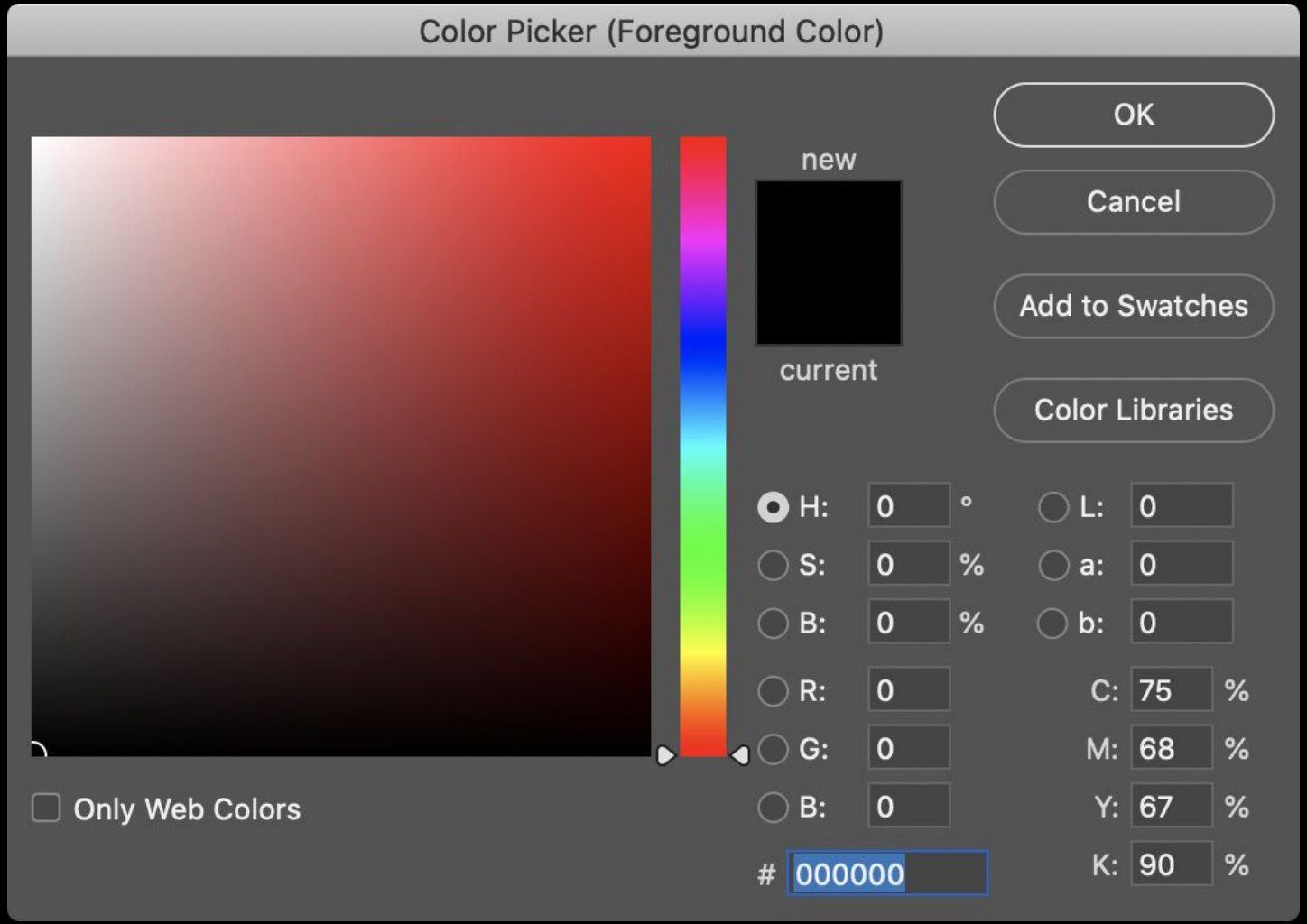
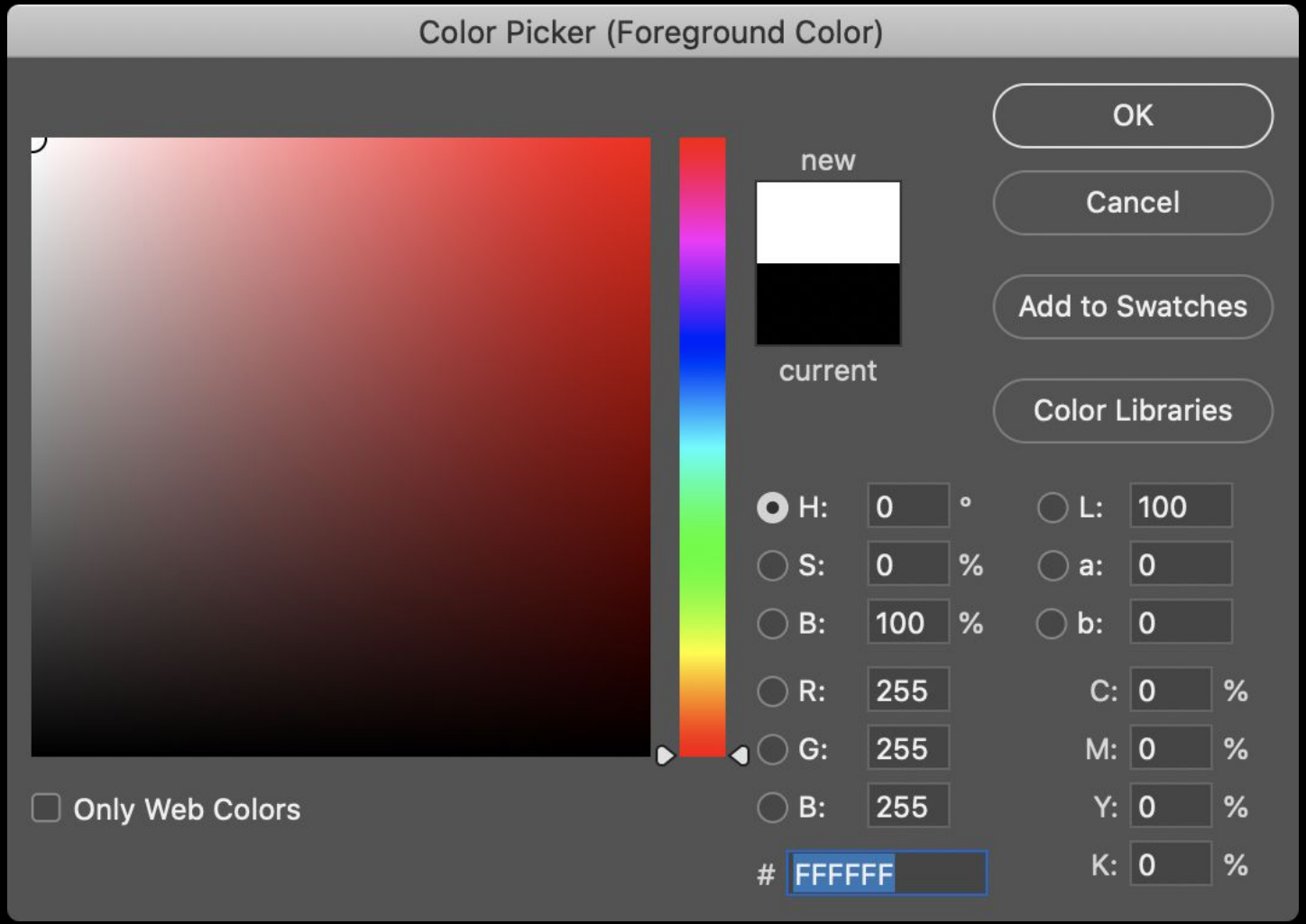
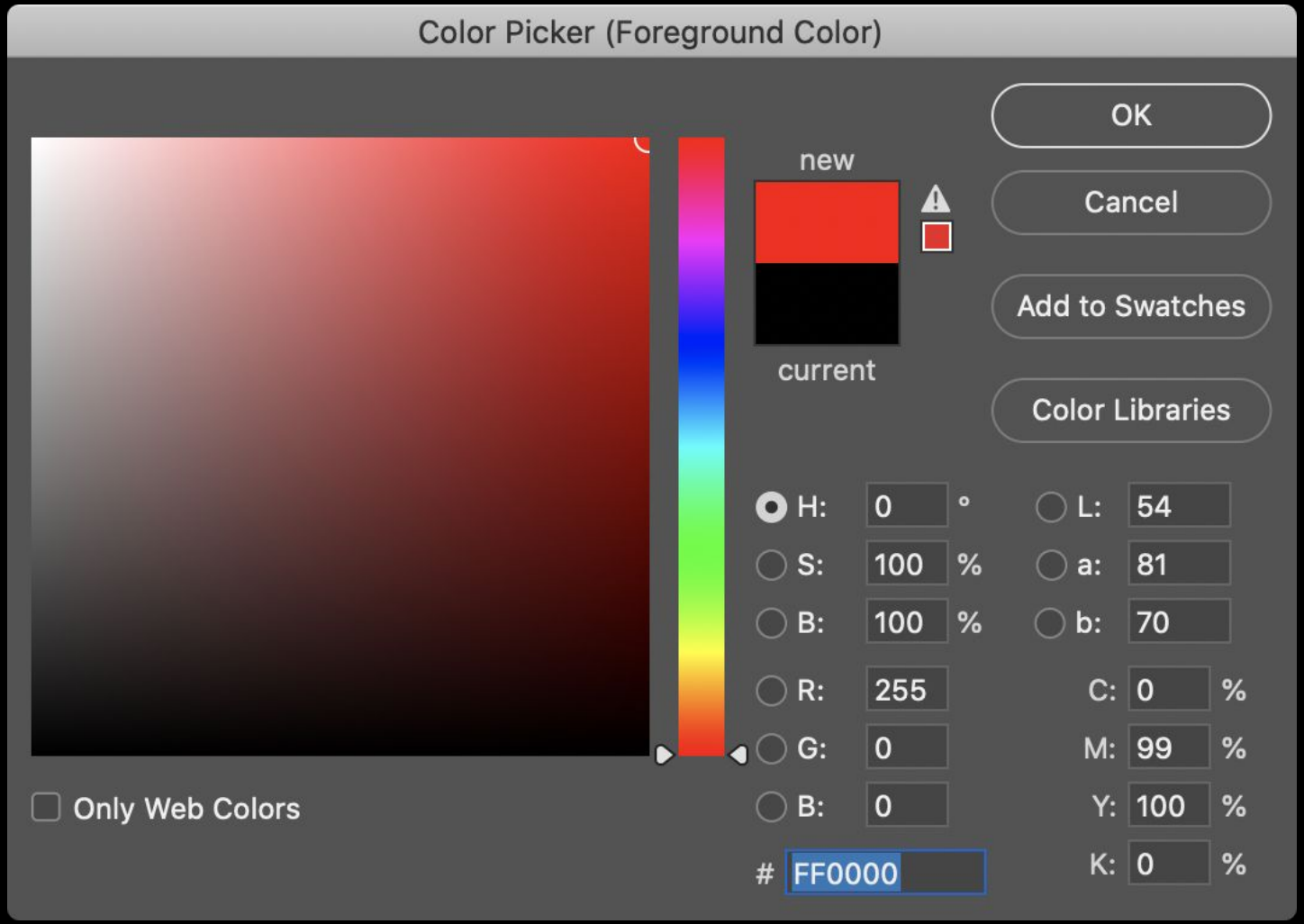
This is CS50

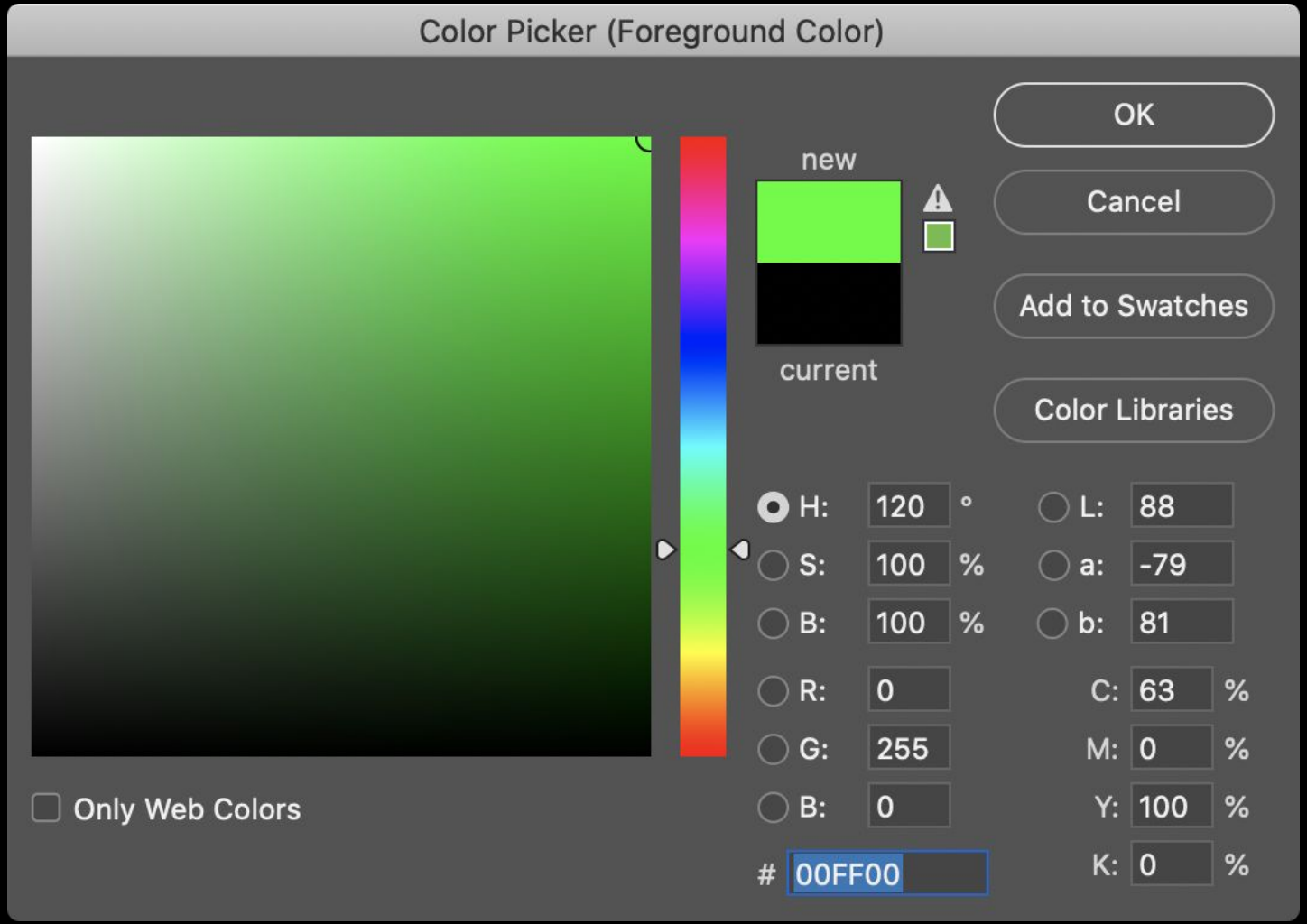

HARVARD

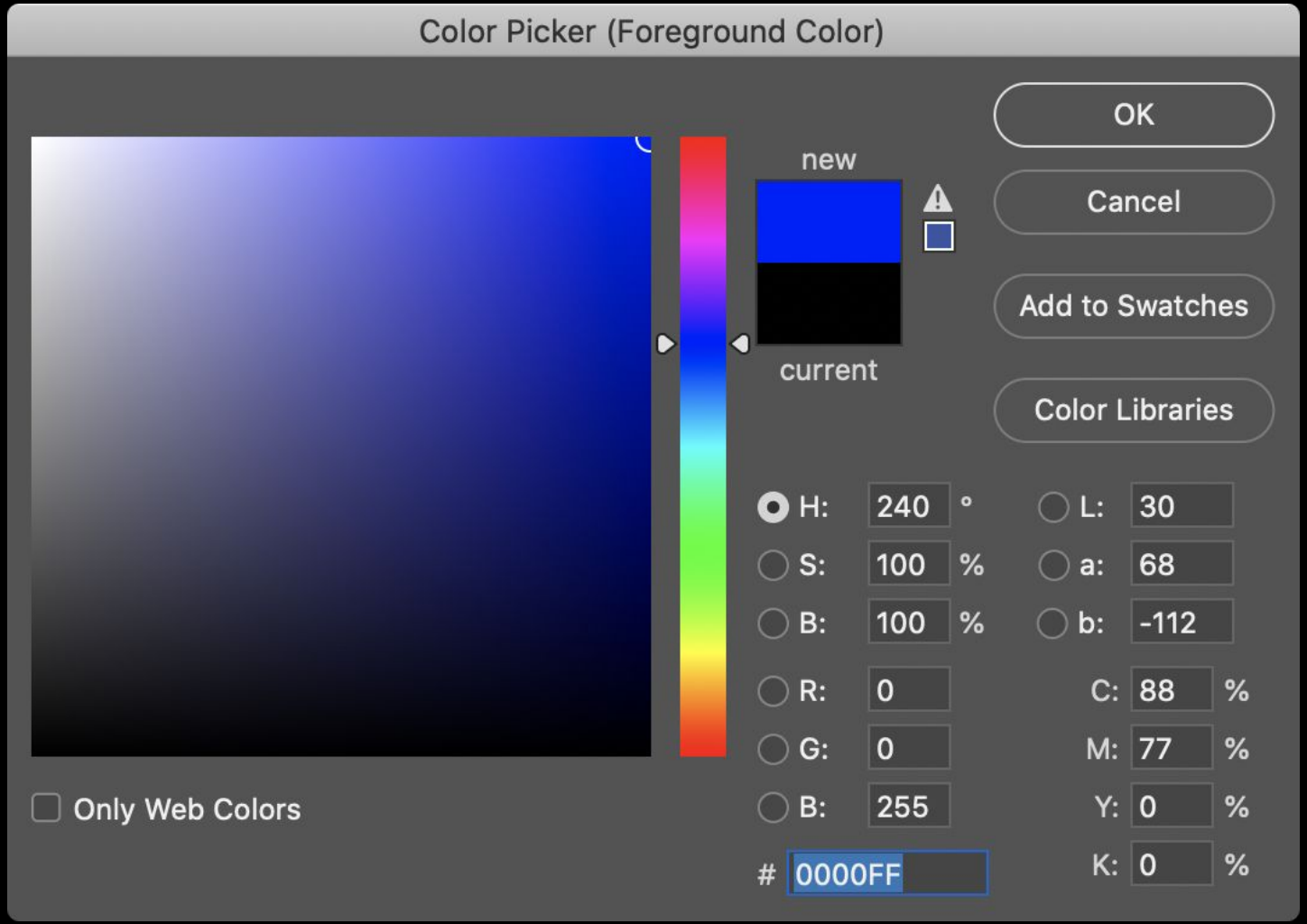
cs50.ly/art


Color Picker (Foreground Color)
OK
Cancel
Add to Swatches
Color Libraries
new
current
H: 0 °
S: 0 %
B: 0 %
R: 0
G: 0
B: 0
L: 0
a: 0
b: 0
C: 75 %
M: 68 %
Y: 67 %
K: 90 %
000000
Only Web Colors

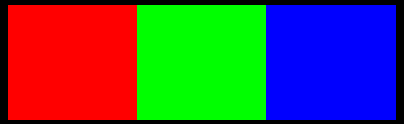

Color Picker (Foreground Color)
OK
Cancel
Add to Swatches
Color Libraries
new
current
H: 0 °
S: 0 %
B: 100 %
R: 255
G: 255
B: 255
FFFFFF
L: 100
a: 0
b: 0
C: 0 %
M: 0 %
Y: 0 %
K: 0 %
Only Web Colors


Color Picker (Foreground Color)
OK
Cancel
Add to Swatches
Color Libraries
new
current
H: 0 °
S: 100 %
B: 100 %
R: 255
G: 0
B: 0
FF0000
L: 54
a: 81
b: 70
C: 0 %
M: 99 %
Y: 100 %
K: 0 %
Only Web Colors

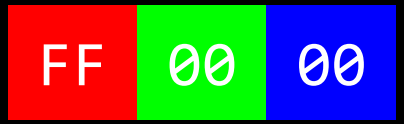

Color Picker (Foreground Color)
OK
Cancel
Add to Swatches
Color Libraries
new
current
H: 120 °
S: 100 %
B: 100 %
R: 0
G: 255
B: 0
00FF00
L: 88
a: -79
b: 81
C: 63 %
M: 0 %
Y: 100 %
K: 0 %
Only Web Colors

