This is CS50

learn how to use memory

reading levels

One fish. Two fish. Red fish. Blue fish.

Before Grade 1

Congratulations! Today is your day. You're off to Great Places! You're off and away!

Grade 3

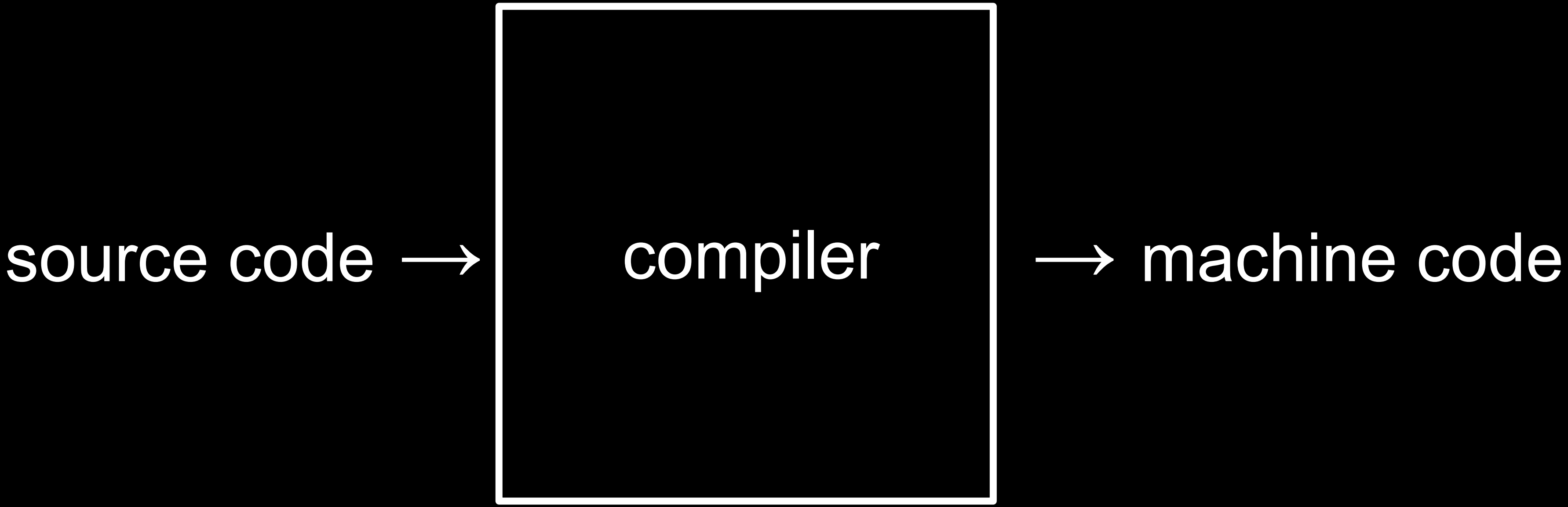
It was a bright cold day in April, and the clocks were striking thirteen. Winston Smith, his chin nuzzled into his breast in an effort to escape the vile wind, slipped quickly through the glass doors of Victory Mansions, though not quickly enough to prevent a swirl of gritty dust from entering along with him.

Grade 10

reading levels

cryptography

UIJT XBT DT50



source code → compiler → machine code

```c
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");
}
```

```
01111111 01000101 01001100 01000110 00000010 00000001 00000001 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000010 00000000 00111110 00000000 00000001 00000000 00000000 00000000
10110000 00000101 01000000 00000000 00000000 00000000 00000000 00000000
01000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
11010000 00010011 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 01000000 00000000 00111000 00000000
00001001 00000000 01000000 00000000 00100100 00000000 00100001 00000000
00000110 00000000 00000000 00000000 00000101 00000000 00000000 00000000
01000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
01000000 00000000 01000000 00000000 00000000 00000000 00000000 00000000
01000000 00000000 01000000 00000000 00000000 00000000 00000000 00000000
11111000 00000001 00000000 00000000 00000000 00000000 00000000 00000000
11111000 00000001 00000000 00000000 00000000 00000000 00000000 00000000
00001000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000011 00000000 00000000 00000000 00000100 00000000 00000000 00000000
00111000 00000010 00000000 00000000 00000000 00000000 00000000 00000000
...
```

```
make hello

./hello
```

```
clang hello.c

./a.out
```

```
clang -o hello hello.c

./hello
```

```
#include <stdio.h>


int main(void)
{
    printf("hello, world\n");
}
```

```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    string name = get_string("What's your name? ");
    printf("hello, %s\n", name);
}
```

```
clang -o hello hello.c -lcs50

./hello
```

```
make hello

./hello
```

compiling

preprocessing

compiling

assembling

linking

preprocessing

compiling

assembling

linking

```c
#include <stdio.h>

void meow(void);

int main(void)
{
    for (int i = 0; i < 3; i++)
    {
        meow();
    }
}

void meow(void)
{
    printf("meow\n");
}
```

```c
#include <stdio.h>

void meow(void);

int main(void)
{
    for (int i = 0; i < 3; i++)
    {
        meow();
    }
}

void meow(void)
{
    printf("meow\n");
}
```

```c
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    string name = get_string("What's your name? ");
    printf("hello, %s\n", name);
}
```

```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    string name = get_string("What's your name? ");
    printf("hello, %s\n", name);
}
```

```
string get_string(string prompt);
#include <stdio.h>

int main(void)
{
    string name = get_string("What's your name? ");
    printf("hello, %s\n", name);
}
```

```
string get_string(string prompt);
#include <stdio.h>

int main(void)
{
    string name = get_string("What's your name? ");
    printf("hello, %s\n", name);
}
```

```
string get_string(string prompt);
int printf(string format, ...);

int main(void)
{
    string name = get_string("What's your name? ");
    printf("hello, %s\n", name);
}
```

preprocessing

compiling

assembling

linking

```
string get_string(string prompt);
int printf(string format, ...);

int main(void)
{
    string name = get_string("What's your name? ");
    printf("hello, %s\n", name);
}
```

```
...
main:                                   # @main
    .cfi_startproc
# BB#0:
    pushq    %rbp
.Ltmp0:
    .cfi_def_cfa_offset 16
.Ltmp1:
    .cfi_offset %rbp, -16
    movq    %rsp, %rbp
.Ltmp2:
    .cfi_def_cfa_register %rbp
    subq    $16, %rsp
    xorl    %eax, %eax
    movl    %eax, %edi
    movabsq    $.L.str, %rsi
    movb    $0, %al
    callq    get_string
    movabsq    $.L.str.1, %rdi
    movq    %rax, -8(%rbp)
    movq    -8(%rbp), %rsi
    movb    $0, %al
    callq    printf
    ...
```

```
...
main:                                   # @main
    .cfi_startproc
# BB#0:
    pushq    %rbp
.Ltmp0:
    .cfi_def_cfa_offset 16
.Ltmp1:
    .cfi_offset %rbp, -16
    movq    %rsp, %rbp
.Ltmp2:
    .cfi_def_cfa_register %rbp
    subq    $16, %rsp
    xorl    %eax, %eax
    movl    %eax, %edi
    movabsq    $.L.str, %rsi
    movb    $0, %al
    callq    get_string
    movabsq    $.L.str.1, %rdi
    movq    %rax, -8(%rbp)
    movq    -8(%rbp), %rsi
    movb    $0, %al
    callq    printf
    ...
```

```
...
main:                                   # @main
    .cfi_startproc
# BB#0:
    pushq    %rbp
.Ltmp0:
    .cfi_def_cfa_offset 16
.Ltmp1:
    .cfi_offset %rbp, -16
    movq    %rsp, %rbp
.Ltmp2:
    .cfi_def_cfa_register %rbp
    subq    $16, %rsp
    xorl    %eax, %eax
    movl    %eax, %edi
    movabsq    $.L.str, %rsi
    movb    $0, %al
    callq    get_string
    movabsq    $.L.str.1, %rdi
    movq    %rax, -8(%rbp)
    movq    -8(%rbp), %rsi
    movb    $0, %al
    callq    printf
    ...
```

preprocessing

compiling

assembling

linking

```
...
main:                                   # @main
    .cfi_startproc
# BB#0:
    pushq    %rbp
.Ltmp0:
    .cfi_def_cfa_offset 16
.Ltmp1:
    .cfi_offset %rbp, -16
    movq    %rsp, %rbp
.Ltmp2:
    .cfi_def_cfa_register %rbp
    subq    $16, %rsp
    xorl    %eax, %eax
    movl    %eax, %edi
    movabsq    $.L.str, %rsi
    movb    $0, %al
    callq    get_string
    movabsq    $.L.str.1, %rdi
    movq    %rax, -8(%rbp)
    movq    -8(%rbp), %rsi
    movb    $0, %al
    callq    printf
    ...
```

```
01111111010001010100110001000110
00000010000000010000000100000000
00000000000000000000000000000000
00000000000000000000000000000000
00000001000000000011111000000000
00000001000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
10100000000000100000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
01000000000000000000000000000000
00000000000000000100000000000000
00001010000000000000000100000000
01010101010010001000100111100101
01001000100000111110110000010000
00110001110000001000100111000111
01001000101111100000000000000000
00000000000000000000000000000000
00000000000000001011000000000000
11101000000000000000000000000000
00000000010010001011111100000000
00000000000000000000000000000000
00000000000000000000000001001000
...
```

