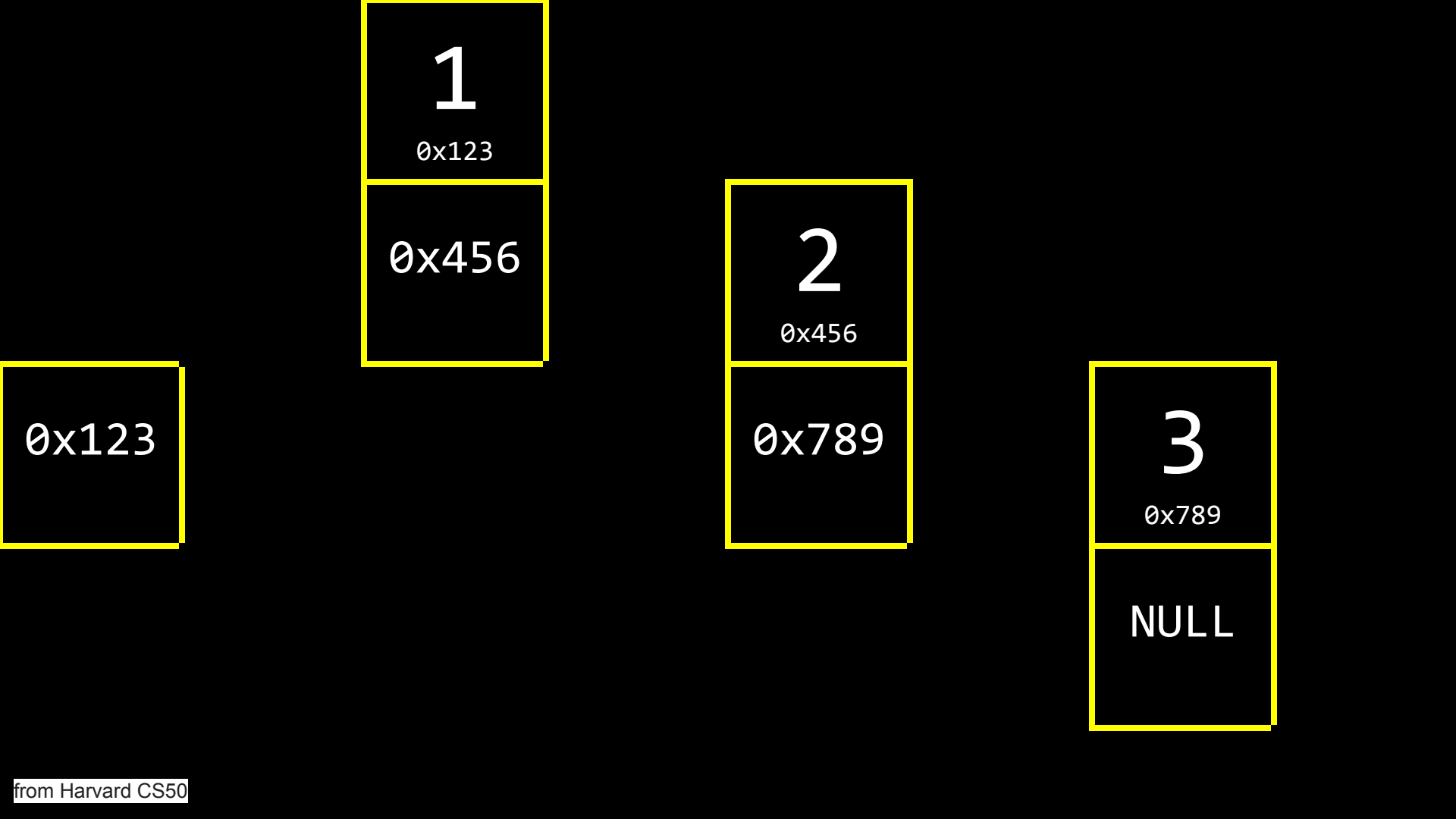
from Harvard CS50



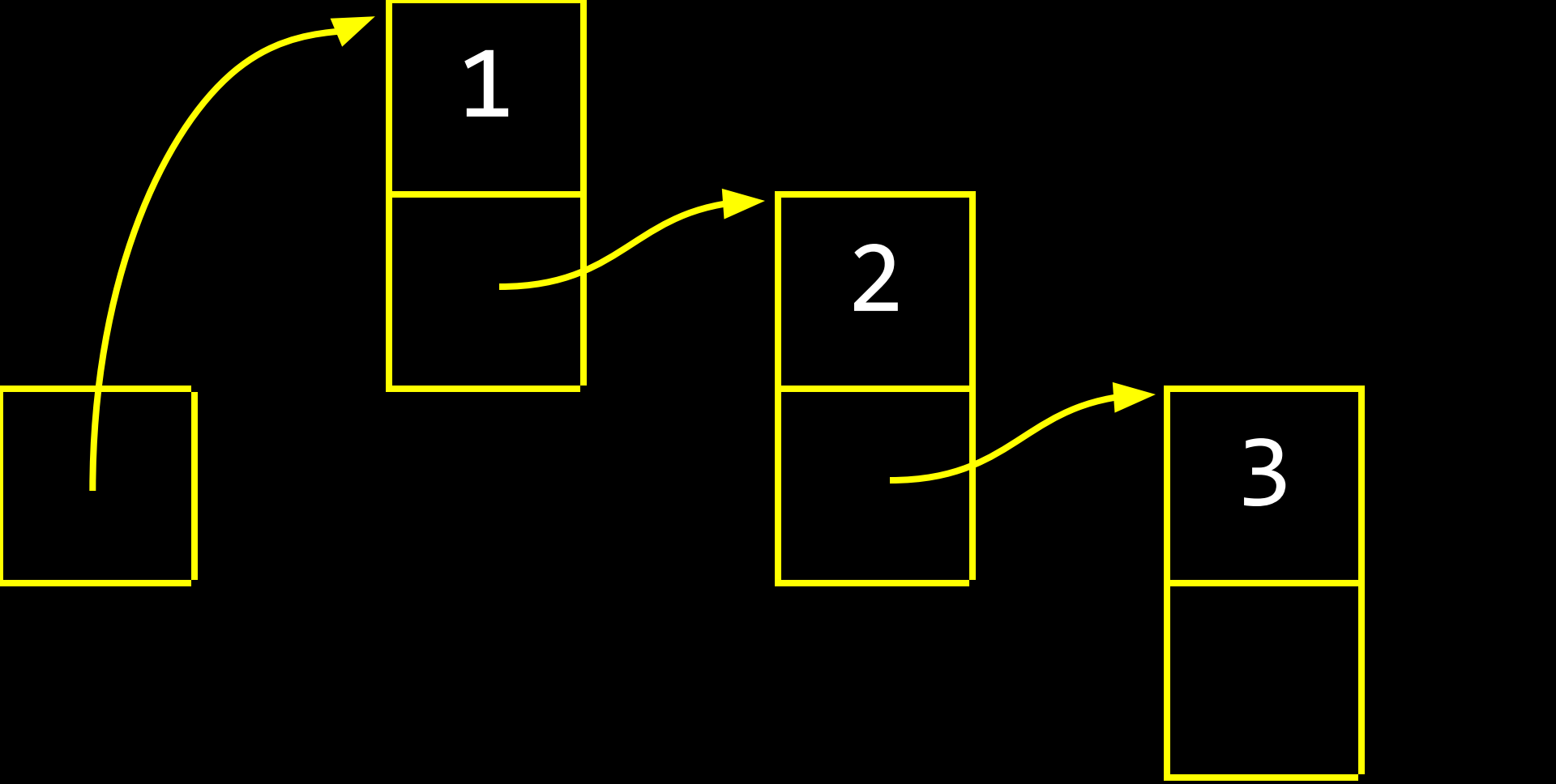
from Harvard CS50



from Harvard CS50



from Harvard CS50



from Harvard CS50