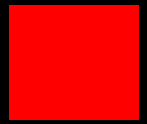

Color Picker (Foreground Color)
OK
Cancel
Add to Swatches
Color Libraries
new
current
H: 240 °
S: 100 %
B: 100 %
R: 0
G: 0
B: 255
0000FF
L: 30
a: 68
b: -112
C: 88 %
M: 77 %
Y: 0 %
K: 0 %
Only Web Colors

RGB

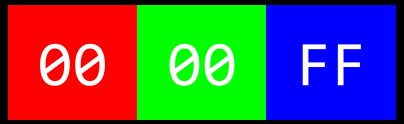


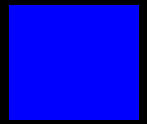

72
73
33


FF
00
00




00
FF
00


00
00
FF




FF
FF
FF

0 1

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9 A B C D E F

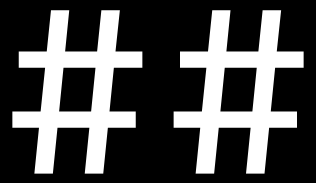
base-16

hexadecimal


16^1
16^0

# #

16 1



16 1

00

16 1

01

16 1

02

16 1

03

16 1

04

16 1

05

16 1

06

16 1

07

16 1

08

16 1

09

16 1

0A

16 1

0B

16 1

0C

16 1

0D

16 1

0E

16 1

0F

16 1

10

16 1

11

16 1

12

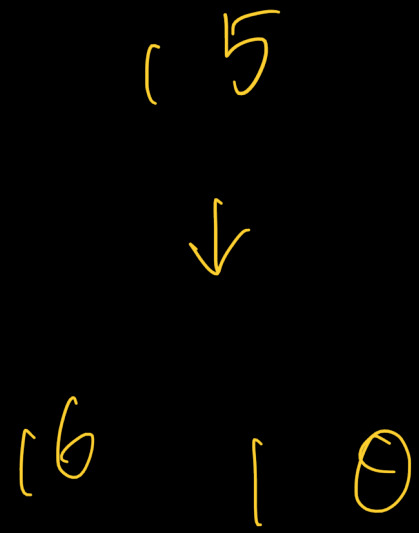
16 1

13

16 1

14

16 1



16 1

FF

16 1

F F

16 × F + 1 × F

largest number

16 1

F F

16 × 15 + 1 × 15

16 1

F F

240 + 15

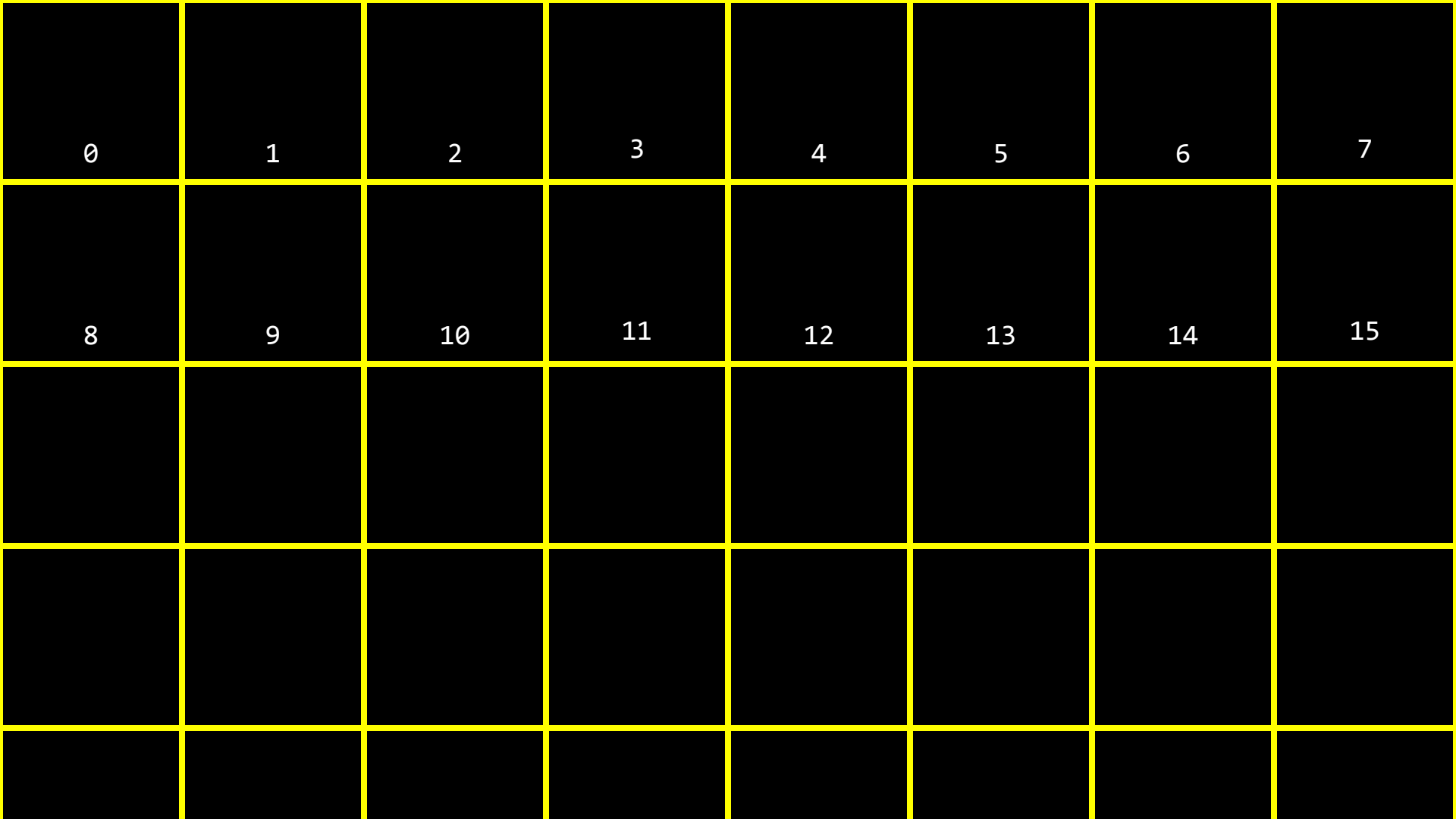
16 1

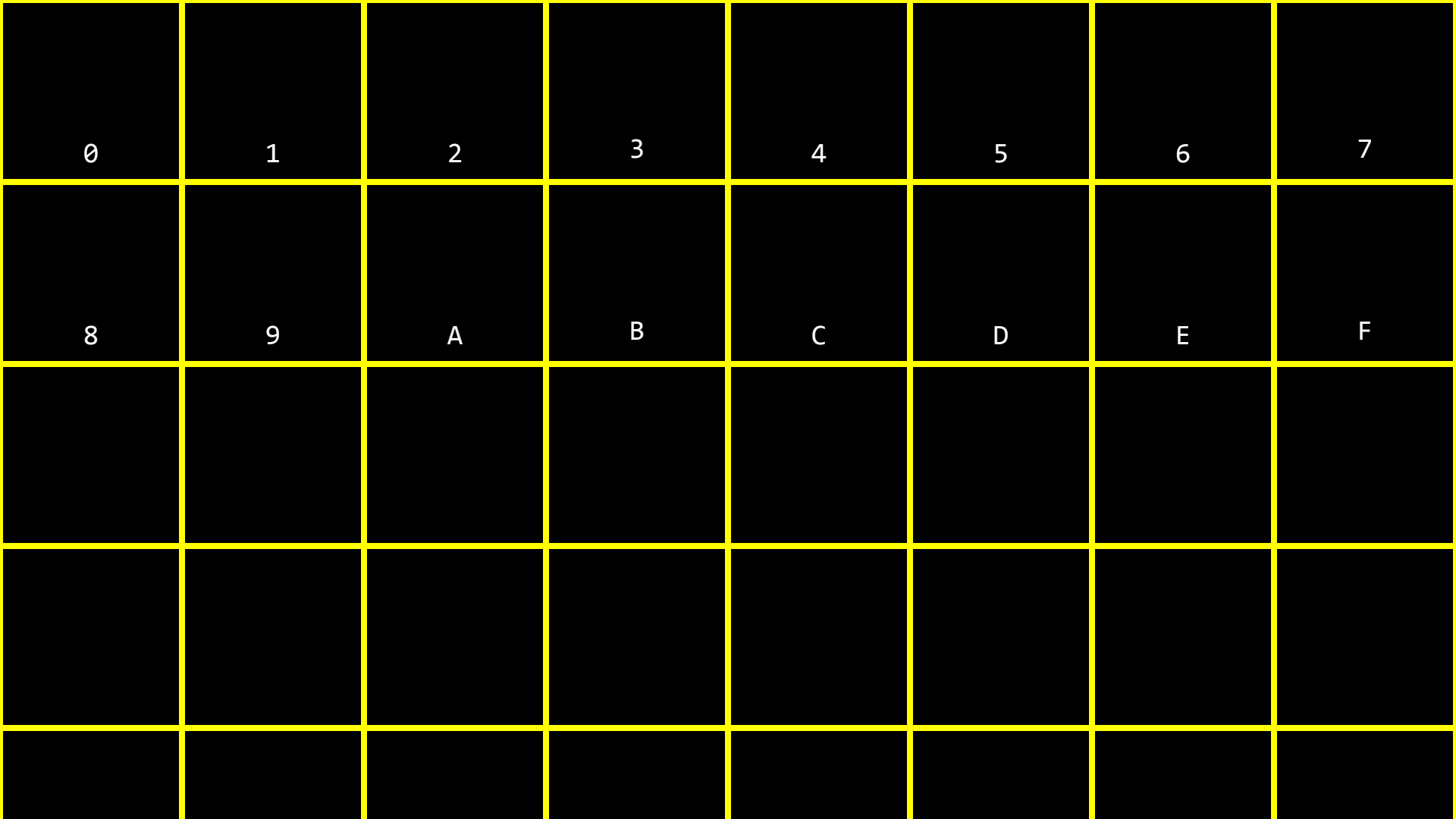
255

128 64 32 16 8 4 2 1

11111111

255


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

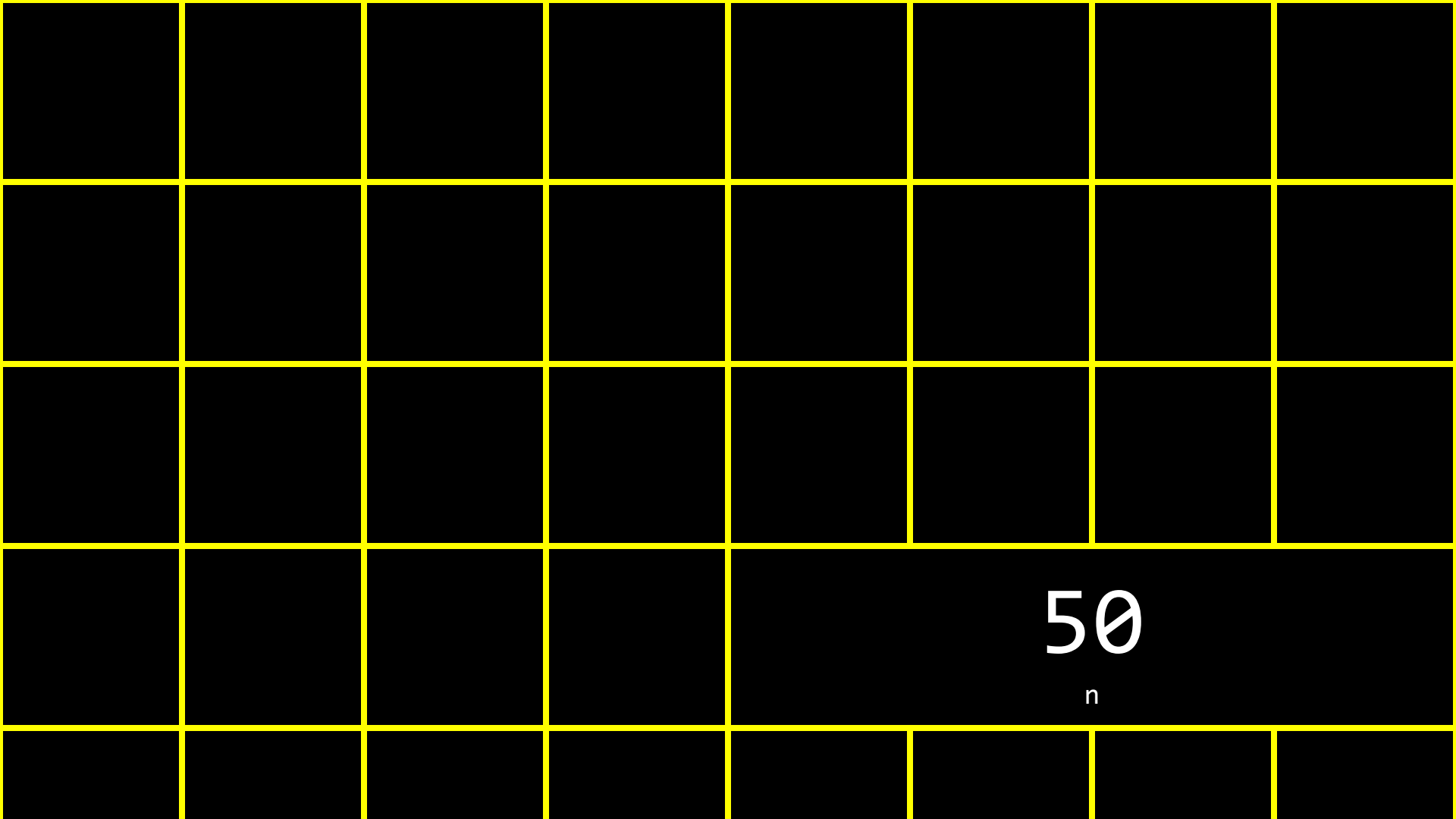

0
1
2
3
4
5
6
7
8
9
A
B
C
D
E
F

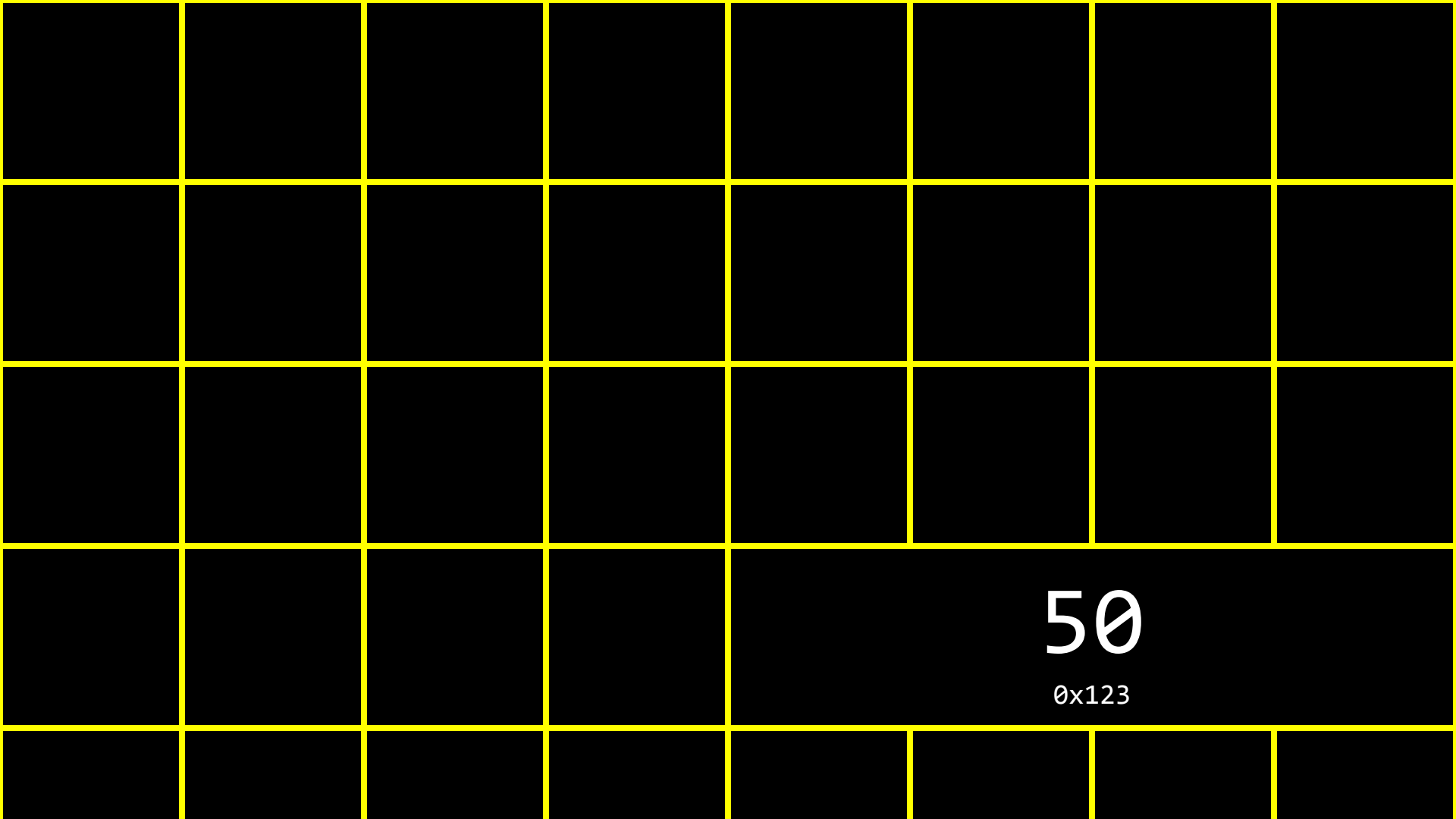
0	1	2	3	4	5	6	7
8	9	A	B	C	D	E	F
10	11	12	13	14	15	16	17
18	19	1A	1B	1C	1D	1E	1F

HEXIDEXIMAL

0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17
0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F

```
int n = 50;
```