preprocessing

compiling

assembling

linking

```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    string name = get_string("What's your name? ");
    printf("hello, %s\n", name);
}
```

hello.c

hello.c cs50.c

hello.c cs50.c stdio.c

```
01111111010001010100110001000110
00000010000000010000000100000000
00000000000000000000000000000000
00000000000000000000000000000000
00000001000000000011111000000000
00000001000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
10100000000000100000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
01000000000000000000000000000000
00000000000000000100000000000000
00001010000000000000000100000000
01010101010010001000100111100101
01001000100000111110110000010000
00110001110000001000100111000111
01001000101111100000000000000000
00000000000000000000000000000000
00000000000000001011000000000000
11101000000000000000000000000000
00000000010010001011111100000000
00000000000000000000000000000000
00000000000000000000000001001000
...
```

cs50.c

stdio.c

```
01111111010001010100110001000110   01111111010001010100110001000110
00000010000000010000000100000000   00000010000000010000000100000000
00000000000000000000000000000000   00000000000000000000000000000000
00000000000000000000000000000000   00000000000000000000000000000000
00000001000000000011111000000000   00000011000000000011111000000000
00000001000000000000000000000000   00000001000000000000000000000000
00000000000000000000000000000000   11000000000011110000000000000000
00000000000000000000000000000000   00000000000000000000000000000000
00000000000000000000000000000000   01000000000000000000000000000000
00000000000000000000000000000000   00000000000000000000000000000000
10100000000000010000000000000000   00101000000110010000000000000000
00000000000000000000000000000000   00000000000000000000000000000000
00000000000000000000000000000000   00000000000000000000000000000000
01000000000000000000000000000000   01000000000000000001110000000000
00000000000000000100000000000000   00000111000000000100000000000000
00001010000000000000000100000000   00011100000000000001100100000000
01010101010010001000100111100101   00000001000000000000000000000000
01001000100000111110110000010000   00000101000000000000000000000000
00110001110000001000100111000111   00000000000000000000000000000000
01001000101111110000000000000000   00000000000000000000000000000000
00000000000000000000000000000000   00000000000000000000000000000000
00000000000000001011000000000000   00000000000000000000000000000000
11101000000000000000000000000000   00000000000000000000000000000000
00000000010010001011111100000000   00000000000000000000000000000000
00000000000000000000000000000000   01011100001001010000000000000000
00000000000000000000000001001000   00000000000000000000000000000000
...                                ...
```

stdio.c

```
01111111010001010100110001000110
00000010000000010000000100000000
00000000000000000000000000000000
00000000000000000000000000000000
00000001000000000011111000000000
00000001000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
10100000000000100000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
01000000000000000000000000000000
00000000000000000100000000000000
00001010000000000000000100000000
01010101010010001000100111100101
01001000100000111110110000010000
00110001110000001000100111000111
01001000101111100000000000000000
00000000000000000000000000000000
00000000000000001011000000000000
11101000000000000000000000000000
00000000010010001011111100000000
00000000000000000000000000000000
00000000000000000000000001001000
...
```

```
01111111010001010100110001000110
00000010000000010000000100000000
00000000000000000000000000000000
00000000000000000000000000000000
00000011000000000011111000000000
00000001000000000000000000000000
11000000000011110000000000000000
00000000000000000000000000000000
01000000000000000000000000000000
00000000000000000000000000000000
00101000001100100000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
01000000000000000011100000000000
00000111000000000100000000000000
00011100000000000001100100000000
00000001000000000000000000000000
00000101000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
01011100001001010000000000000000
00000000000000000000000000000000
...
```

```
00101111011011000110100101100010
01100011001011100111001101101111
00101110001101100010000000101111
01110101011100110111001000101111
01101100011010010110001000101111
01111000001110000011011001011111
00110110001101000010110101101100
01101001011011100111010101111000
00101101011001110110111001110101
00101111011011000110100101100010
01100011010111110110111001101111
01101110011100110110100001100001
01110010011001010110010000101110
01100001001000000010000001000001
01010011010111110100111001000101
01000101010001000100010101000100
00100000001010000010000000101111
01101100011010010110001000101111
01111000001110000011011001011111
00110110001101000010110101101100
01101001011011100111010101111000
00101101011001110110111001110101
00101111011011000110010000101101
01101100011010010110111001110101
01111000001011010111100000111000
00110110001011010011011000110100
...
```

```
011111110100010101001100010001100000001000000001000000010000000000000000000000000000000000000000000000
000000000000000000000000000000000100000000000011111000000000000000000100000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000001010000000000010000000000000000000000000000000000000000000000000000000000000000000000000
000000000100000000000000000000000000000000000000000000000100000000000000000010100000000000000001000000
000101010101001000100010011110010101001000100000111110110000010000001100011100000010001001110001110100
100010111110000000000000000000000000000000000000000000000000000000000000000010110000000000000011101000
000000000000000000000000000000010010001011111100000000000000000000000000000000000000000000000000000000
0000000001001000...01111111010001010100110001000110000000100000000100000001000000000000000000000000000
000000000000000000000000000000000000000000000000000110000000000111110000000000000000100000000000000000
000000011000000000011110000000000000000000000000000000000000000000000000001000000000000000000000000000
000000000000000000000000000000000001010000000110010000000000000000000000000000000000000000000000000000
000000000000000000000000000010000000000000000111000000000000000011100000000000100000000000000000111000
000000000110010000000000000010000000000000000000000000000010100000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000010111000010010100000000000000
00000000000000000000000000000000000...0010111101101100011010010110001001100011001011100111001101101111
001011100011011000100000001011110111010101110011011100100010111101101100011010010110001000101111011110
000011100000011011001011110011011000110100001011010110110001101001011011100111010101111000001011010110
011101101110011101010010111101101100011010010110001001100011010111110110111001101111011011100111001101
101000011000001011100100110010101100100001011100110000100100000010000001000001010100110101111101001110
010001010100010101000100010001010100010000100000001010000001000000101111011011000110100101100010001011
110111100000111000001101100101111100110110001101000010110101101100011010010110111001110101011110000010
110101100111011011100111010100101111011011000110010000101101011011000110100101101110011101010111100000
1011010111100000011100000110110001011010011011000110100...
```

preprocessing

compiling

assembling

linking

compiling

decompiling

reverse engineering

```
011111110100010101001100010001100000001000000001000000010000000000000000000000000000000000000000
000000000000000000000000000000010000000000111100000000000000000010000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000101000000000000100000000000000000000000000000000000000000000000000000000000000000
000000000100000000000000000000000000000000000000000000100000000000000000000101000000000000000000001000000
000101010101001000100010011110010101001000100000111110110000010000000110001110000000100010011100011101 00
100010111110000000000000000000000000000000000000000000000000000000000000010110000000000000001110100000
000000000000000000000000000000001001000101111110000000000000000000000000000000000000000000000000000
00000000001001000...0111111101000101010011000100011000000010000000010000000100000000000000000000000000
000000000000000000000000000000000000000000011000000000011110000000000000000001000000000000000000
000000011000000000011110000000000000000000000000000000000000000000000010000000000000000000000000000
000000000000000000000000000000000000010100000110010000000000000000000000000000000000000000000000000
000000000000000000000000000001000000000000000000111000000000000000000111000000000010000000000000000000011100000
000000001100100000000000000000010000000000000000000000000000000000010100000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000010111000001001010000000000000000
000000000000000000000000000000000...00101111011011000110100101100010011000110010111001110011011011 11
001011100011011000100000001011110111010101110011011100100010111101101100011010010110001000101111011110
000011100000110110010111110011011000110100001011010110110001101001011011100111010101111000001011010110
011101101110011101010010111101101100011010010110001001100011010111110110111001101111011011100111001101
101000011000010111001001100101011001000010111001100001001000000010000000100000101010011010111110100111 0
0100010101000101010001000100010101000100001000000010100000100000000101111011011000110100101100010001011
110111110000011100000110110010111110011011000110100001011010110110001101001011011100111010101111000001 0
1101011001110110111001110101001011110110110001100100001011010110110001101001011011100111010101111 00000
1011010111100000111000001101100010110100110110001101 00...
```