50
n


50
0x123

pointers

```
int n = 50;
int *p = &n;
```

the address of the operator

pointer

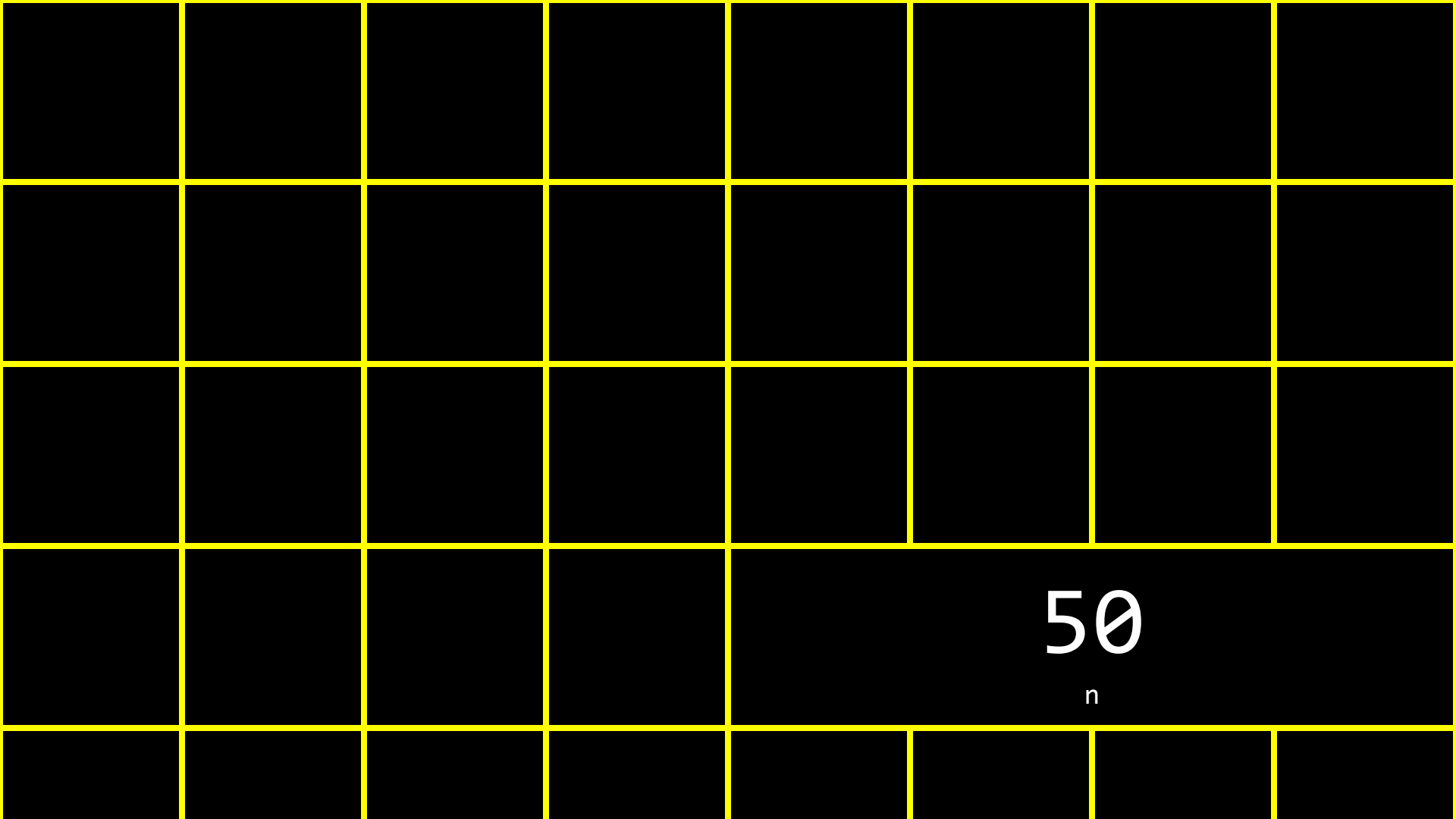
is the address of something

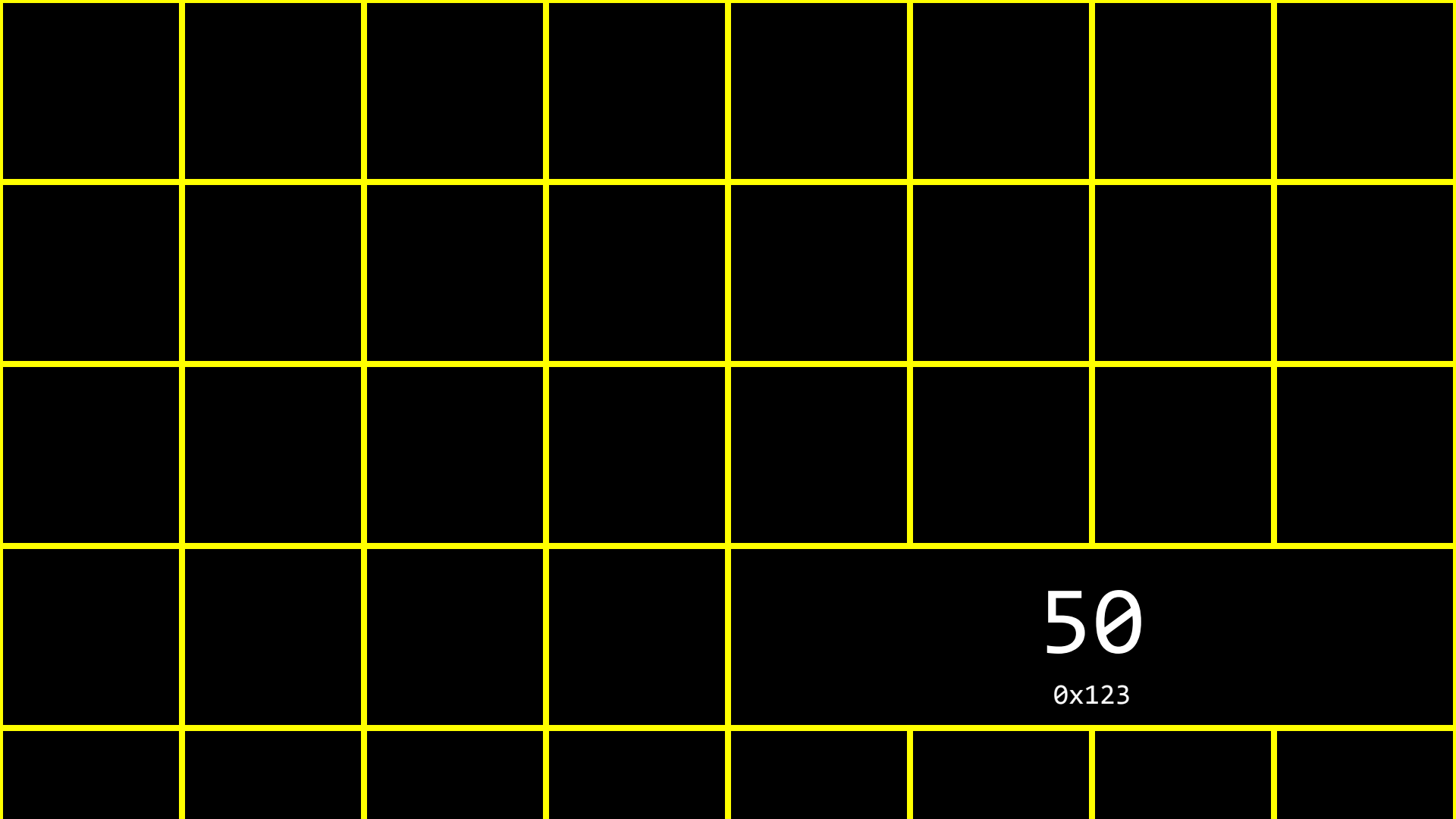
Where the memory in the pc, shall be a random.

```
printf("%p\n", p)
```

to print a pointer

⇒ `printf("%p\n", &n)`


50
n


50
0x123

0x123

p

50

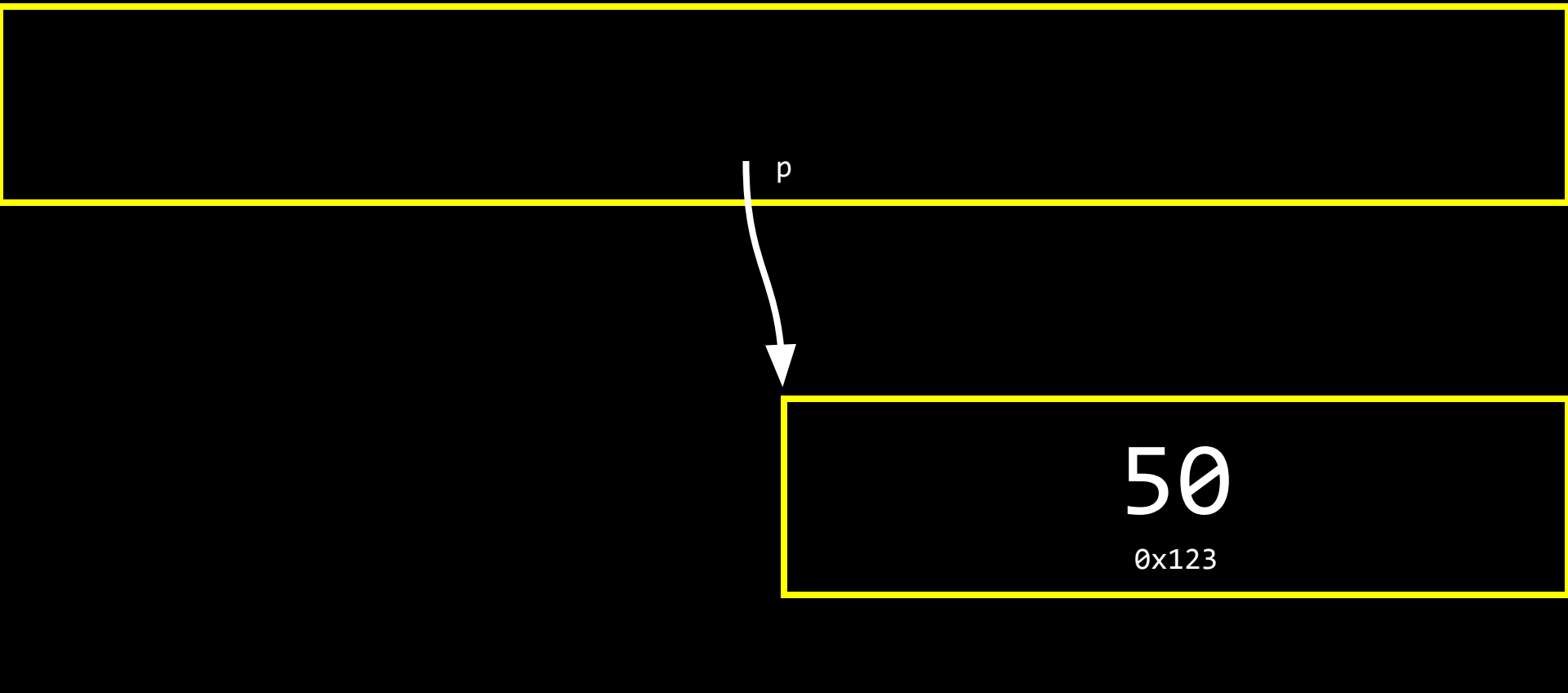
0x123

0x123

p

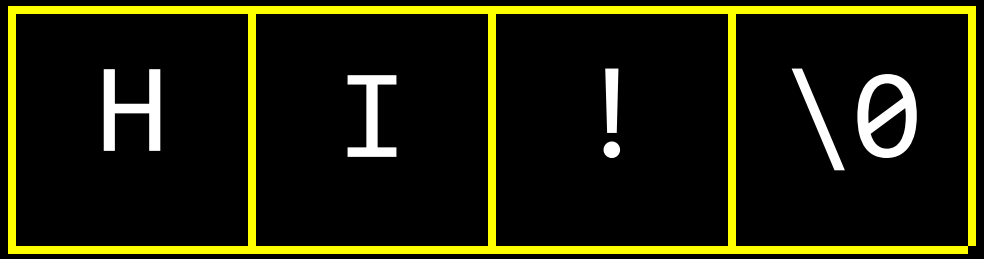
50

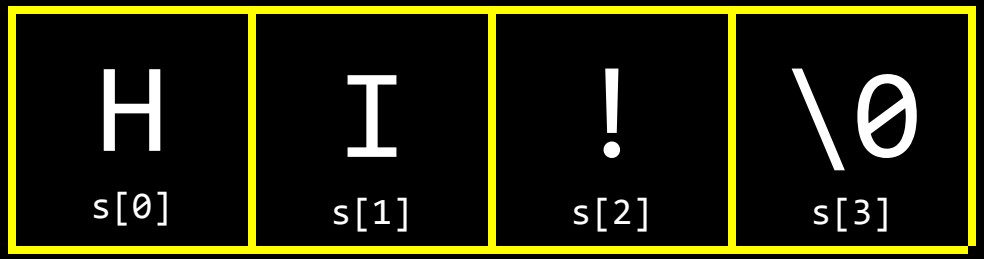
0x123

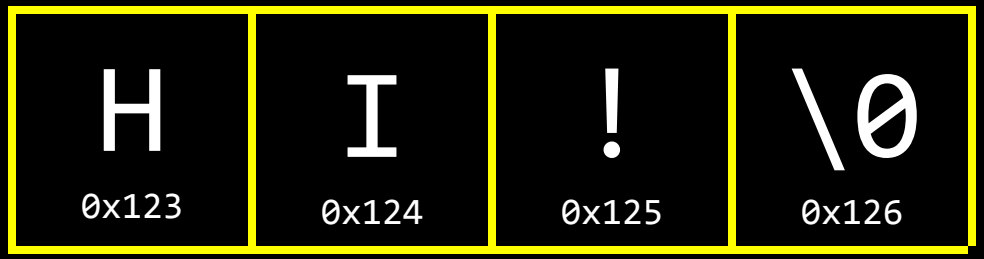

p
50
0x123

string

```
string s = "HI!";
```

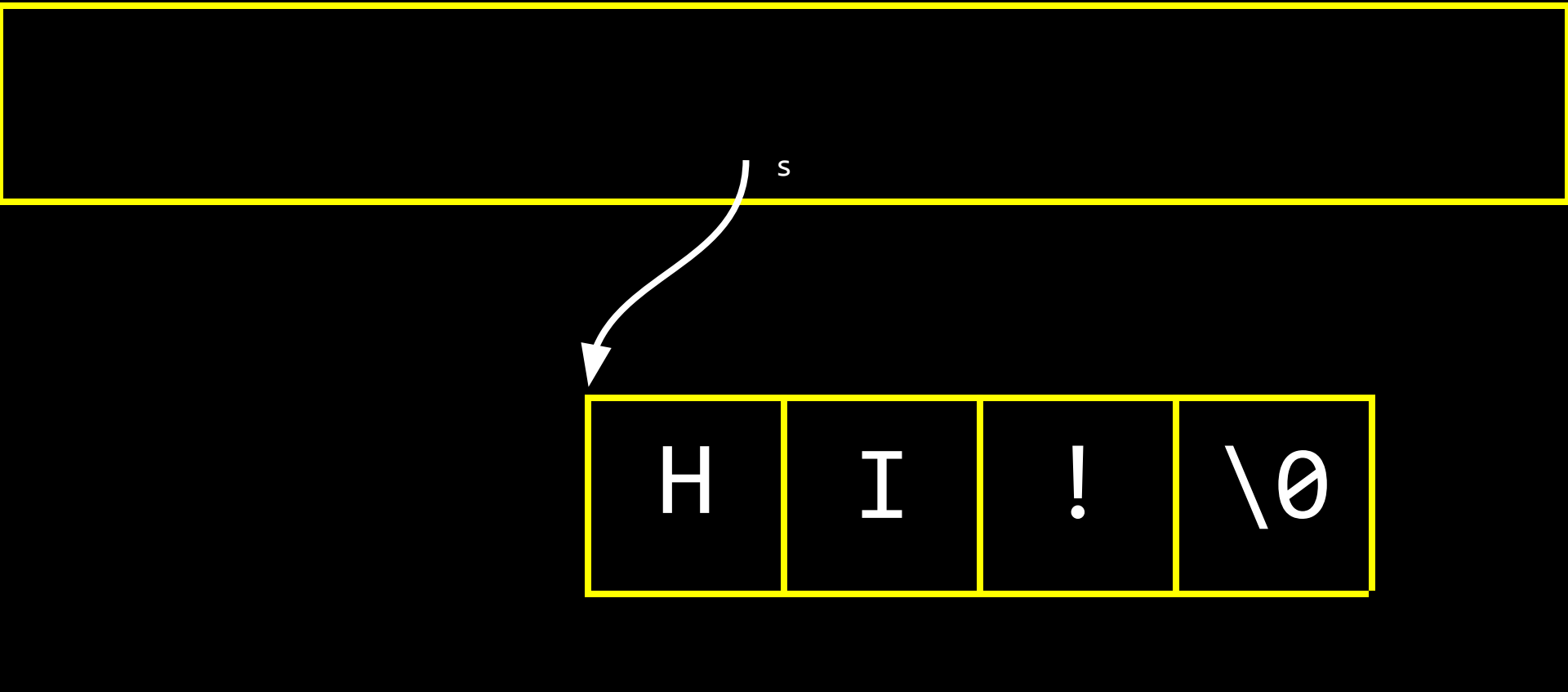

H
I
!
\0


H
s[0]
I
s[1]
!
s[2]
\0
s[3]


H
0x123
I
0x124
!
0x125
\0
0x126

0x123

s


s
H
I
!
\0

```
string s = "HI!";
```

```
char *s = "HI!";
```

```
char *s = "HI!";
```

```
typedef struct
{
    string name;
    string number;
}
person;
```

```
typedef struct
{
    string name;
    string number;
}
person;
```

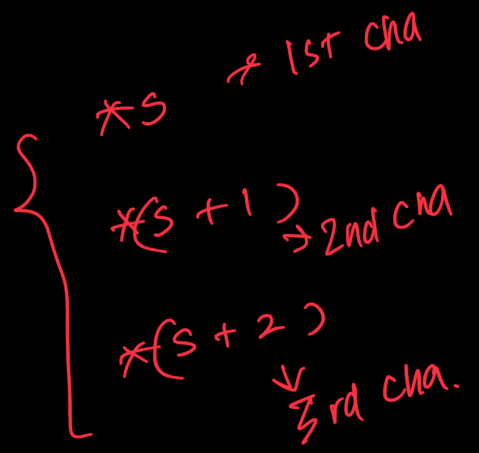
```
typedef struct
{
    string name;
    string number;
}
person;
```

```
typedef char *string;
```

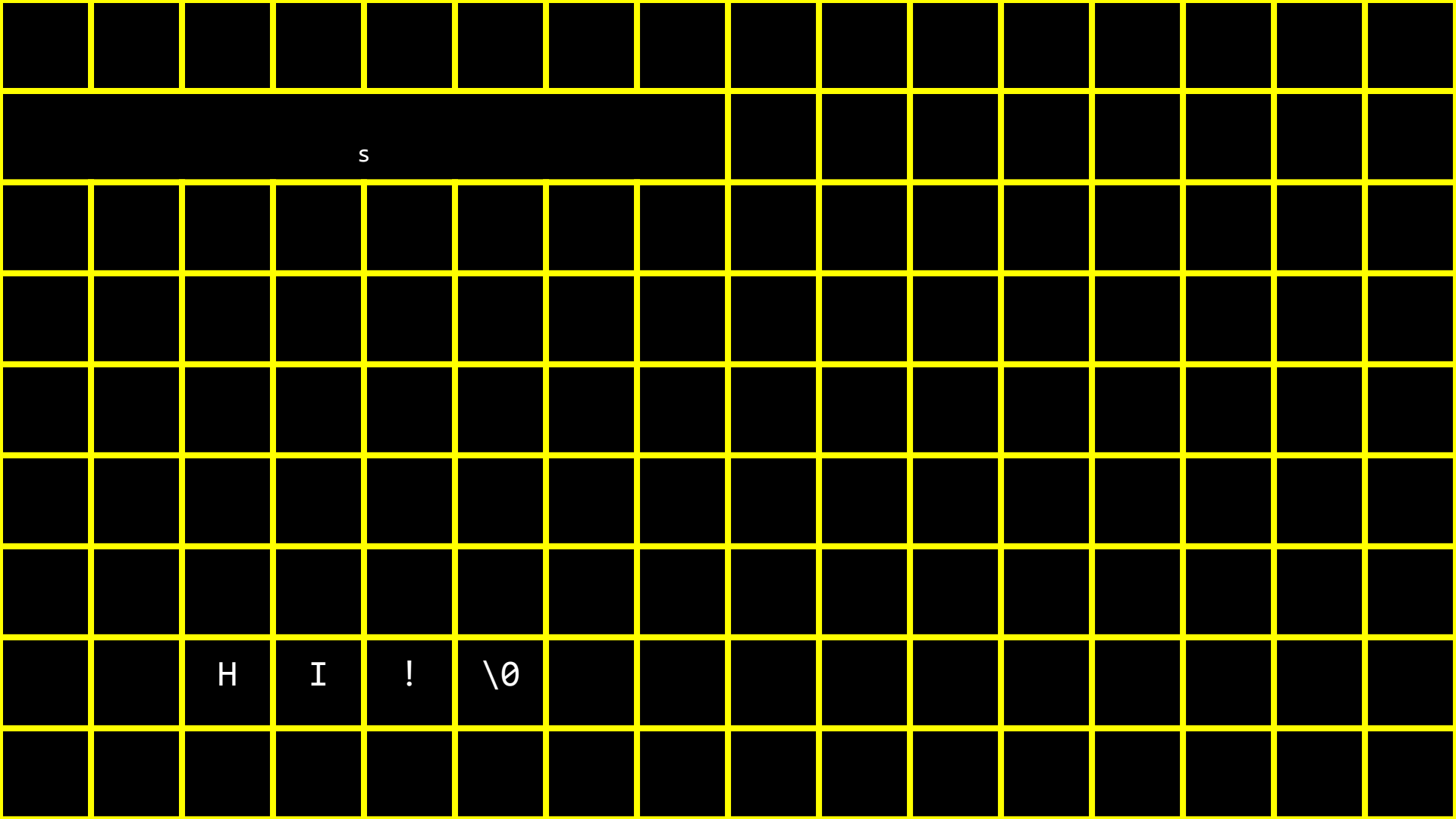
```
typedef char *string;
```

```
typedef char *string;
```

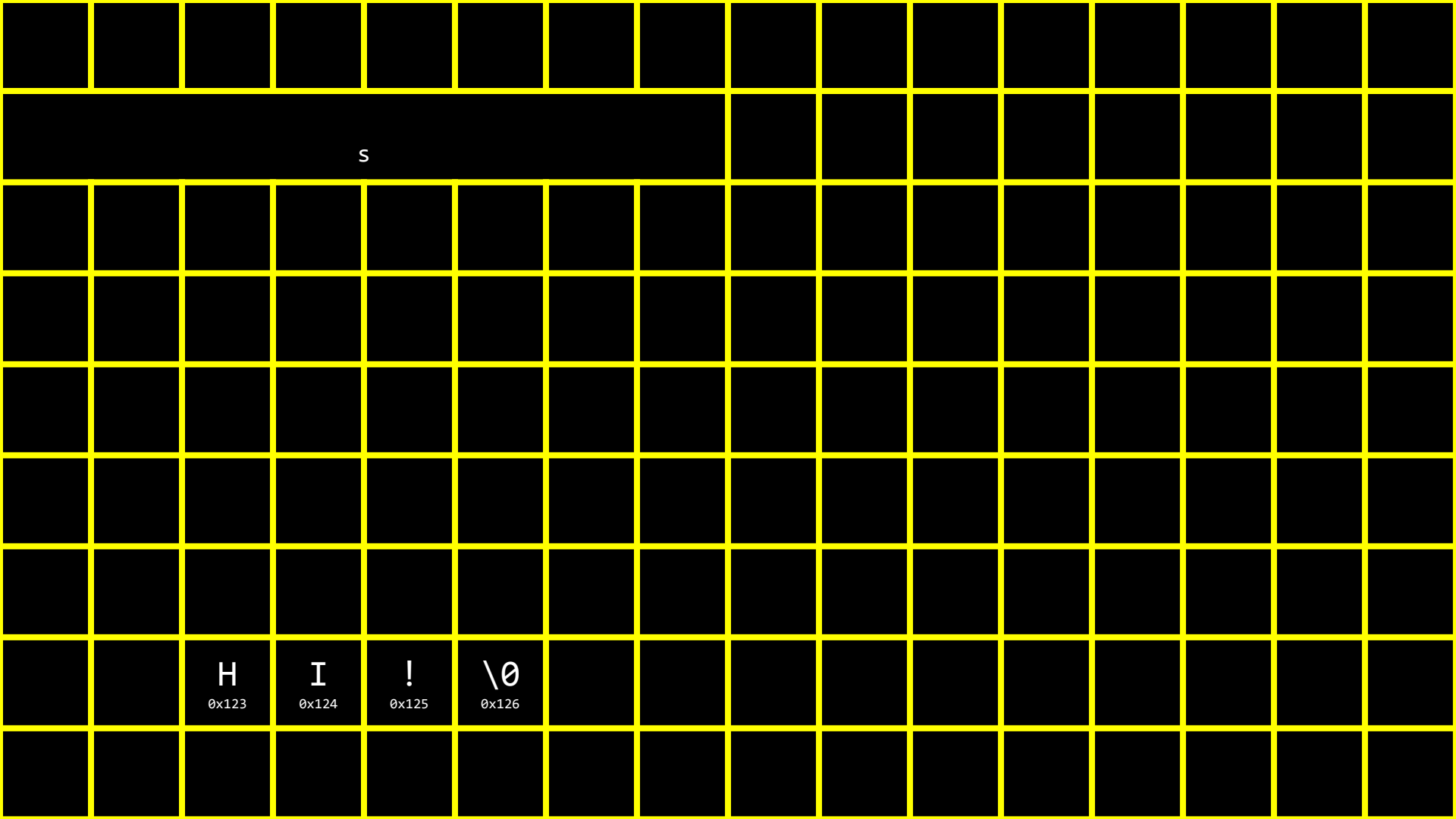
pointer arithmetic

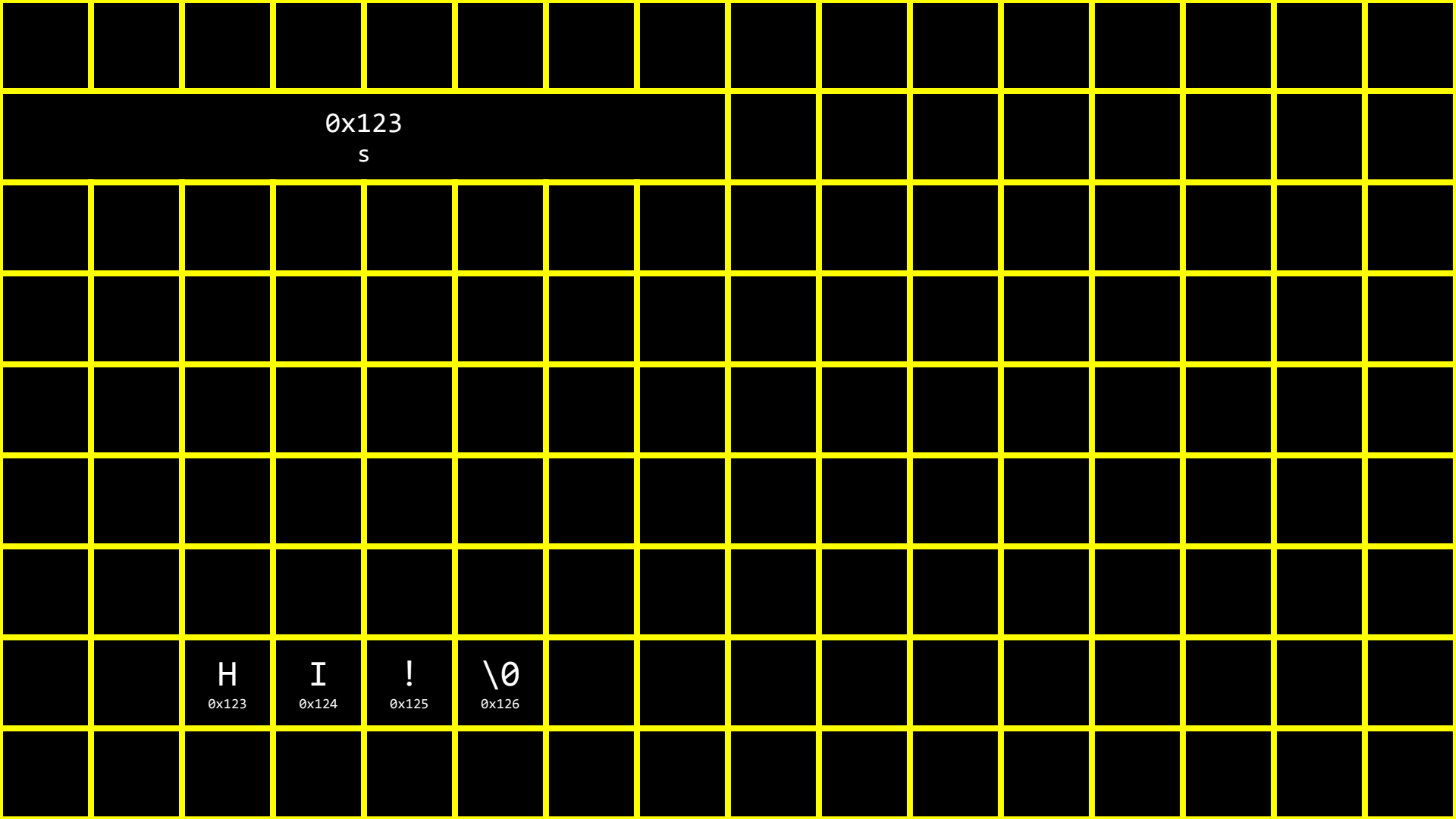


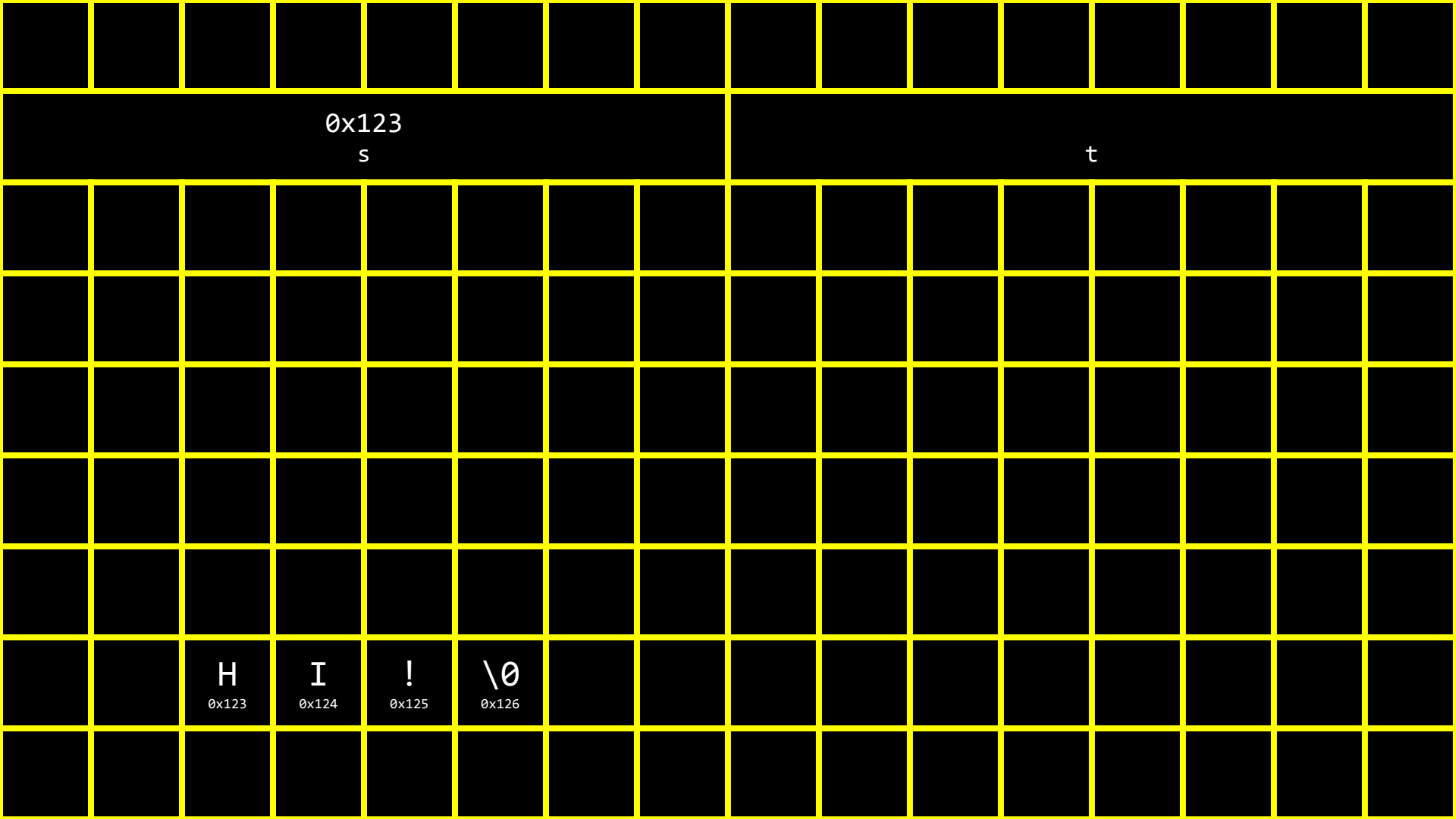
s

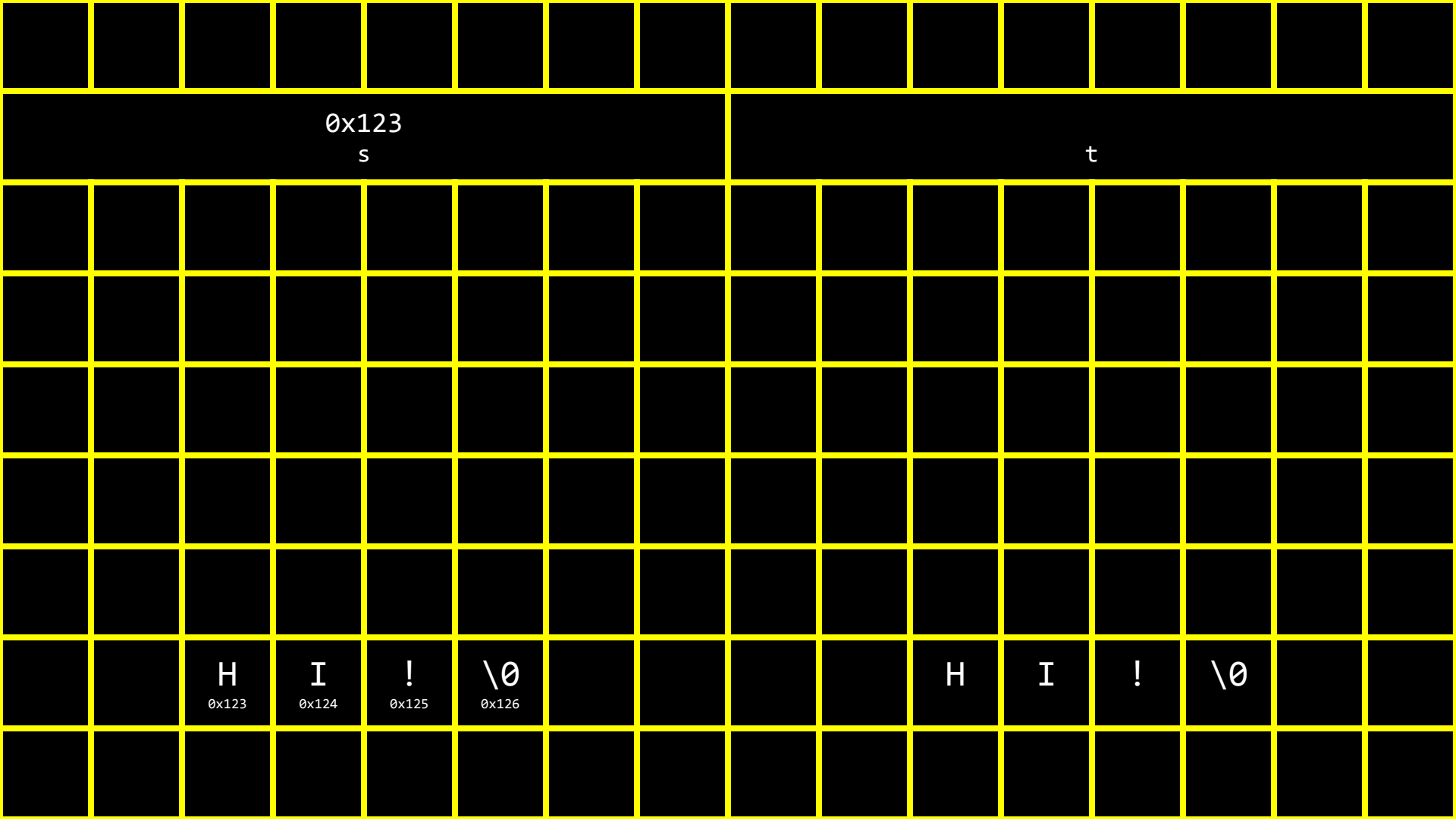

s
H
I
!
\0

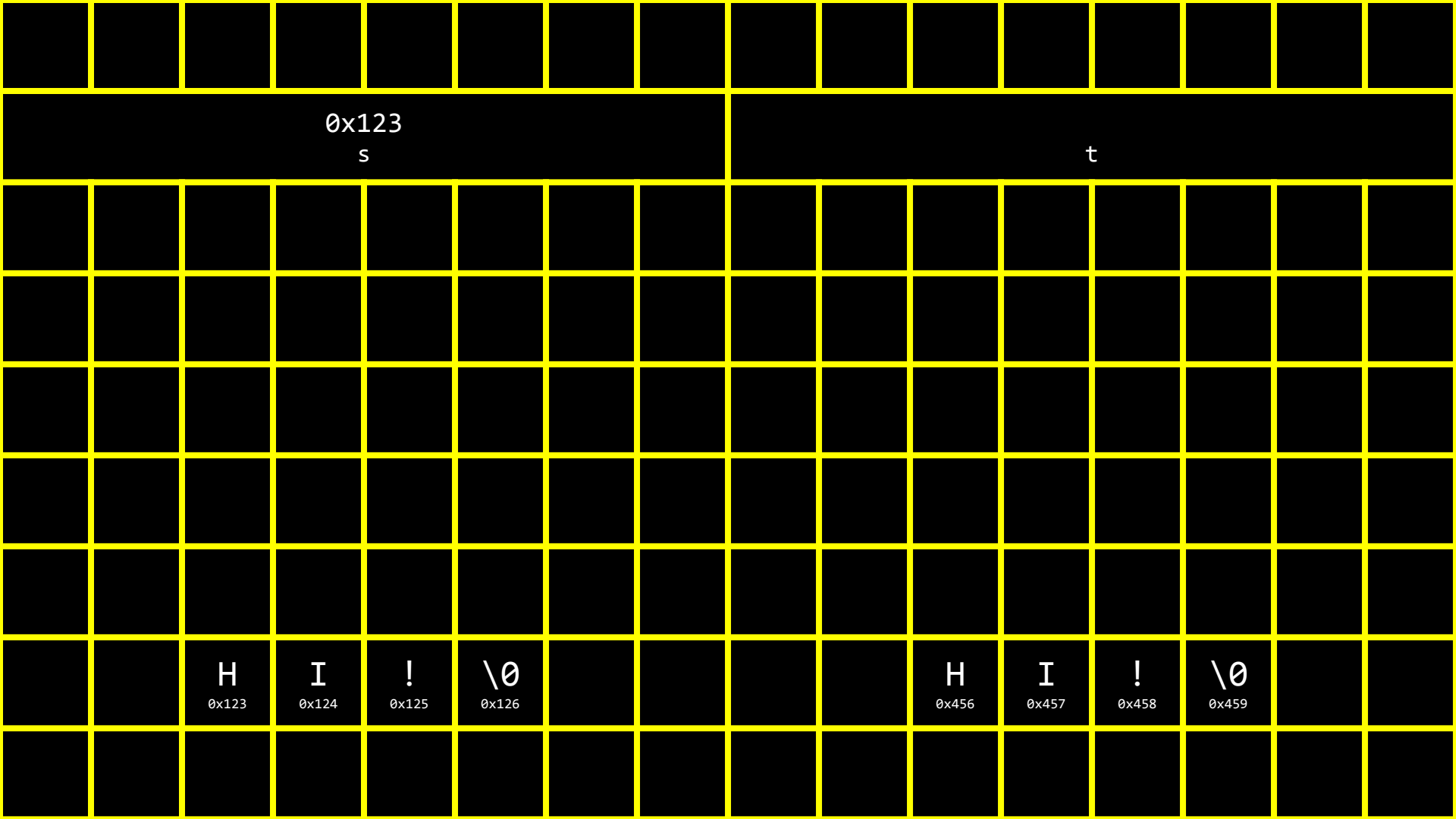
s

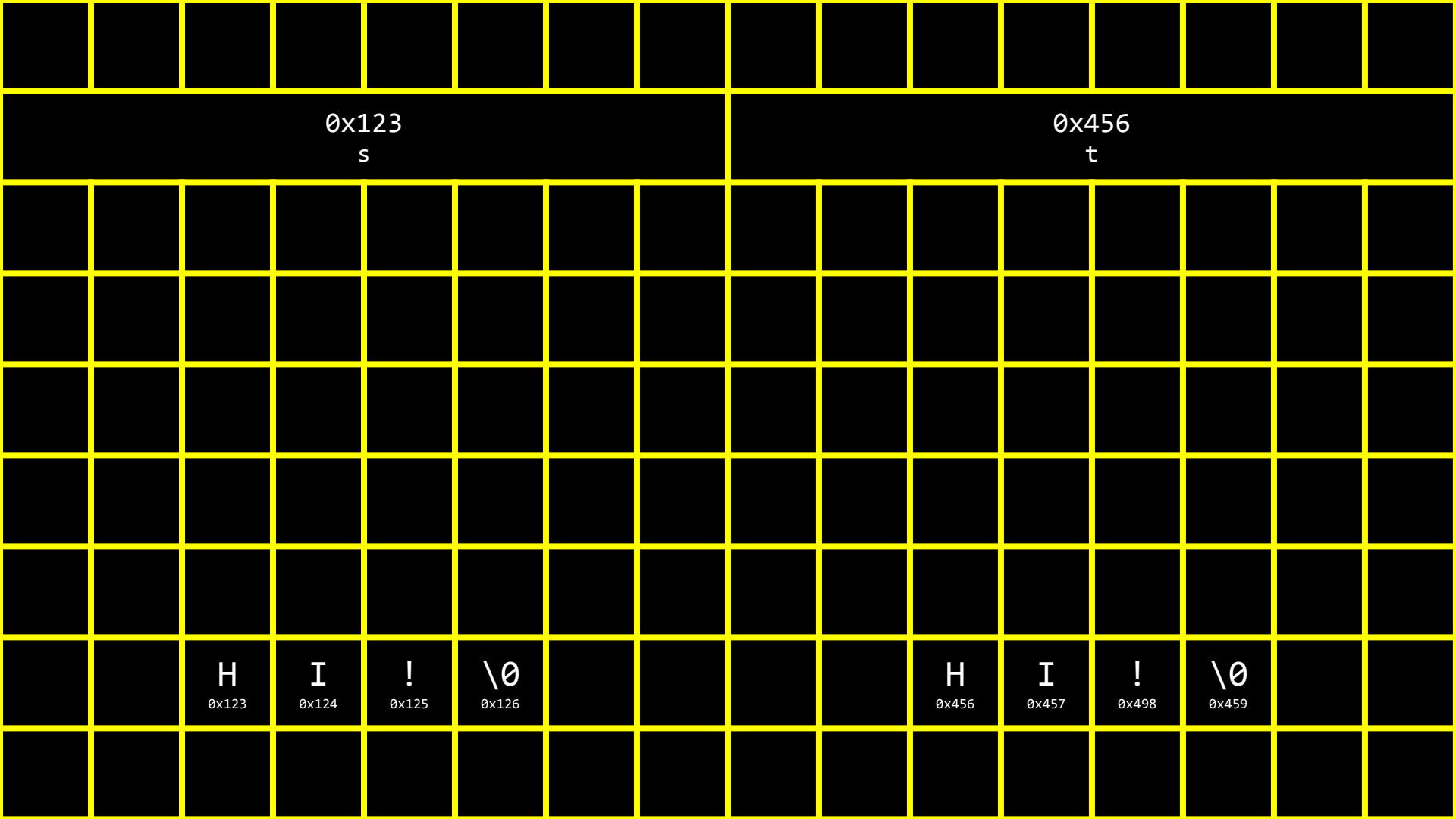


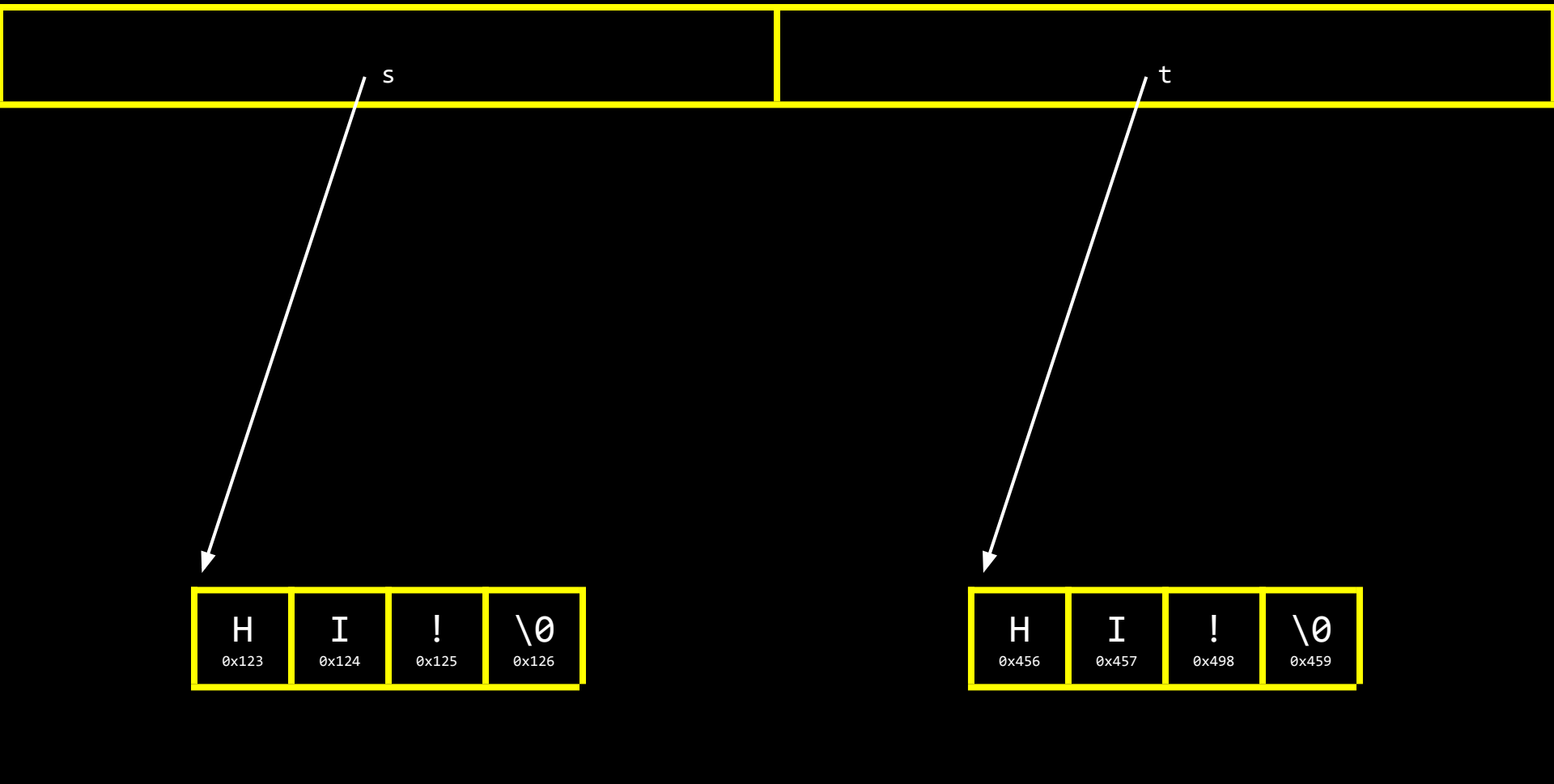

0x123