debugging

COMO HOPPE

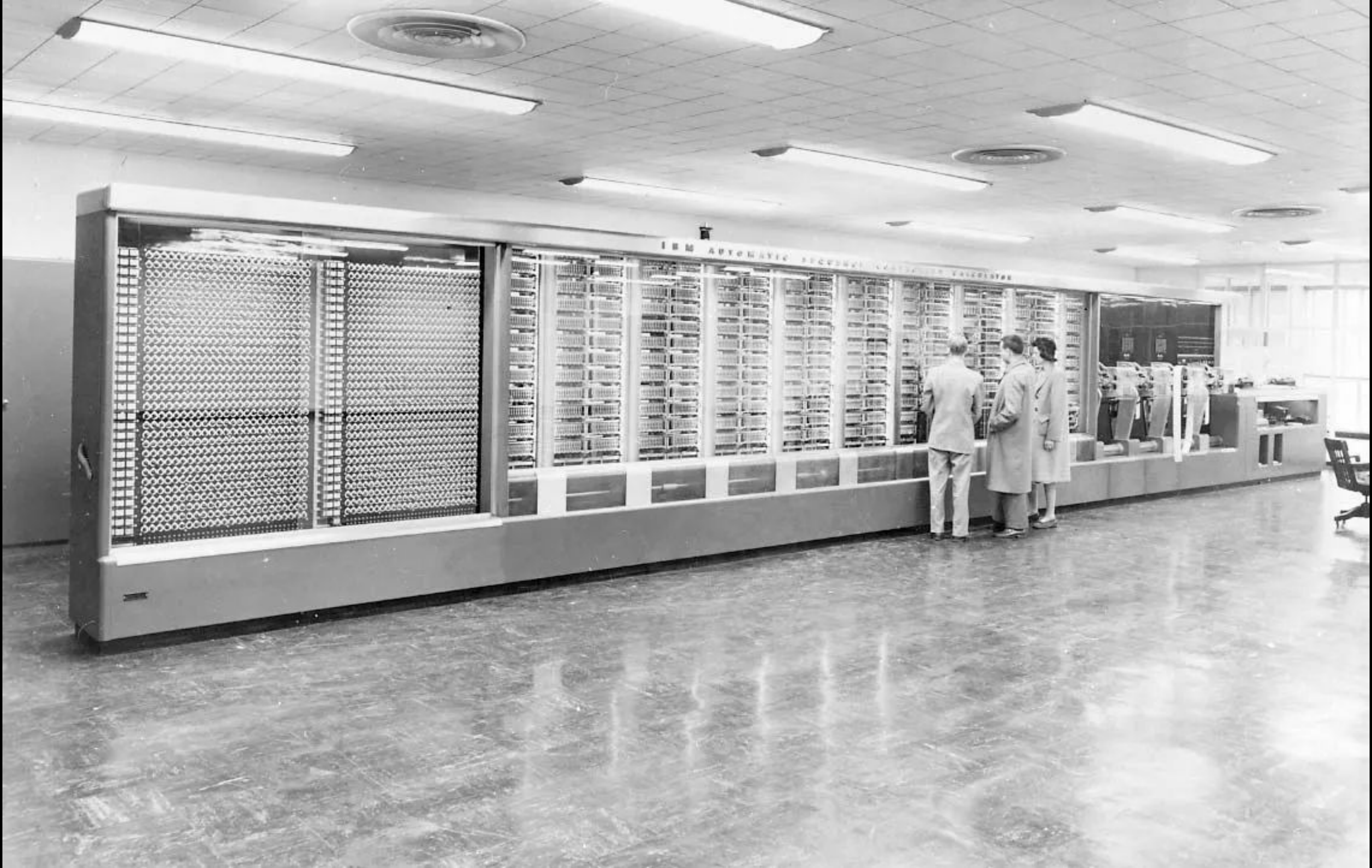
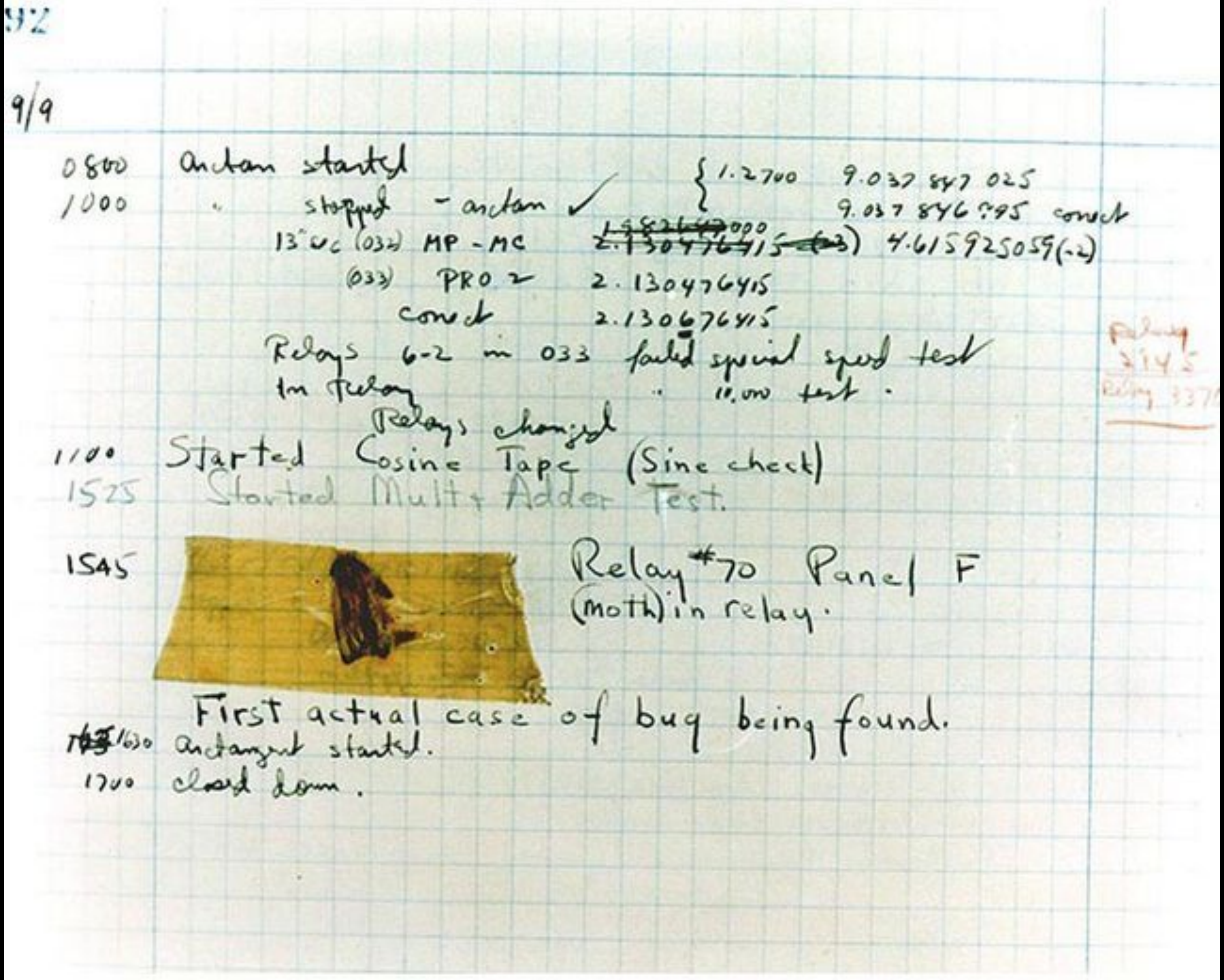

IBM AUTOMATIC SEQUENCE CONTROLLED CALCULATOR

Photo # NH 96566-KN (Color) First Computer "Bug", 1947



92

9/9

0800 arctan started

1000 " stopped - arctan ✓ { 1.2700 9.037 847 025
9.037 846 995 correct

13" uc (032) MP - MC ~~1.982147000~~ ~~2.130476415 (-3)~~ 4.615925059(-2)

(033) PRO 2 2.130476415

correct 2.130676415

Relays 6-2 in 033 failed special speed test

In relay " 10.000 test.