s
H
0x123
I
0x124
!
0x125
\0
0x126


0x123
s
t
H
0x123
I
0x124
!
0x125
\0
0x126

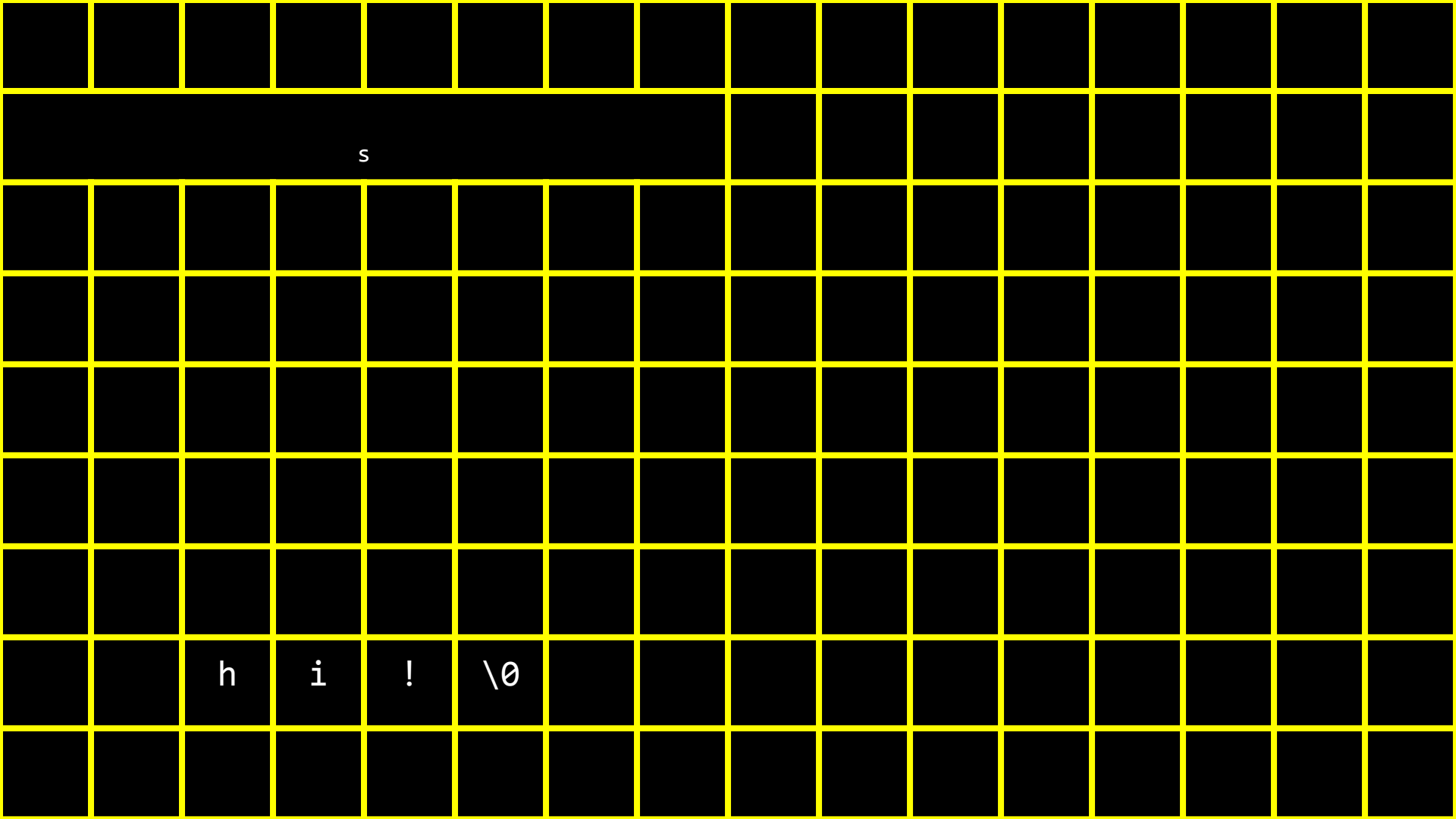

0x123
s
t
H
0x123
I
0x124
!
0x125
\0
0x126
H
I
!
\0


0x123
s
t
H
0x123
I
0x124
!
0x125
\0
0x126
H
0x456
I
0x457
!
0x458
\0
0x459


0x123
s
0x456
t
H
0x123
I
0x124
!
0x125
\0
0x126
H
0x456
I
0x457
!
0x498
\0
0x459

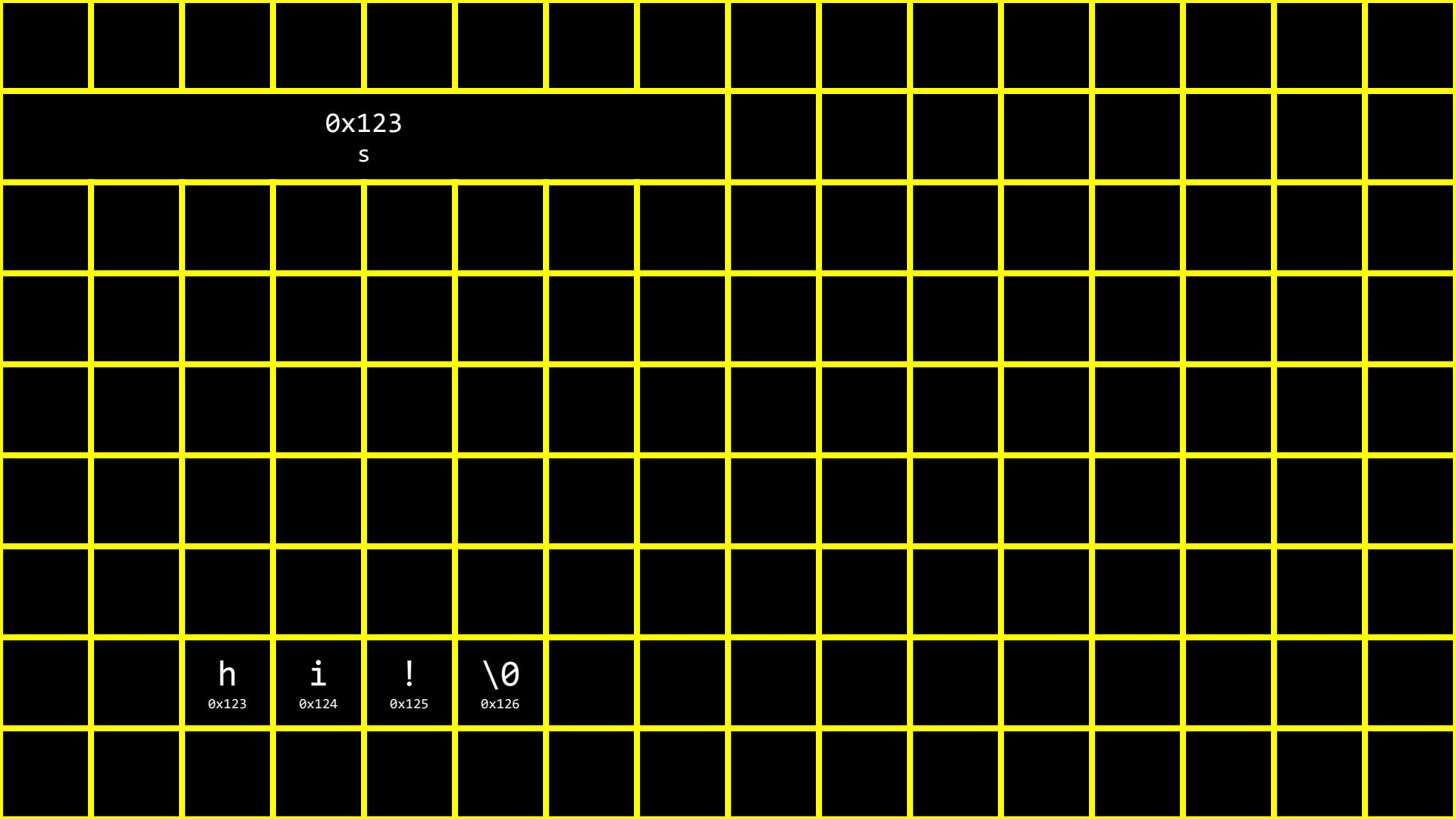

s
t
H
0x123
I
0x124
!
0x125
\0
0x126
H
0x456
I
0x457
!
0x498
\0
0x459

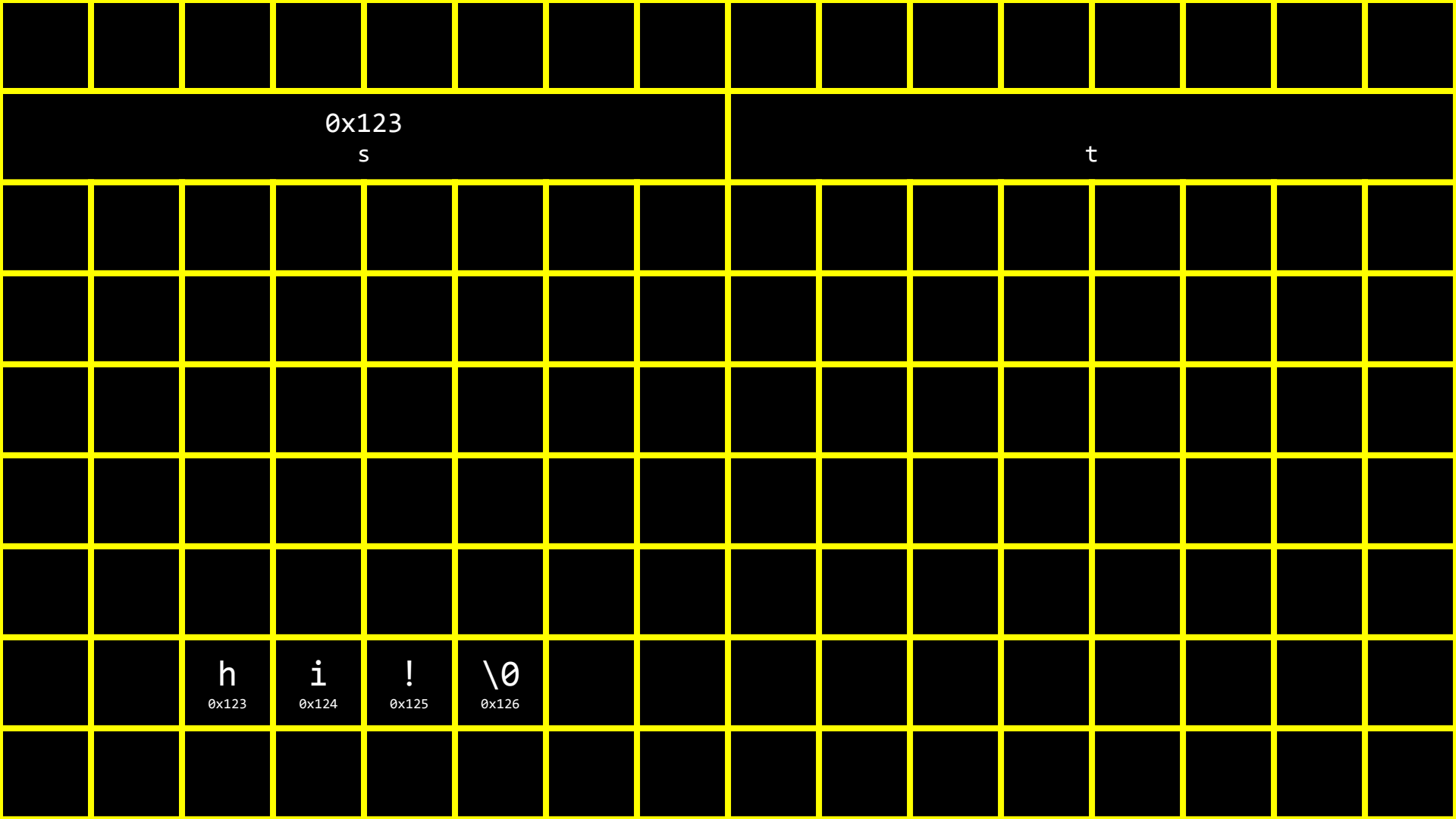
s

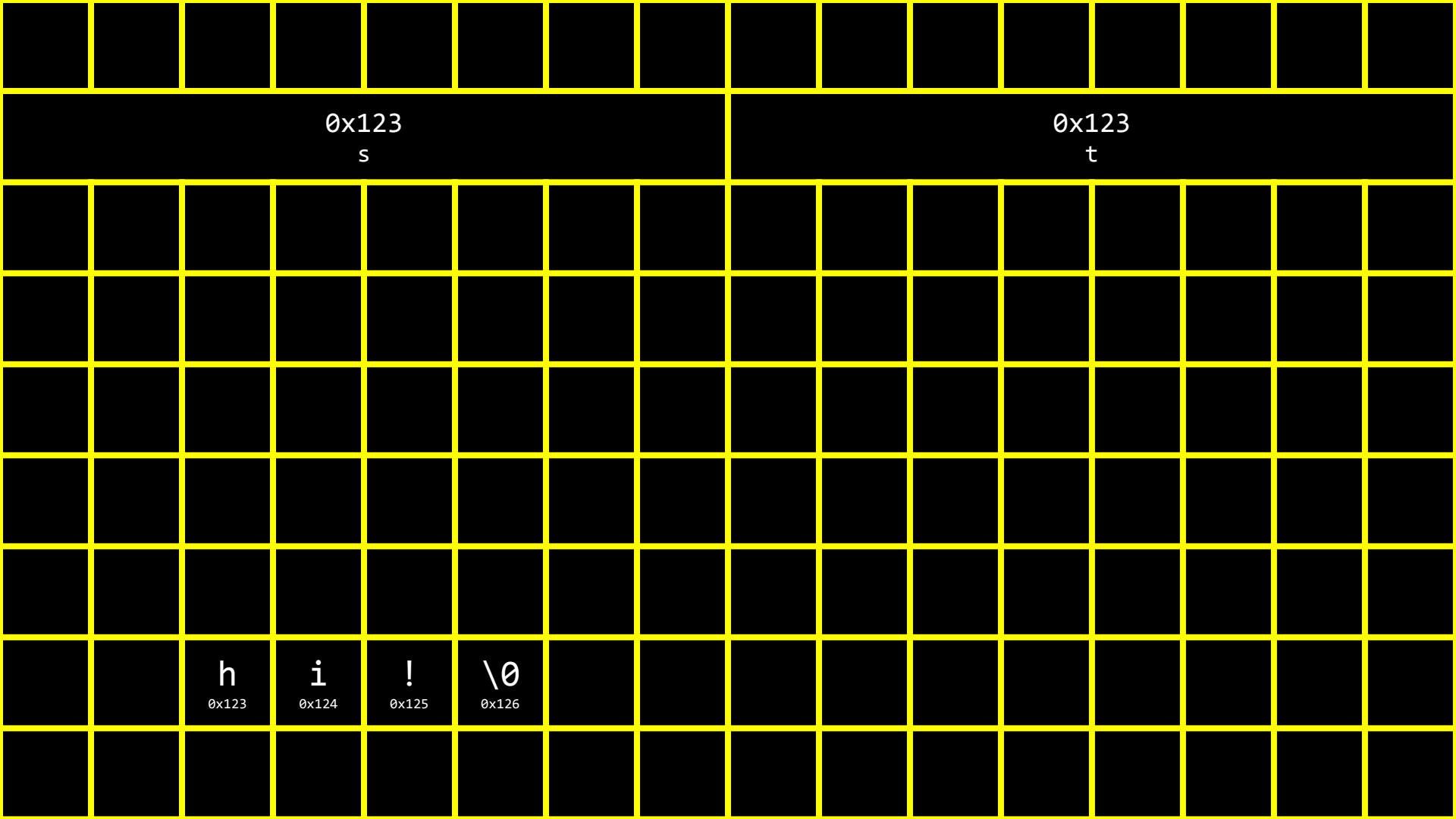

s
h
i
!
\0

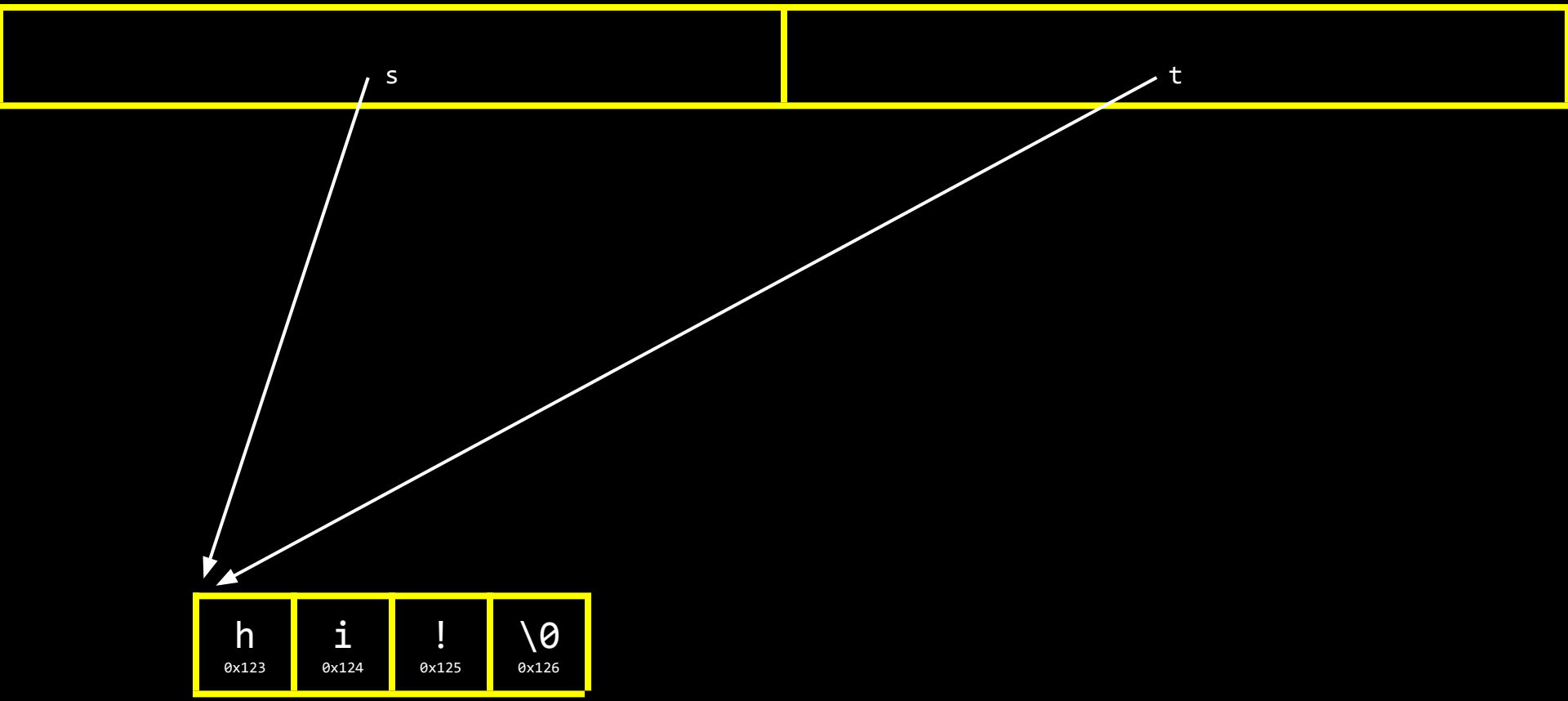
s

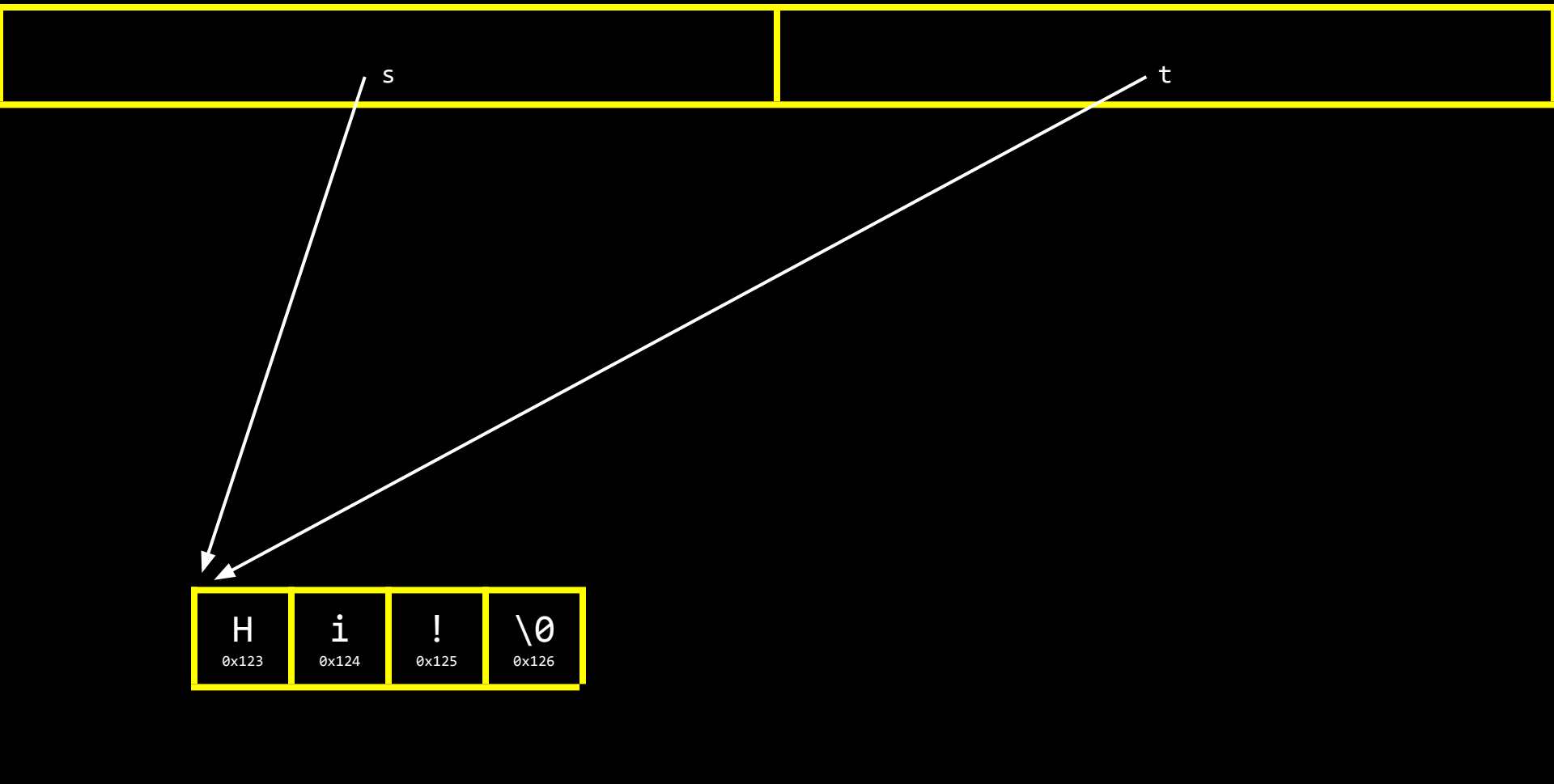
h	i	!	\0
0x123	0x124	0x125	0x126


0x123
s
h
0x123
i
0x124
!
0x125
\0
0x126


0x123
s
t
h
0x123
i
0x124
!
0x125
\0
0x126


0x123
s
0x123
t
h
0x123
i
0x124
!
0x125
\0
0x126

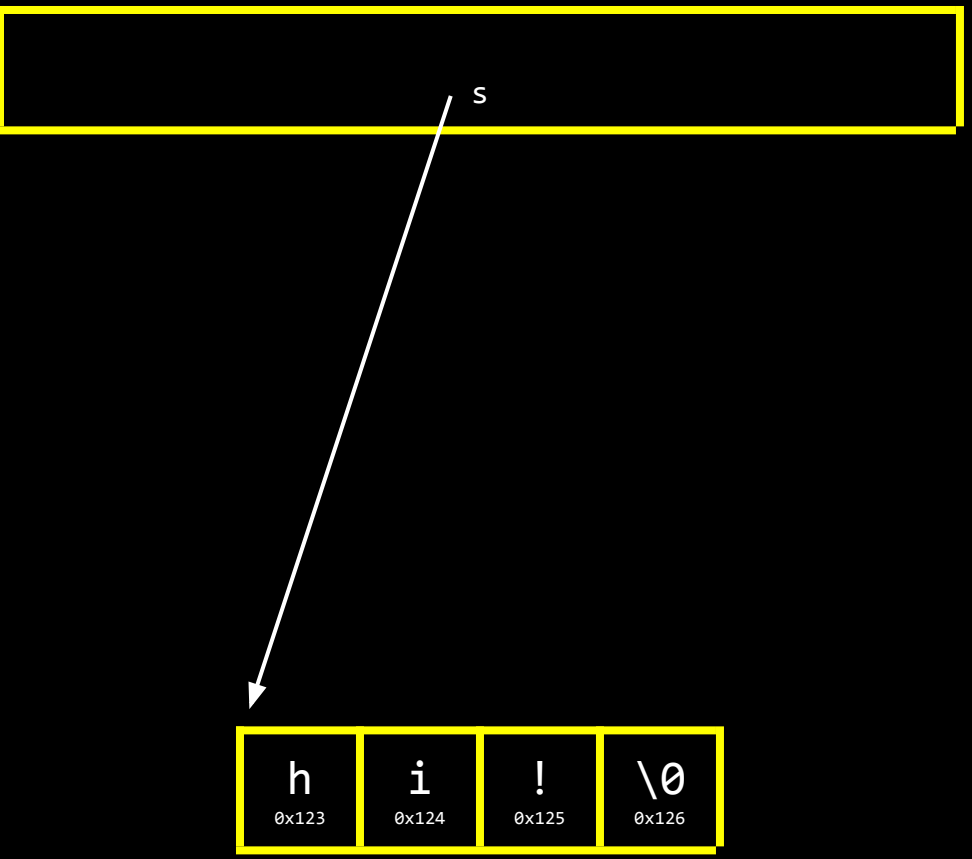

s
t
h
0x123
i
0x124
!
0x125
\0
0x126


s
t
H
0x123
i
0x124
!
0x125
\0
0x126

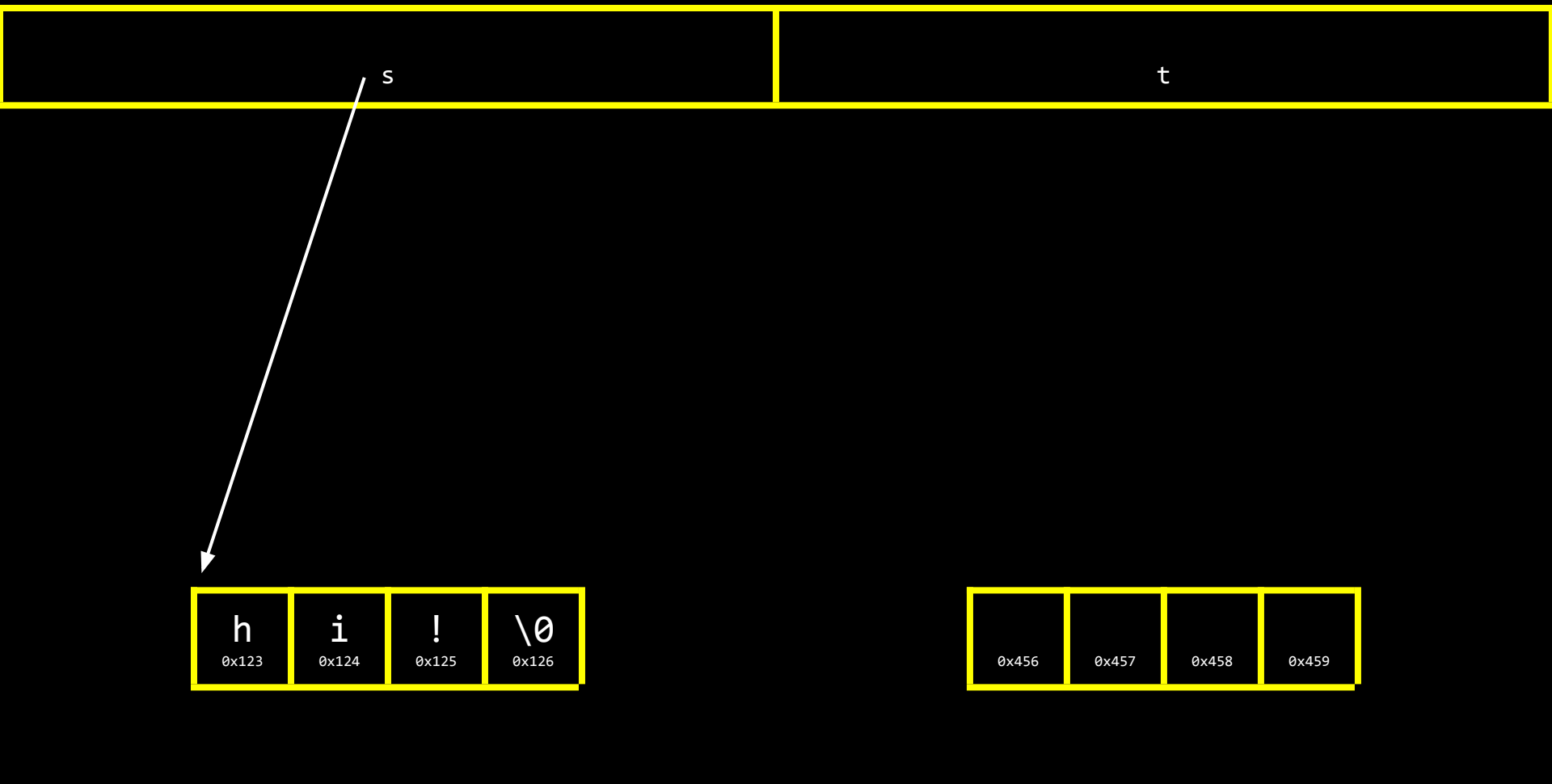
malloc

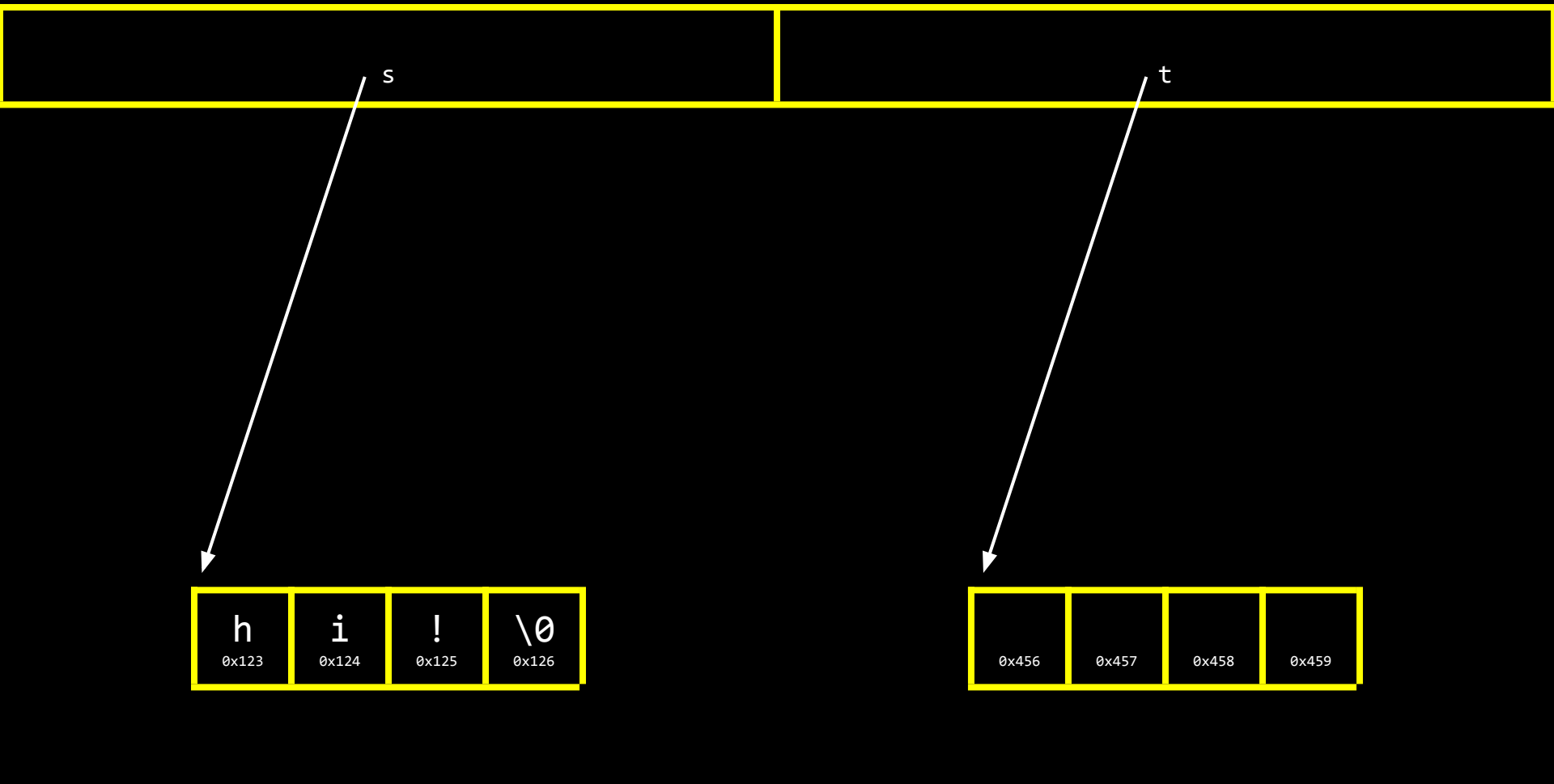
free

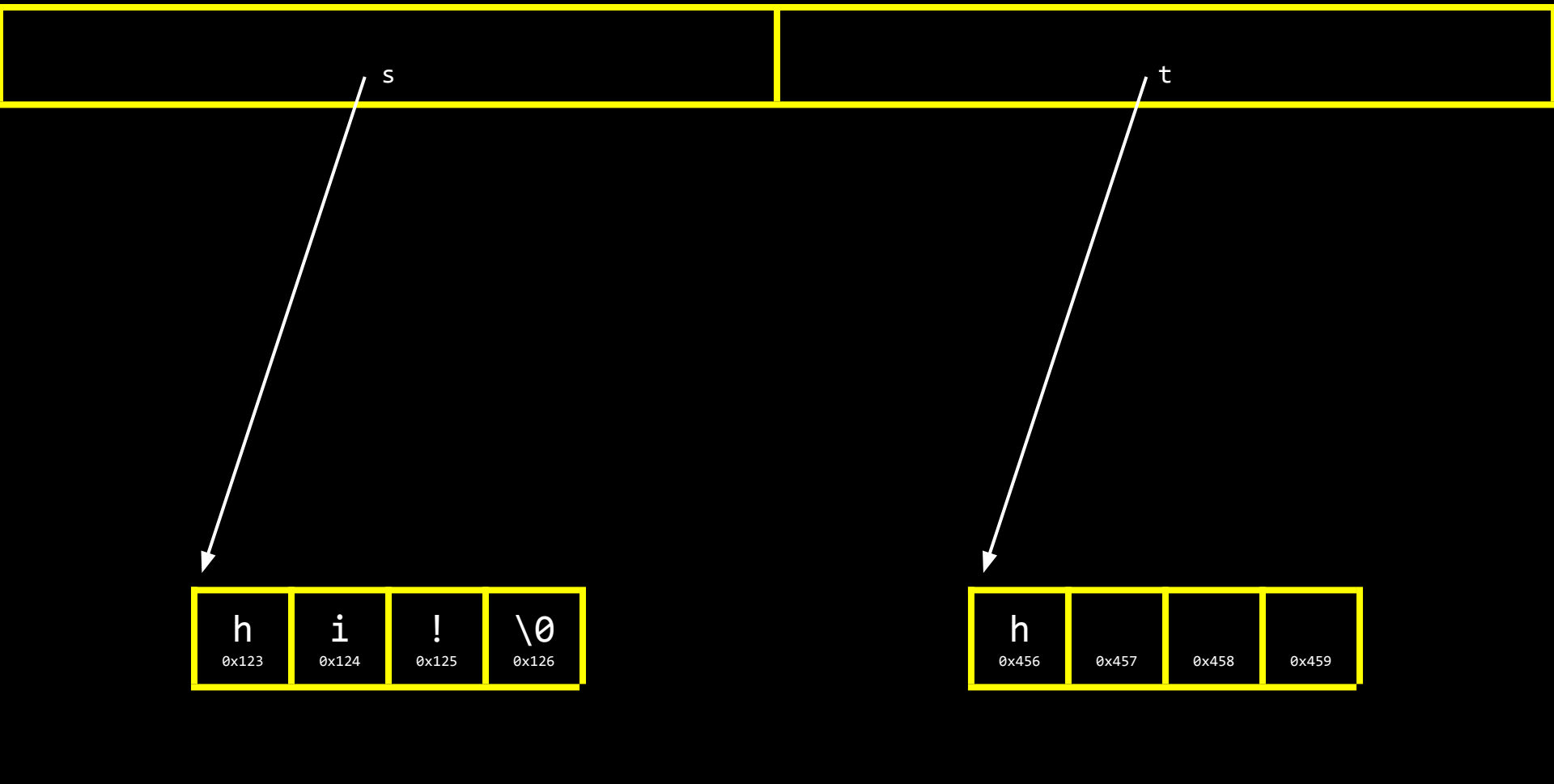
…

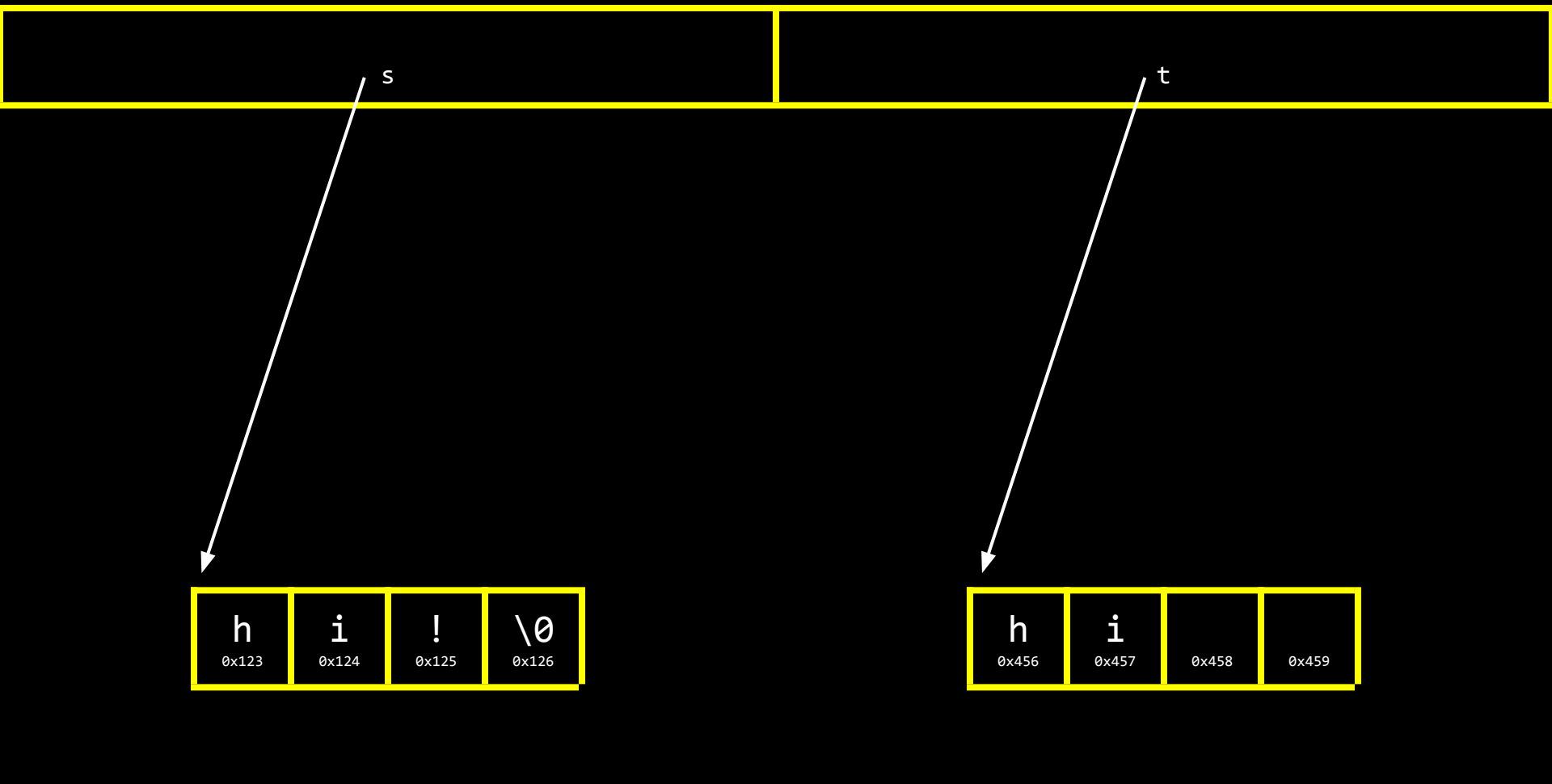

s
h
0x123
i
0x124
!
0x125
\0
0x126

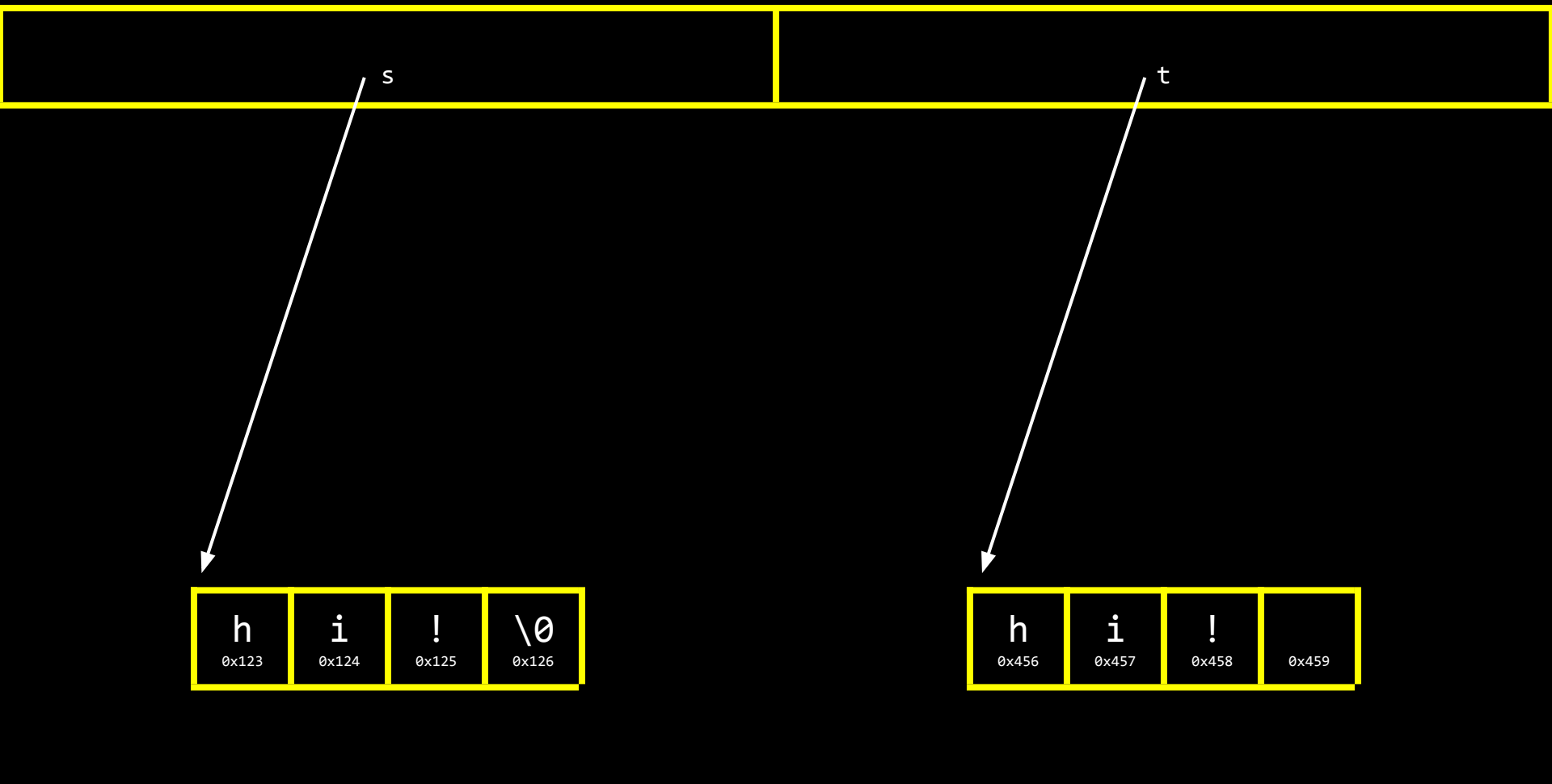

s
t
h
0x123
i
0x124
!
0x125
\0
0x126

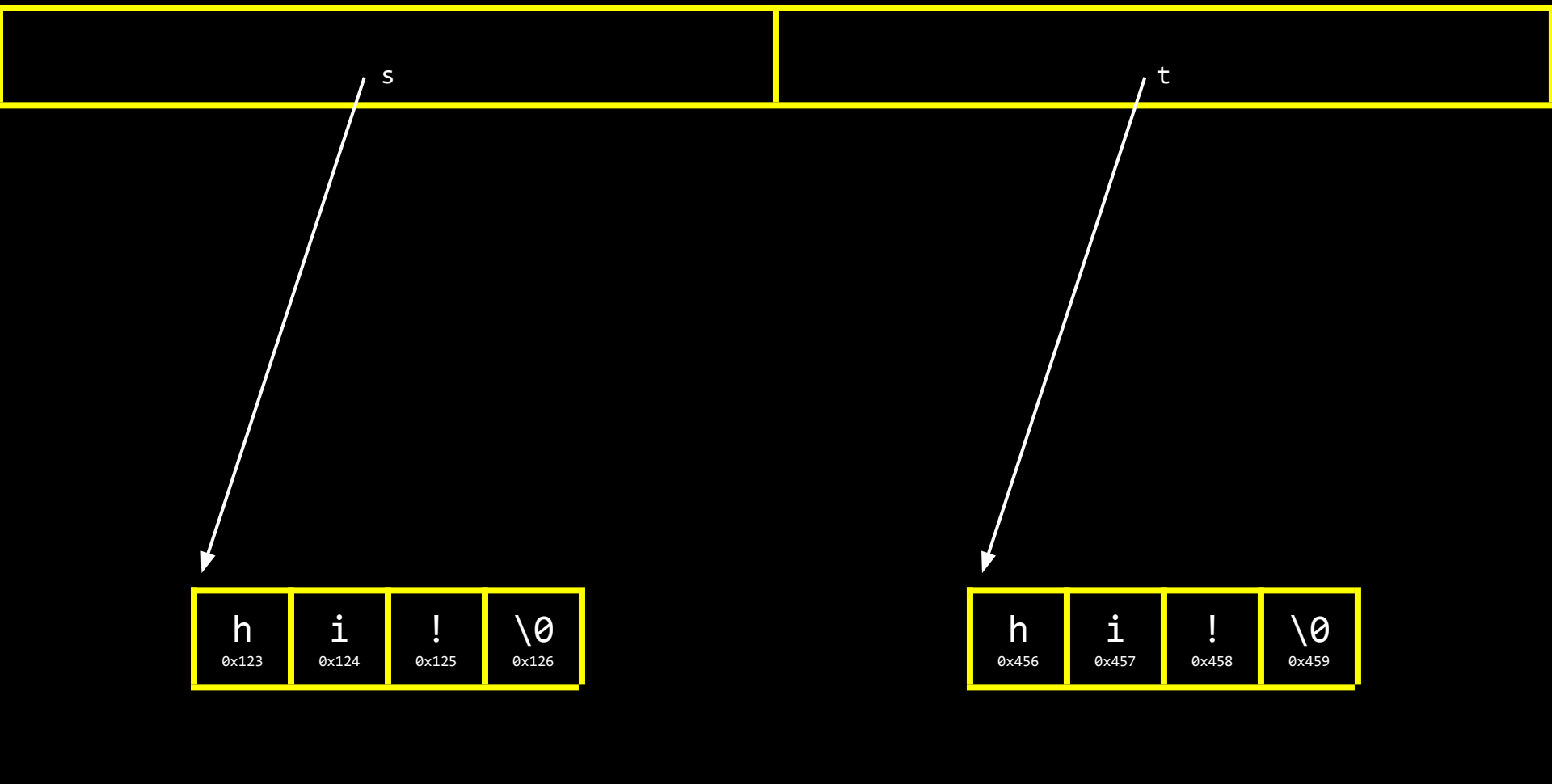

s
t
h
0x123
i
0x124
!
0x125
\0
0x126
0x456
0x457
0x458
0x459

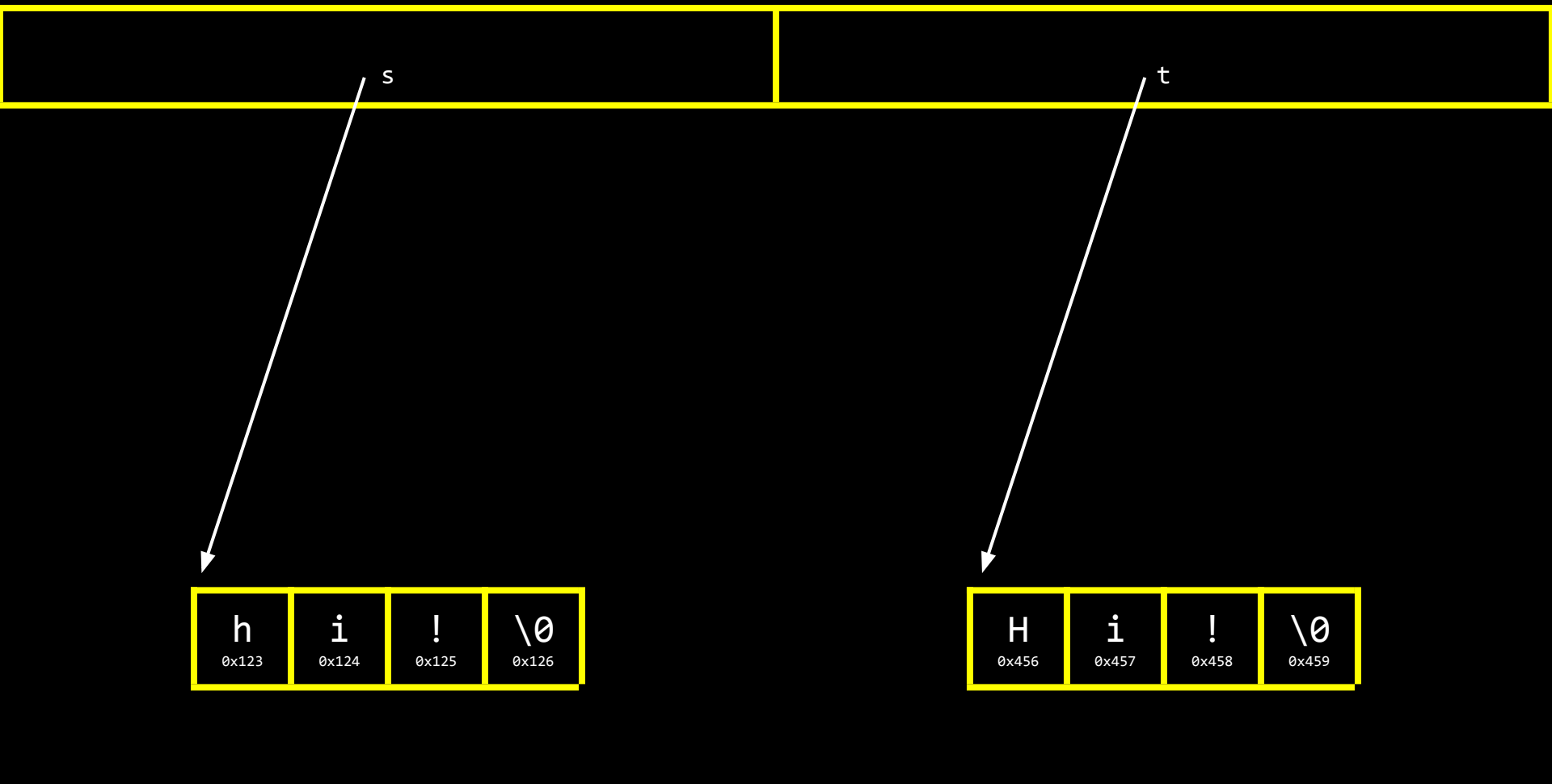

s
t
h
0x123
i
0x124
!
0x125
\0
0x126
0x456
0x457
0x458
0x459


s
t
h
0x123
i
0x124
!
0x125
\0