Relays changed

1100 Started Cosine Tape (Sine check)

1525 Started Mult + Adder Test.

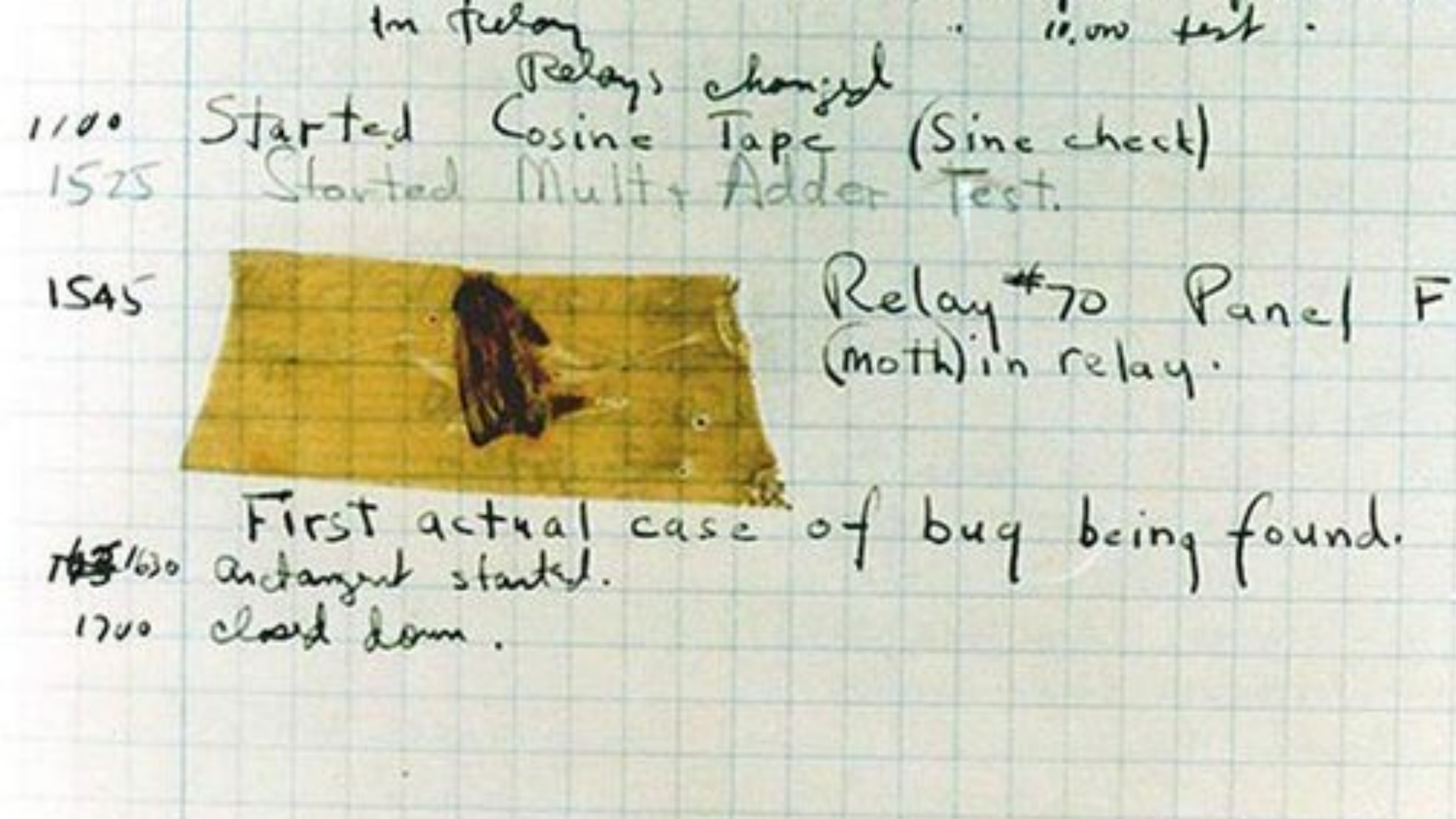
1545 Relay #70 Panel F
(moth) in relay.

First actual case of bug being found.

~~143~~ 1630 arctangent started.

1700 closed down.

Relay 2145
Relay 3376



in relay . 11.00 test .

Relays changed

1100 Started Cosine Tape (Sine check)

1525 Started Mult + Adder Test.

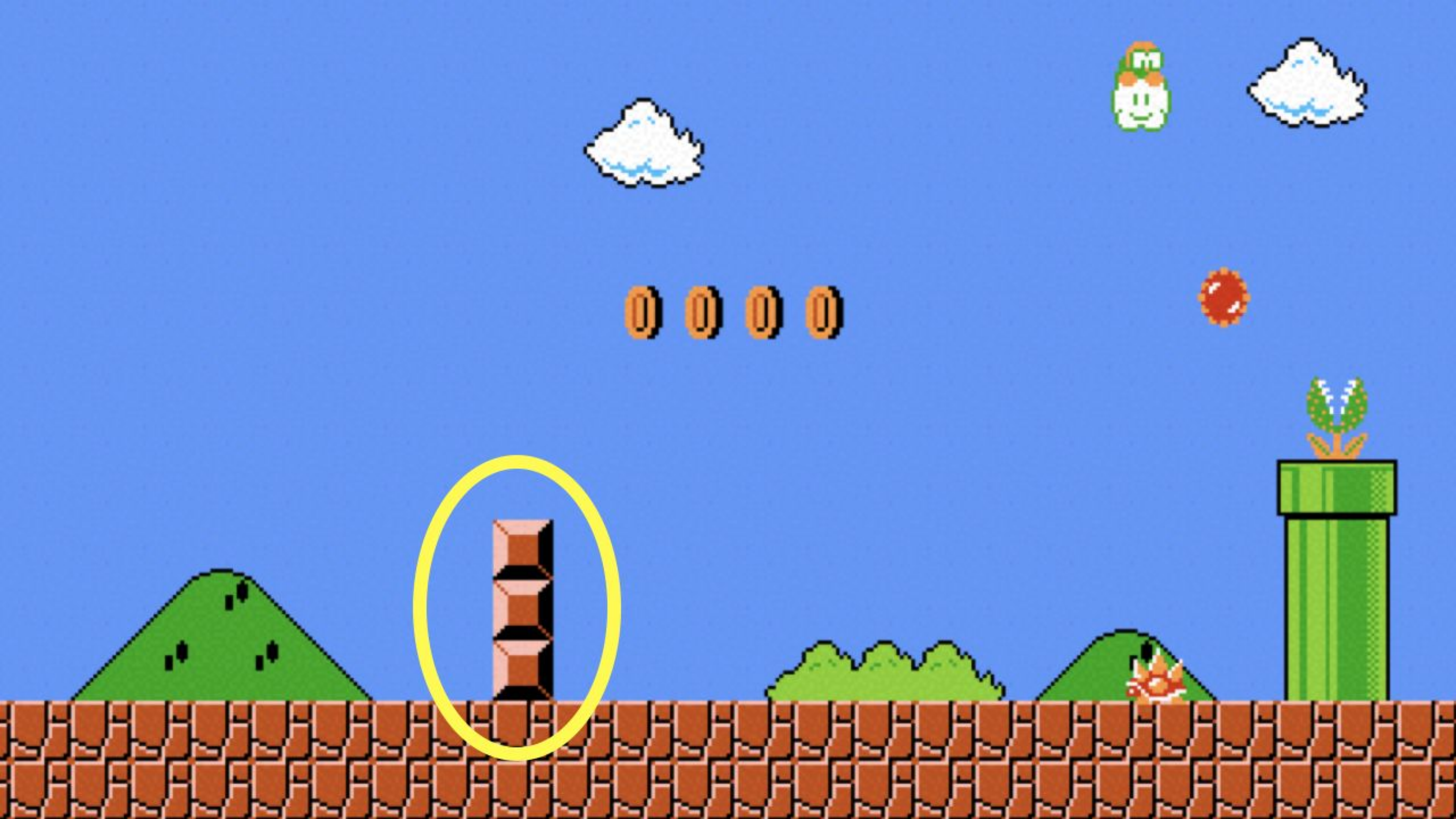
1545 Relay #70 Panel F

(moth) in relay.

First actual case of bug being found.

~~1600~~ 1630 antangent started.

1700 closed down.



```
printf
```

`printf`

debug50

`printf`

`debug50`

rubber duck

rubber duck debugging

`printf`

`debug50`

rubber duck

cs50.ai

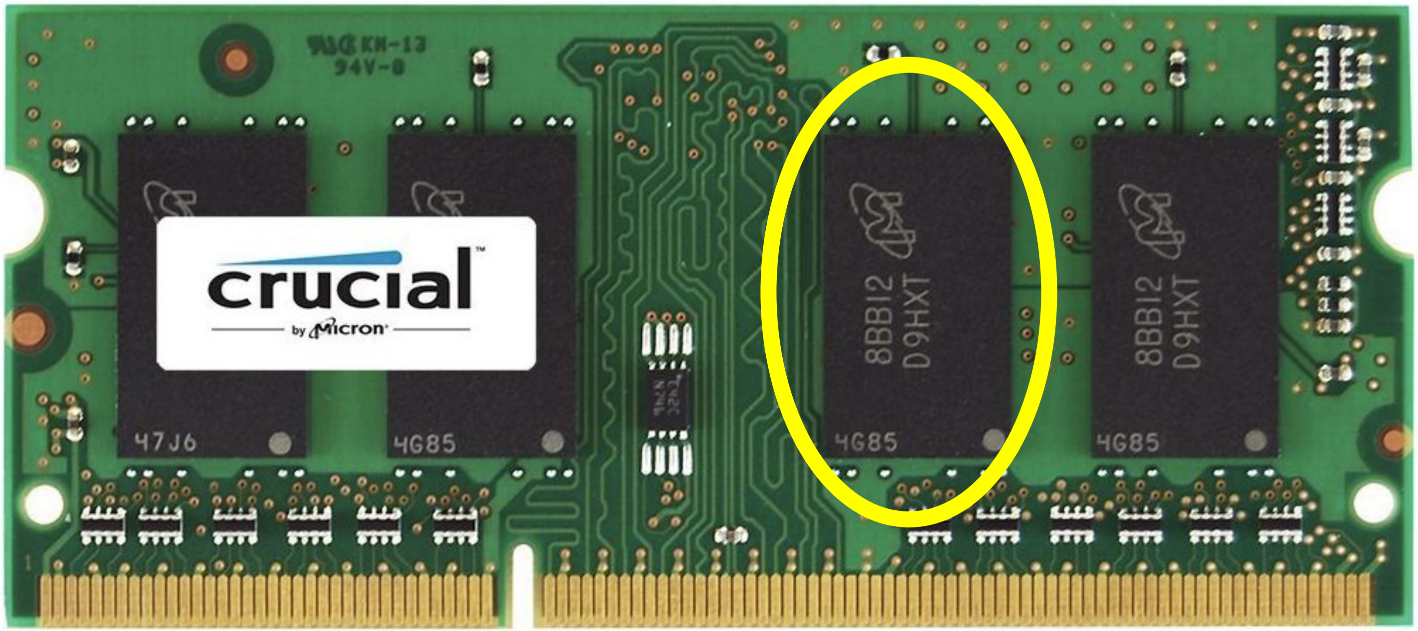
types

```
bool

int

long

float

double

char

string

...
```