0x126
h
0x456
0x457
0x458
0x459


s
t
h
0x123
i
0x124
!
0x125
\0
0x126
h
0x456
i
0x457
0x458
0x459


s
t
h
0x123
i
0x124
!
0x125
\0
0x126
h
0x456
i
0x457
!
0x458
0x459


s
t
h
0x123
i
0x124
!
0x125
\0
0x126
h
0x456
i
0x457
!
0x458
\0
0x459


s
t
h
0x123
i
0x124
!
0x125
\0
0x126
H
0x456
i
0x457
!
0x458
\0
0x459

valgrind

garbage values


MAN, I SUCK AT THIS GAME. CAN YOU GIVE ME A FEW POINTERS?
0x3A28213A
0x6339392C,
0x7363682E.
I HATE YOU.

```
int main(void)
{
    int *x;
    int *y;

    x = malloc(sizeof(int));

    *x = 42;
    *y = 13;

    y = x;

    *y = 13;
}
```

```c
int main(void)
{
    int *x;
    int *y;

    x = malloc(sizeof(int));

    *x = 42;
    *y = 13;

    y = x;

    *y = 13;
}
```

```c
int main(void)
{
    int *x;
    int *y;

    x = malloc(sizeof(int));

    *x = 42;
    *y = 13;

    y = x;

    *y = 13;
}
```

```c
int main(void)
{
    int *x;
    int *y;

    x = malloc(sizeof(int));

    *x = 42;
    *y = 13;

    y = x;

    *y = 13;
}
```

```c
int main(void)
{
    int *x;
    int *y;

    x = malloc(sizeof(int));

    *x = 42;
    *y = 13;

    y = x;

    *y = 13;
}
```