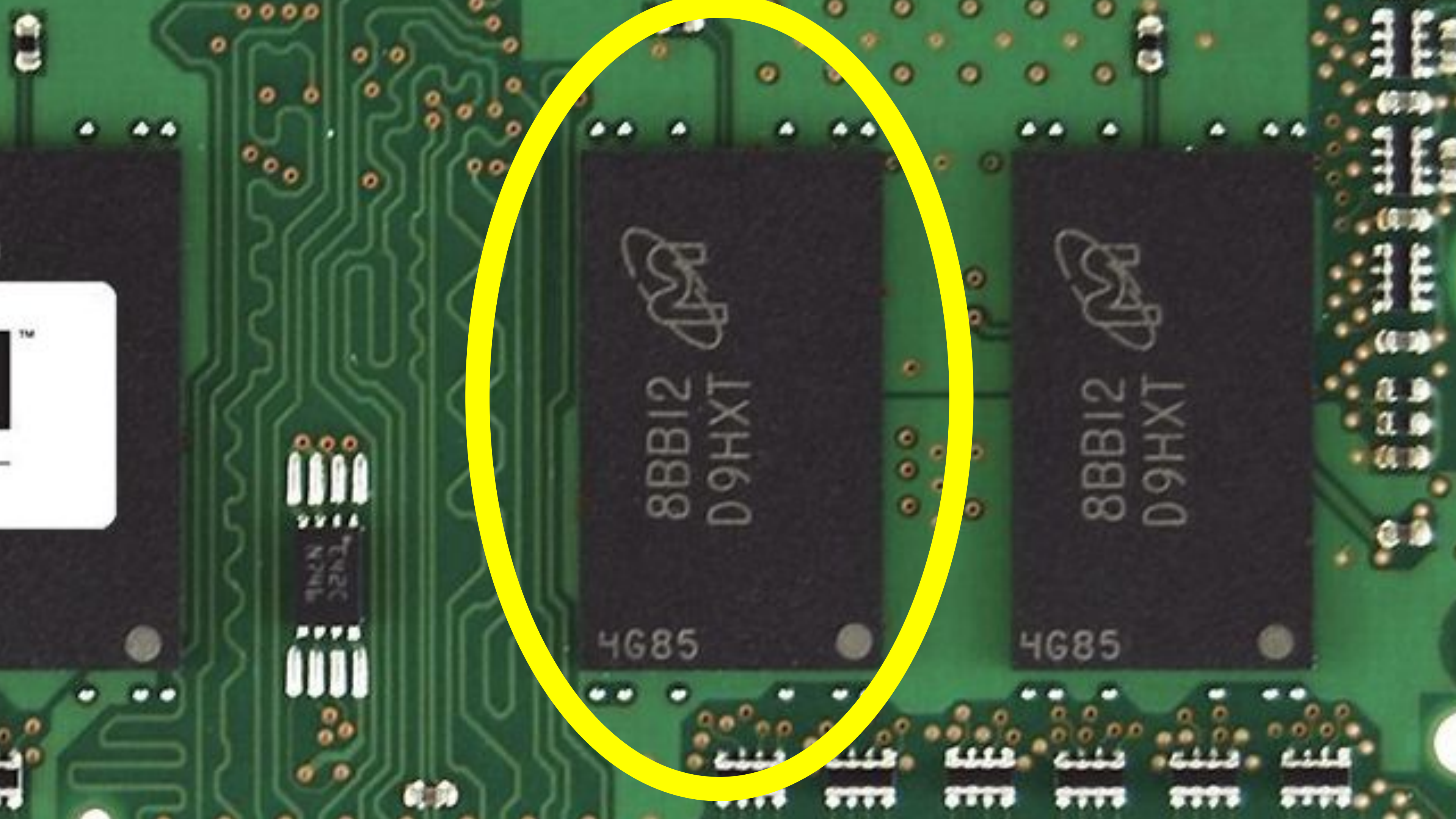
`bool`	1 byte
`int`	4 bytes
`long`	8 bytes
`float`	4 bytes
`double`	8 bytes
`char`	1 byte
`string`	? bytes
`...`	