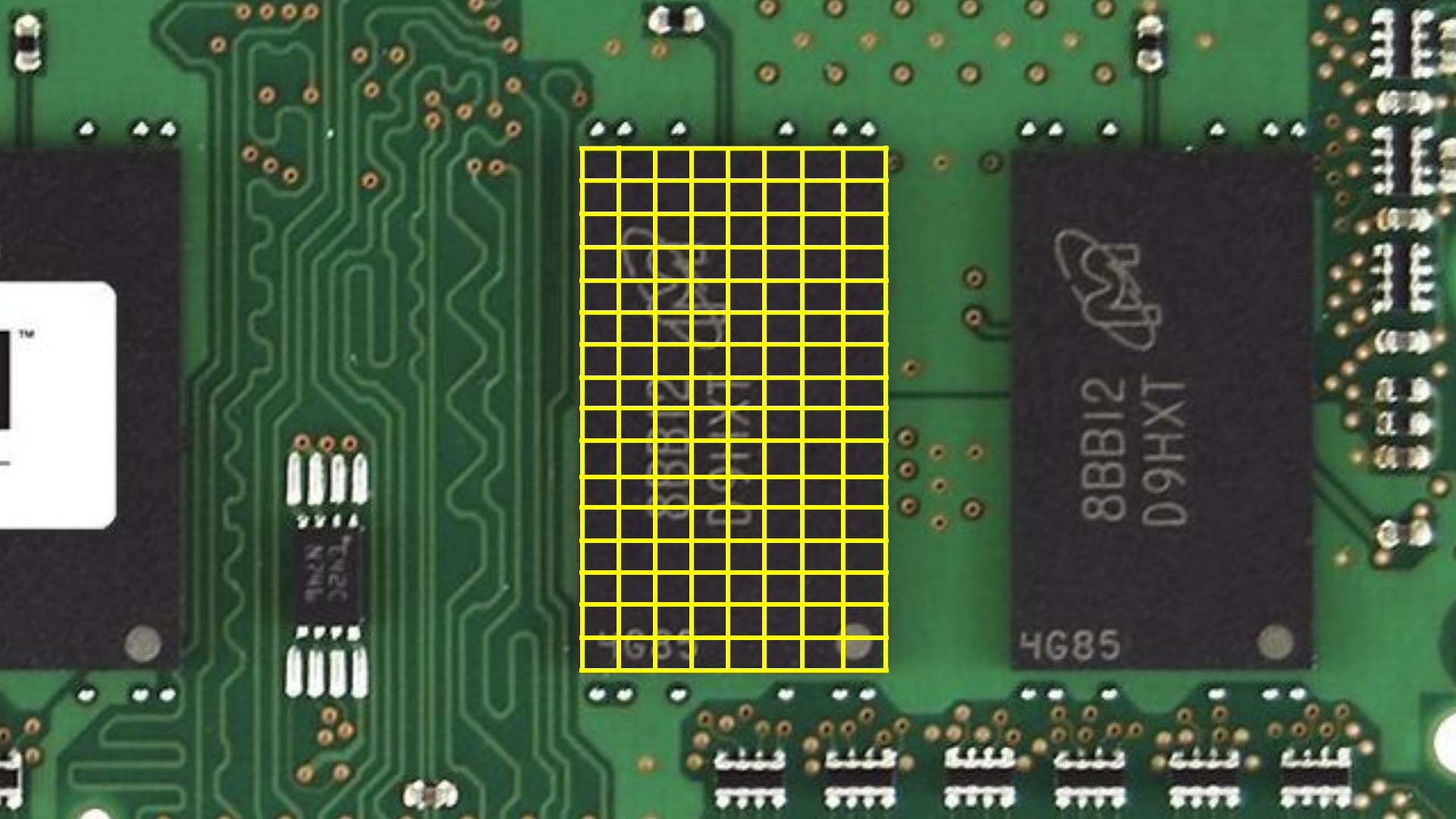
```
void swap(int a, int b)
{



}
```

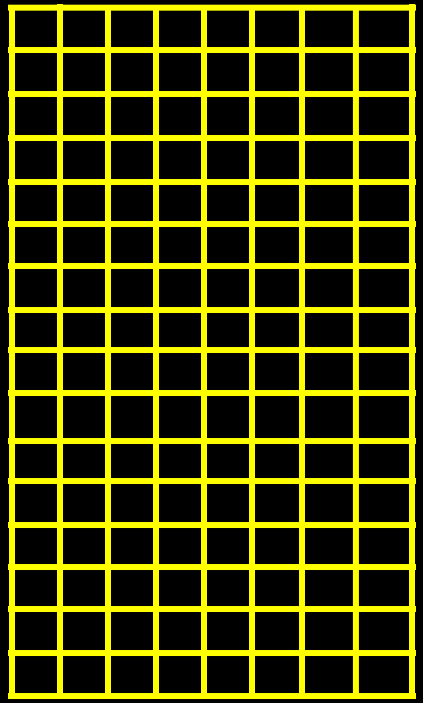
```
void swap(int a, int b)
{
    int tmp = a;
    a = b;
    b = tmp;
}
```

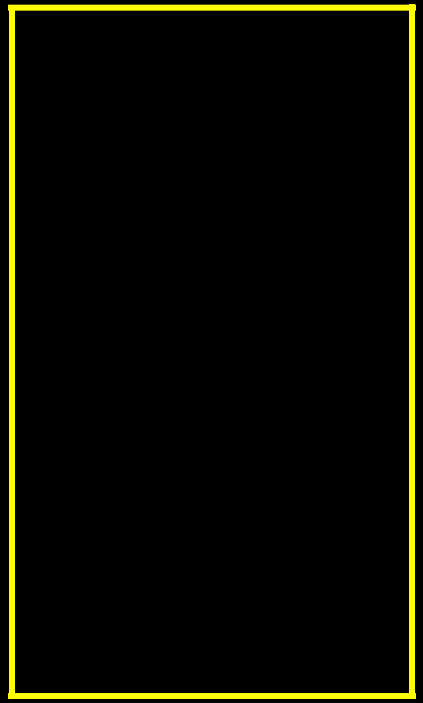
```
void swap(int a, int b)
{
    int tmp = a;
    a = b;
    b = tmp;
}
```

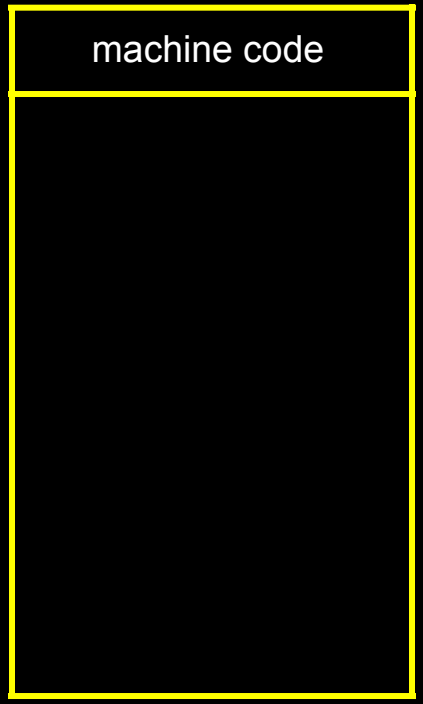

crucial
by Micron
8BBI2
D9HXT
8BBI2
D9HXT
4G85
4G85
4G85
47J6

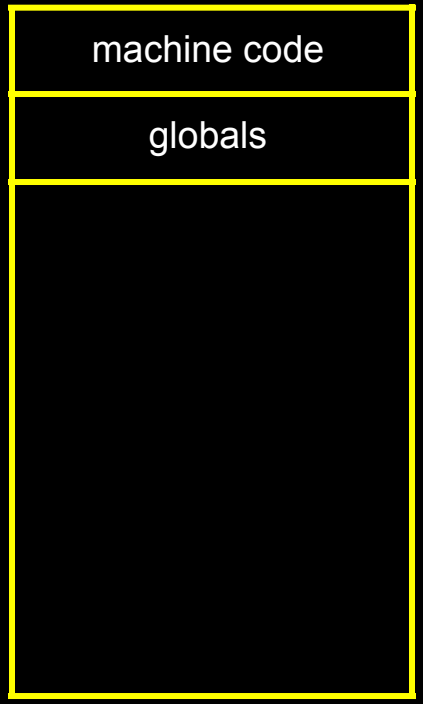

8BBI2
D9HXT
4G85
8BBI2
D9HXT
4G85

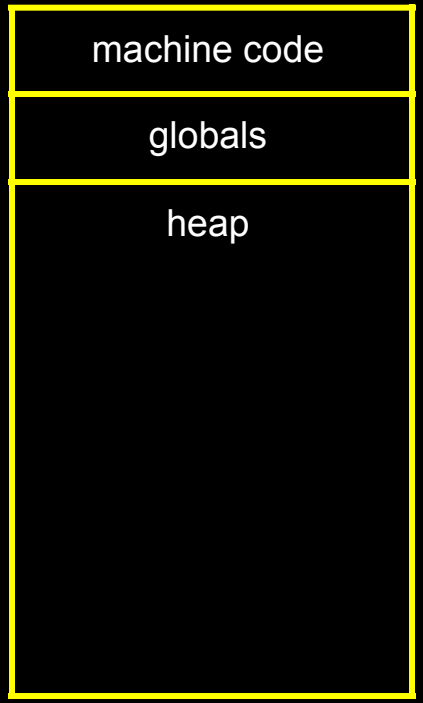

8BBI2
D9HXT
4G85

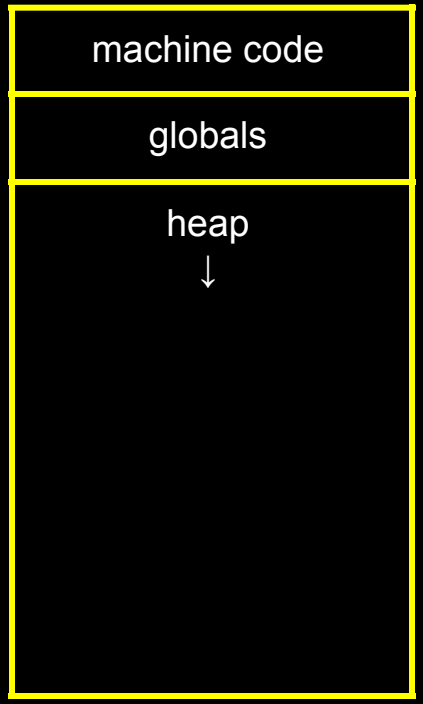


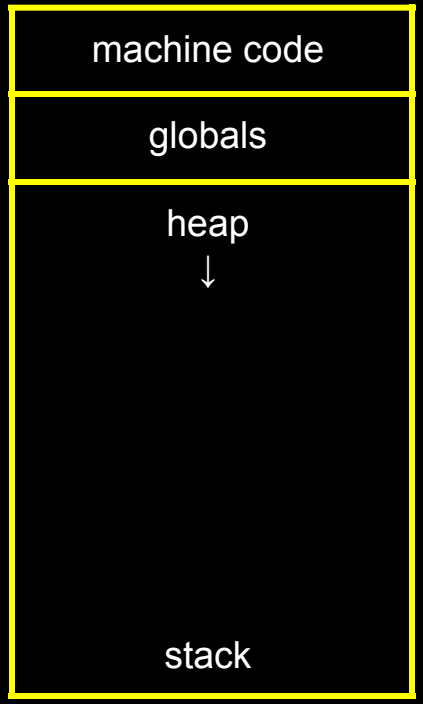


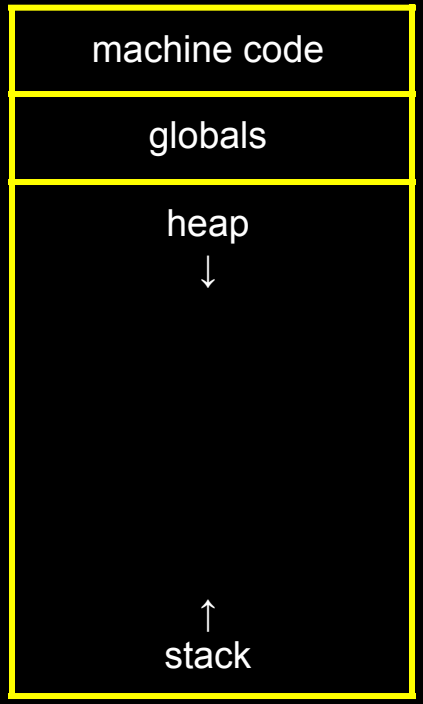

machine code

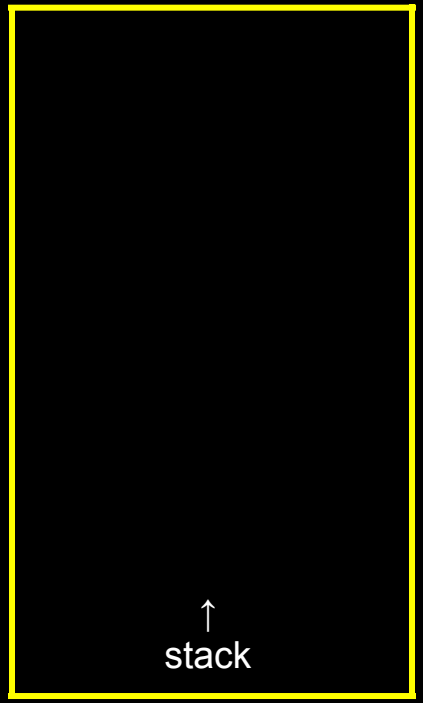

machine code
globals


machine code
globals
heap

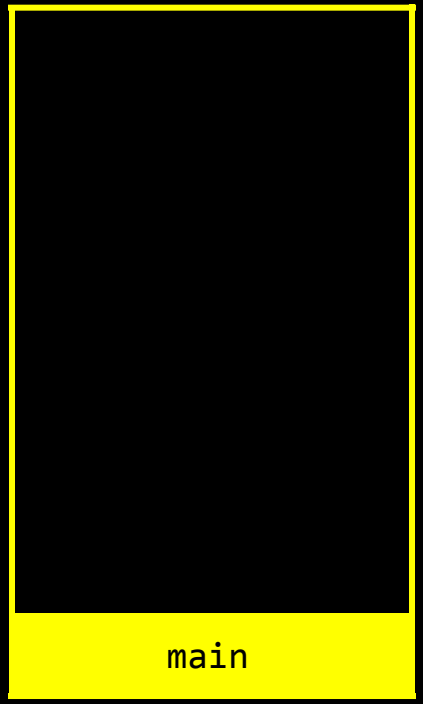

machine code
globals
heap
↓

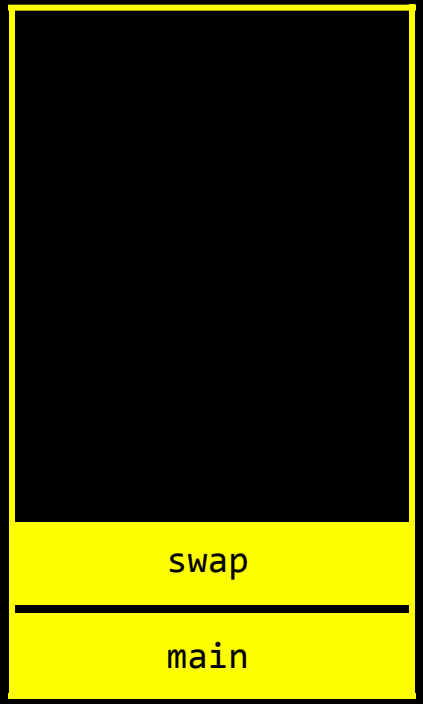

machine code
globals
heap
↓
stack

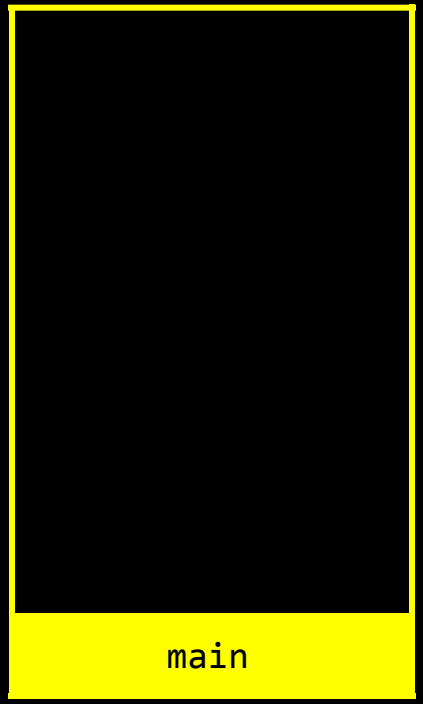

machine code
globals
heap
↓
↑
stack


↑
stack

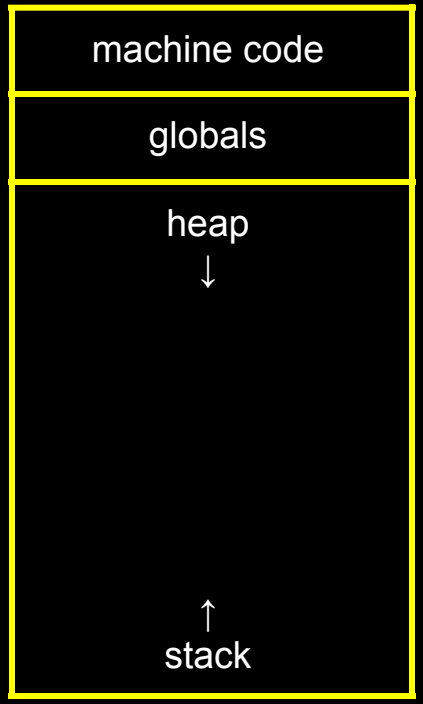
```
void swap(int a, int b)
{
    int tmp = a;
    a = b;
    b = tmp;
}
```

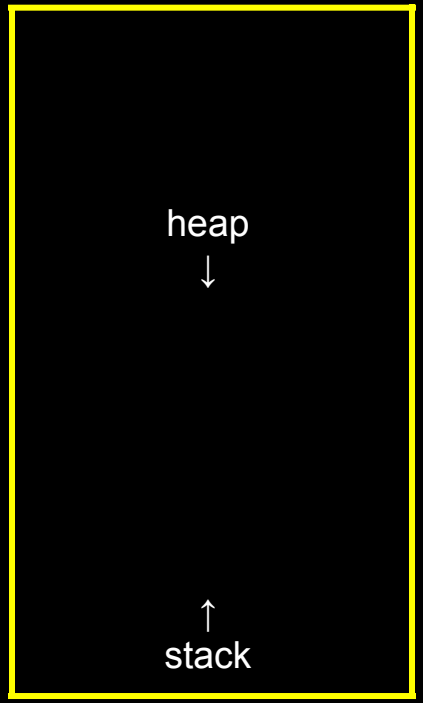

main


swap
main


main

```
void swap(int *a, int *b)
{
    int tmp = *a;
    *a = *b;
    *b = tmp;
}
```


machine code
globals
heap
↓
↑
stack


heap
↓
↑
stack

heap overflow

stack overflow

buffer overflow

```
get_char

get_double

get_float

get_int

get_long

get_string

...
```

scanf

...

file I/O

JPEG

0xFF 0xD8 0xFF

BMP

offset	type	name	
0	WORD	bfType	BITMAPFILEHEADER
2	DWORD	bfSize	
6	WORD	bfReserved1	
8	WORD	bfReserved2	
10	DWORD	bfOffBits	
14	DWORD	biSize	BITMAPINFOHEADER
18	LONG	biWidth	
22	LONG	biHeight	
26	WORD	biPlanes	
28	WORD	biBitCount	
30	DWORD	biCompression	
34	DWORD	biSizeImage	
38	LONG	biXPelsPerMeter	
42	LONG	biYPelsPerMeter	
46	DWORD	biClrUsed	
50	DWORD	biClrImportant	
54	BYTE	rgbtBlue	RGBTRIPLE
55	BYTE	rgbtGreen	
56	BYTE	rgbtRed	
57	BYTE	rgbtBlue	RGBTRIPLE
58	BYTE	rgbtGreen	
59	BYTE	rgbtRed	

...

243	BYTE	rgbtBlue	RGBTRIPLE
244	BYTE	rgbtGreen	
245	BYTE	rgbtRed	

This is CS50