KN-13
94V-0
crucial™
by Micron
47J6
4G85
8BBI2
D9HXT
4G85
8BBI2
D9HXT
4G85
1

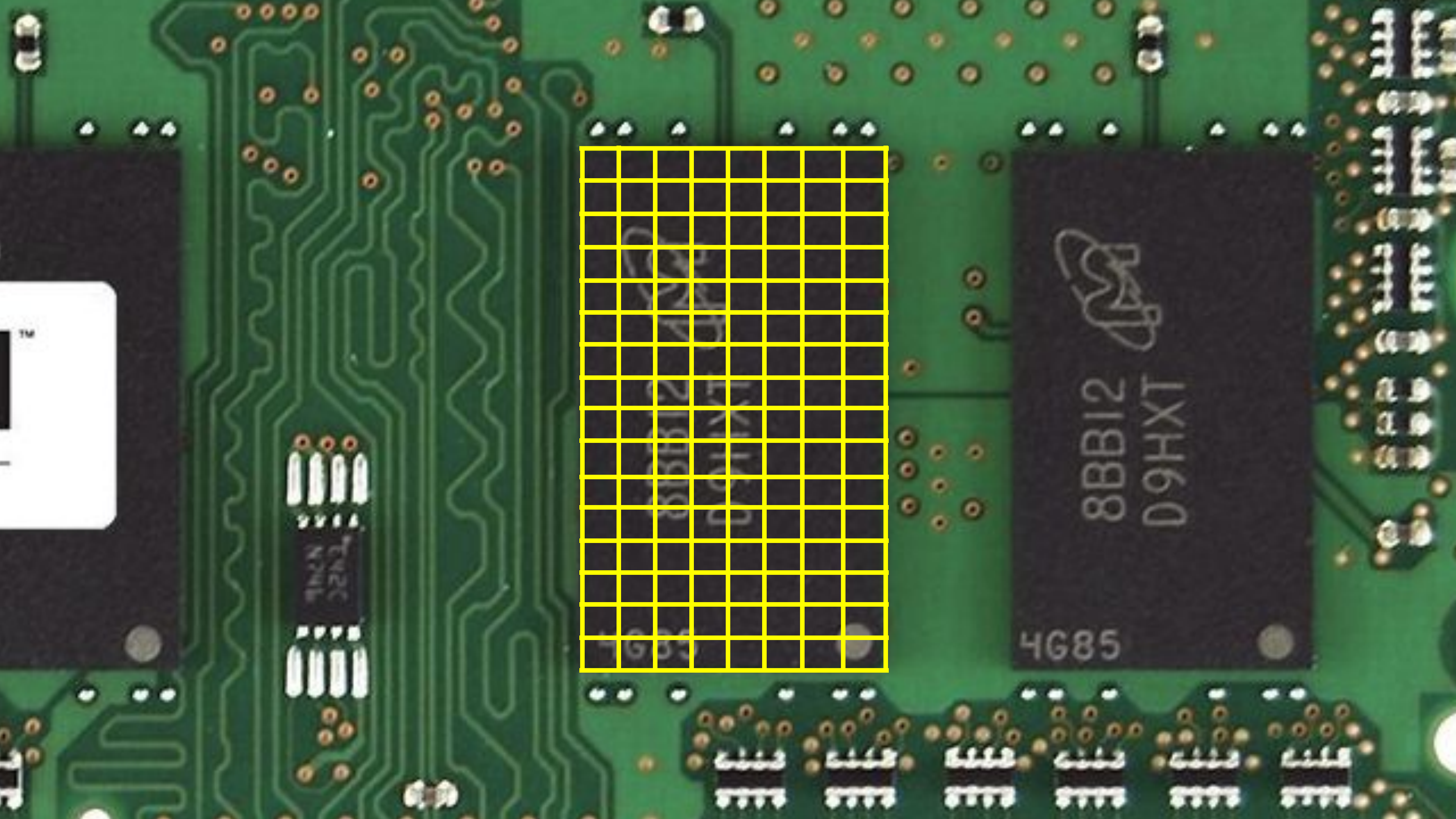
amazon.com

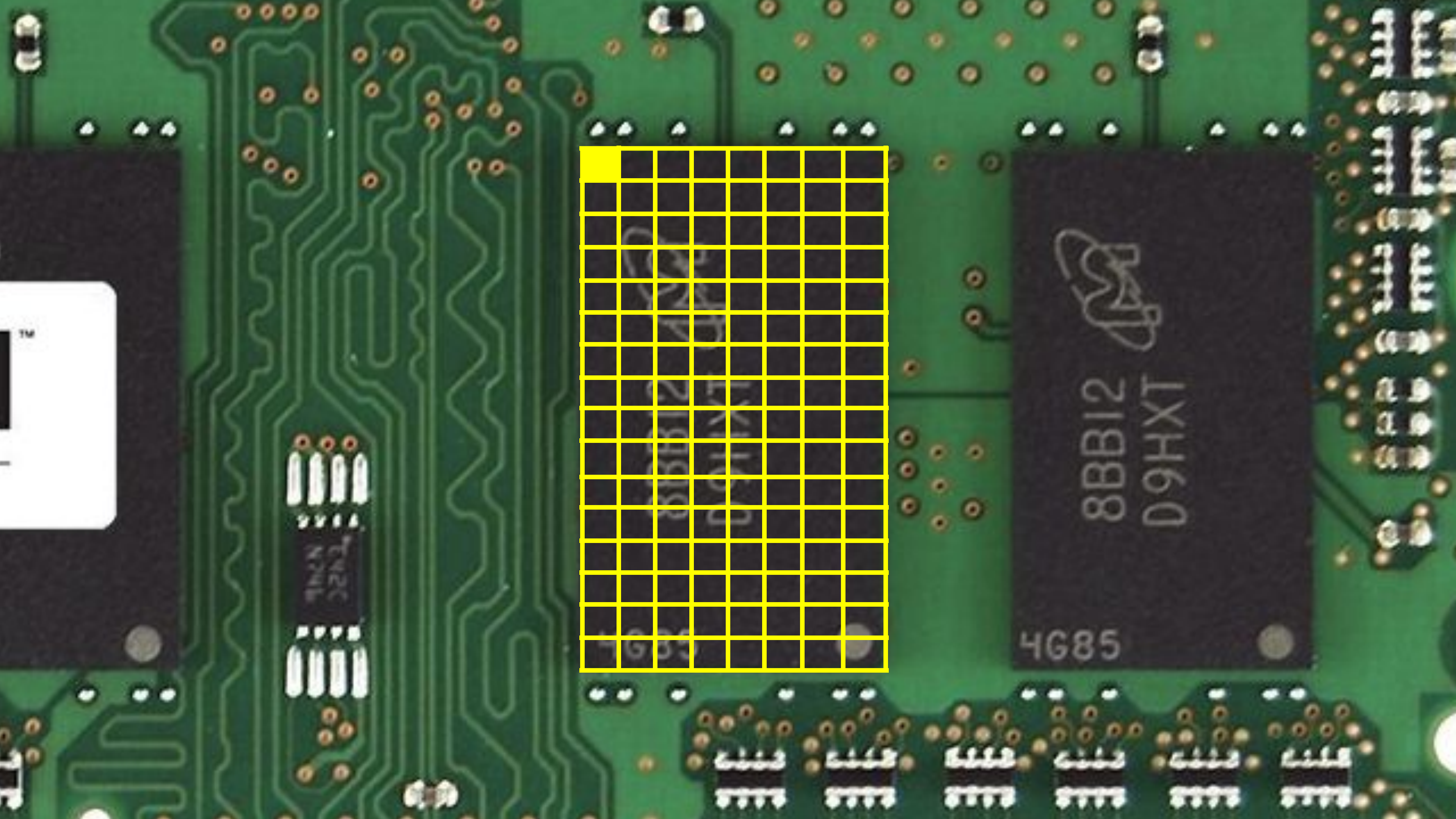

KN-13
94V-0
crucial
by Micron
47J6
4G85
8BBI2
D9HXT
4G85
8BBI2
D9HXT
4G85

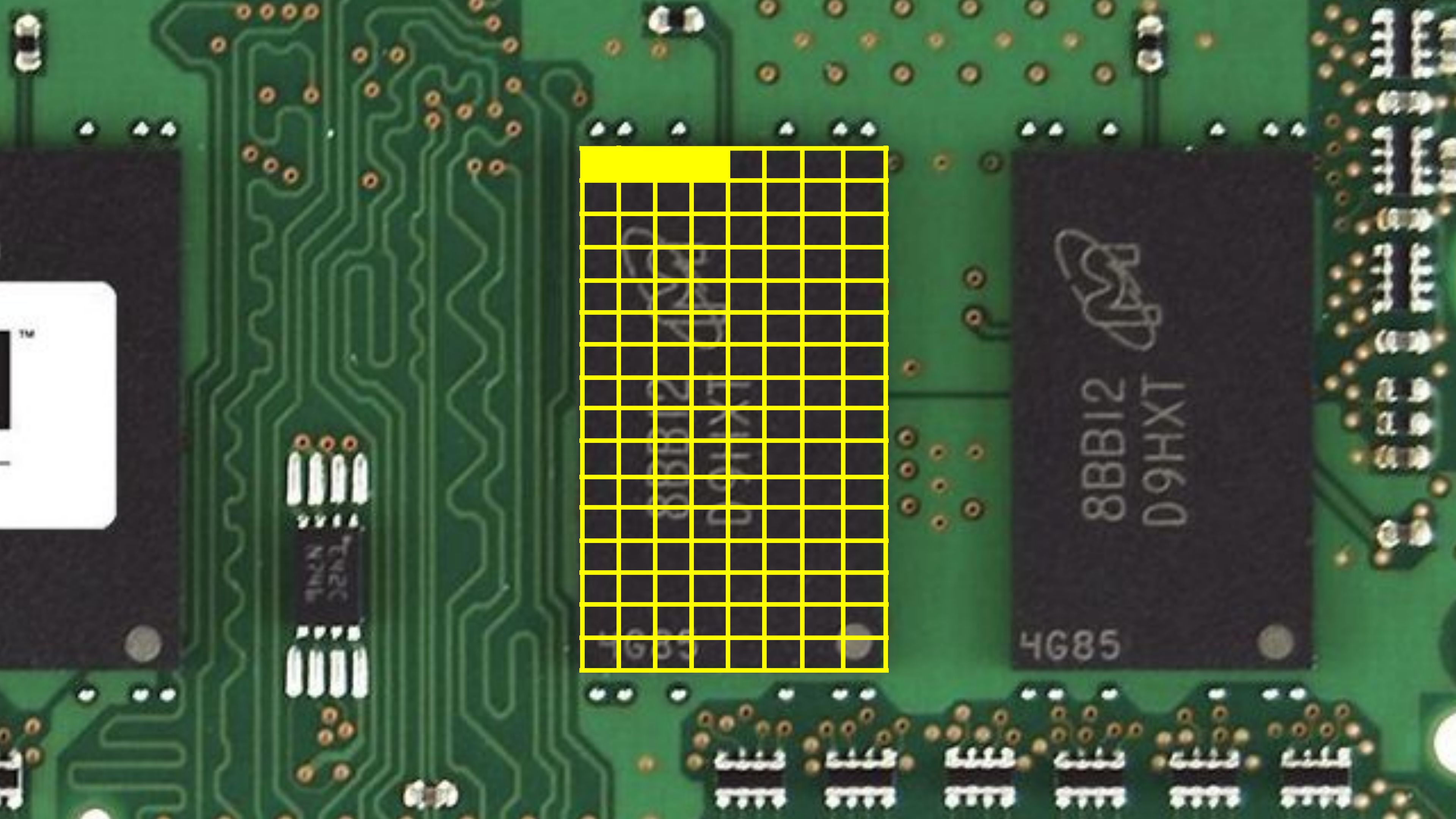
amazon.com


8BBI2
D9HXT
4G85
8BBI2
D9HXT
4G85

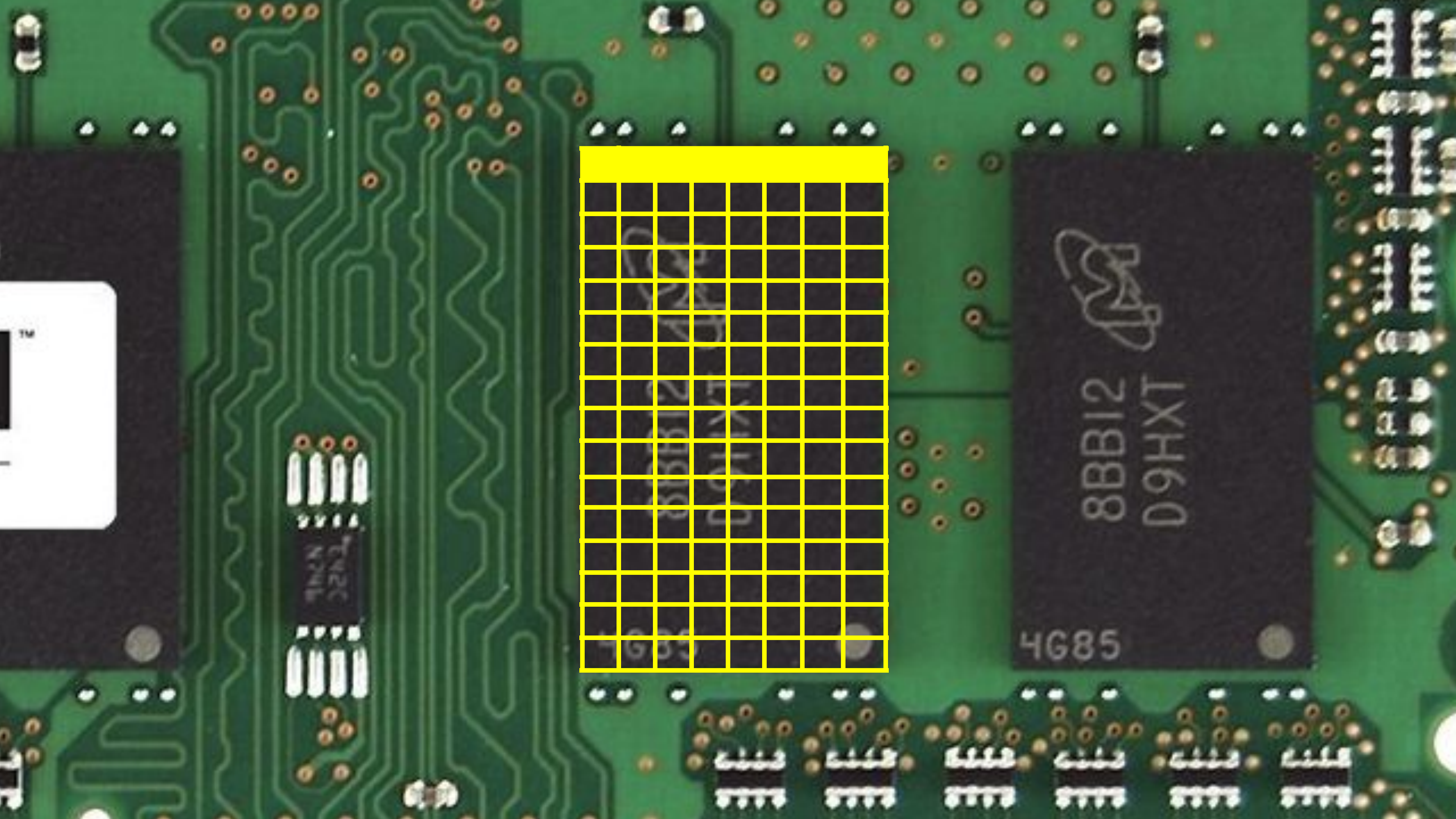
8BB12
D9HXT
4G85
8BB12
D9HXT
4G85

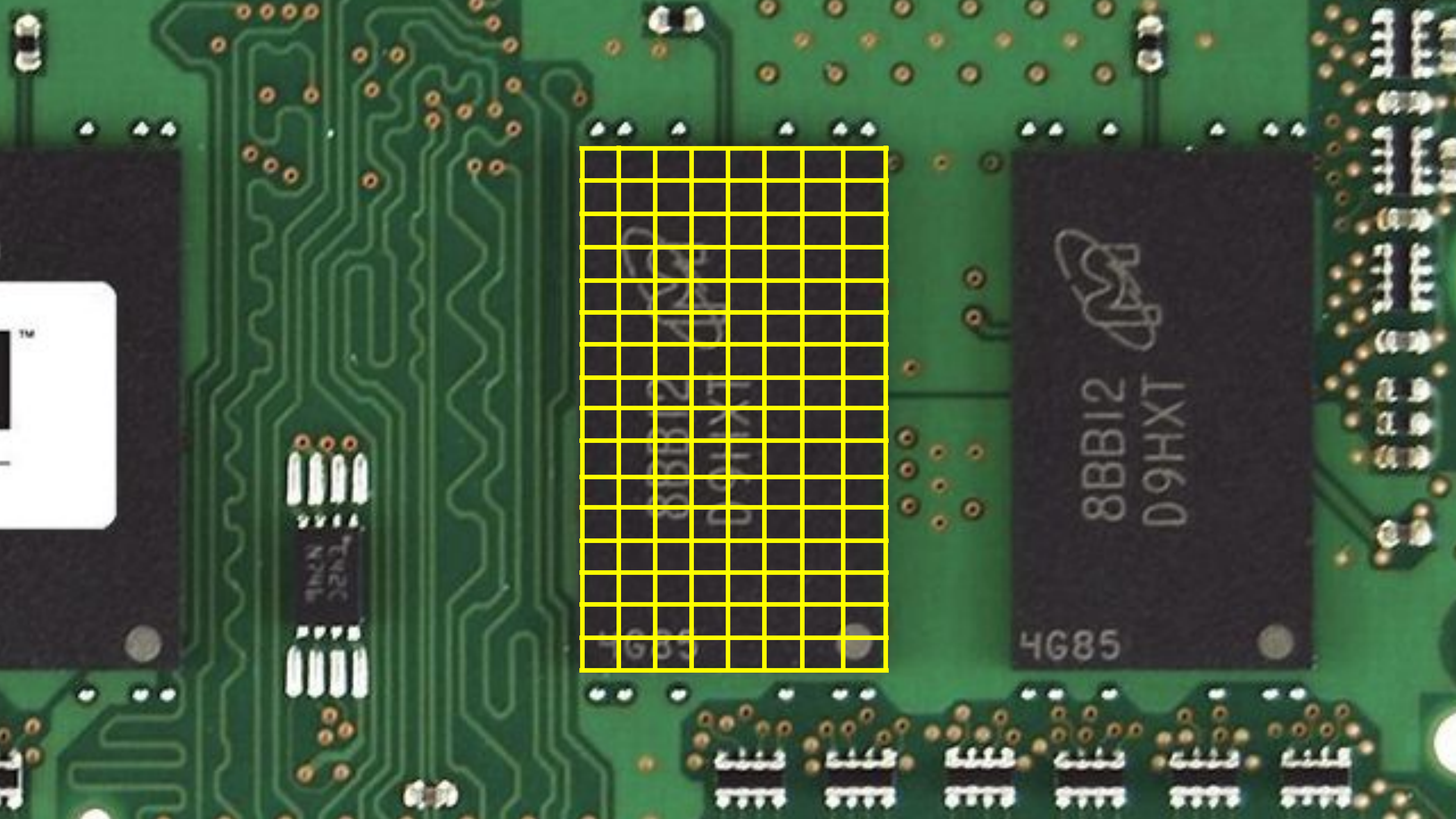

8BB12
D9HXT
4G85
8BB12
D9HXT
4G85


8BB12
D9HXT
4G85
8BB12
D9HXT
4G85



™
8BB12
D9HXT
4G85
8BB12
D9HXT
4G85


8BB12
D9HXT
4G85
8BB12
D9HXT
4G85


8BB12
D9HXT
4G85
8BB12
D9HXT
4G85
™

```
int score1 = 72;

int score2 = 73;

int score3 = 33;
```

72

score1

72

score1

73

score2

72

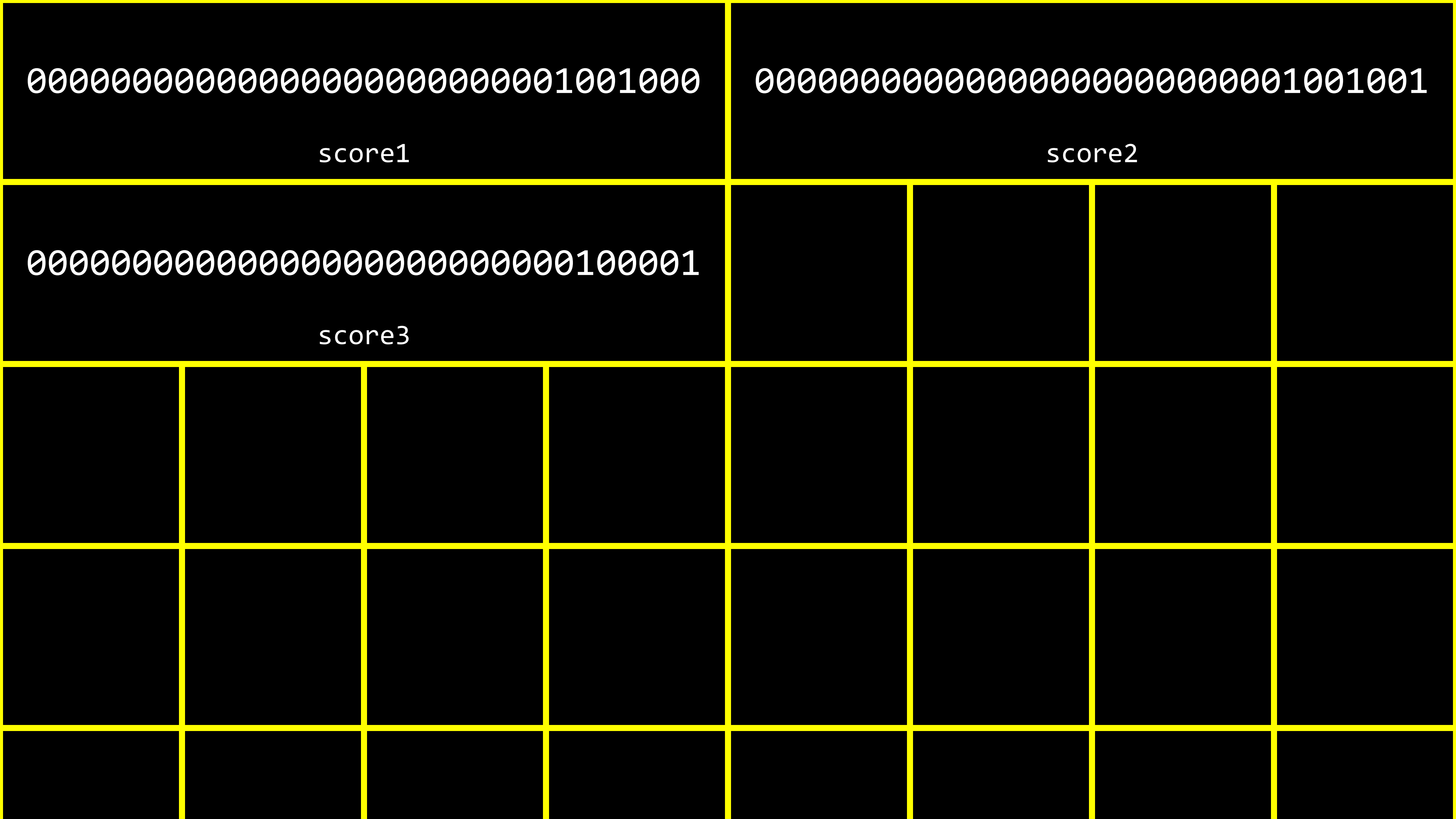
score1

73

score2

33

score3



00000000000000000000000001001000

score1

00000000000000000000000001001001

score2

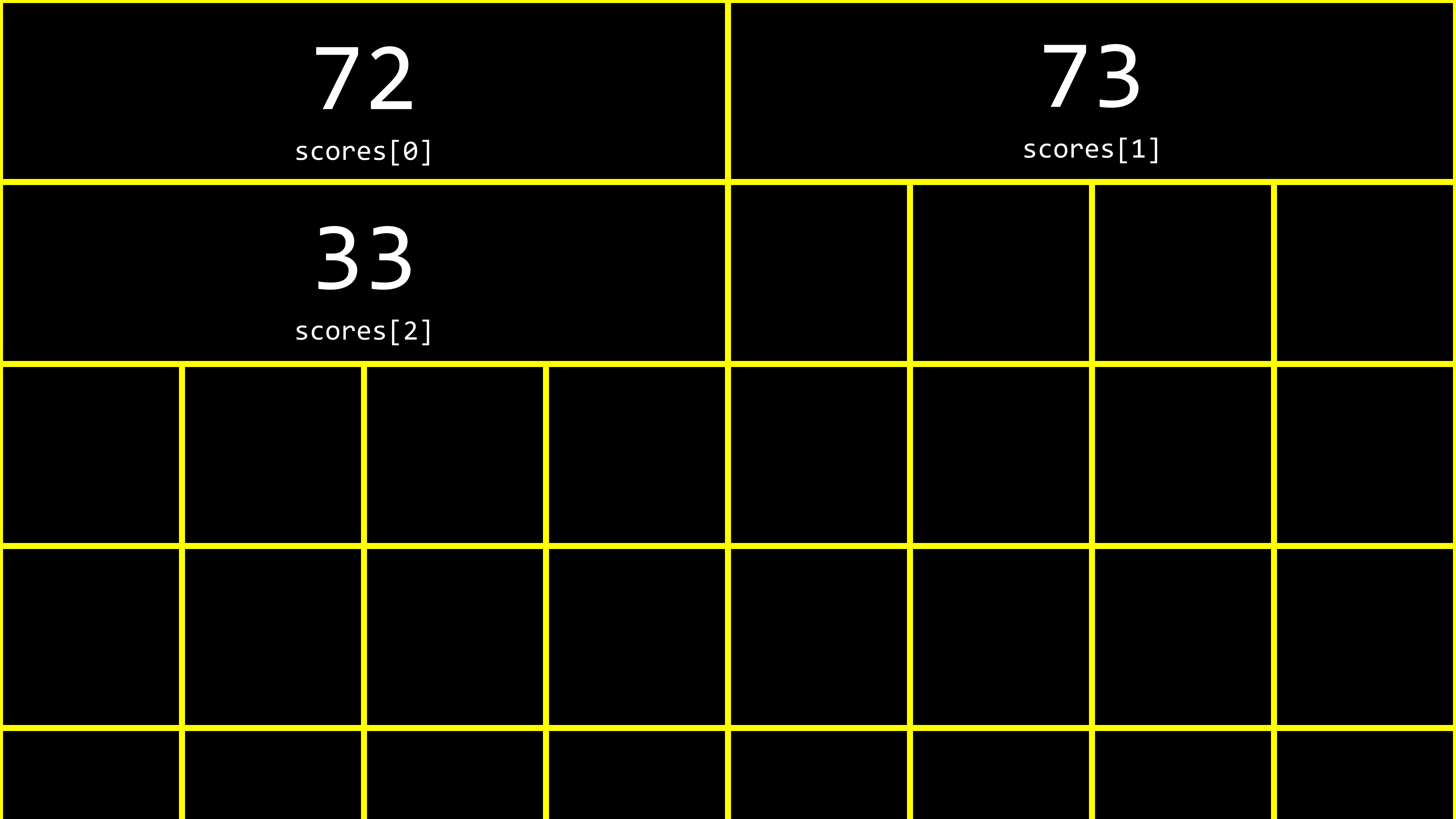
00000000000000000000000000100001

score3

```
int score1 = 72;

int score2 = 73;

int score3 = 33;
```

arrays

```
int scores[3];
int scores[3];

scores[0] = 72;

scores[1] = 73;

scores[2] = 33;
```



72

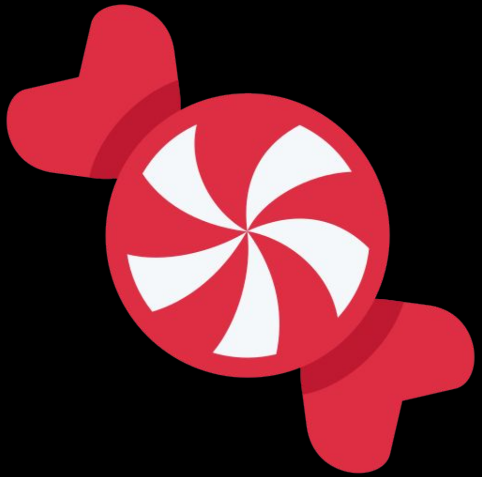
scores[0]

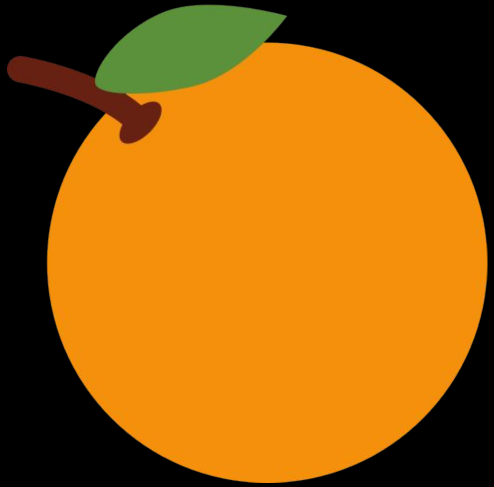
73

scores[1]

33

scores[2]





```
char c1 = 'H';

char c2 = 'I';

char c3 = '!';
```

H c1	I c2	! c3					

72 c1	73 c2	33 c3					

01001000 c1	01001001 c2	00100001 c3					

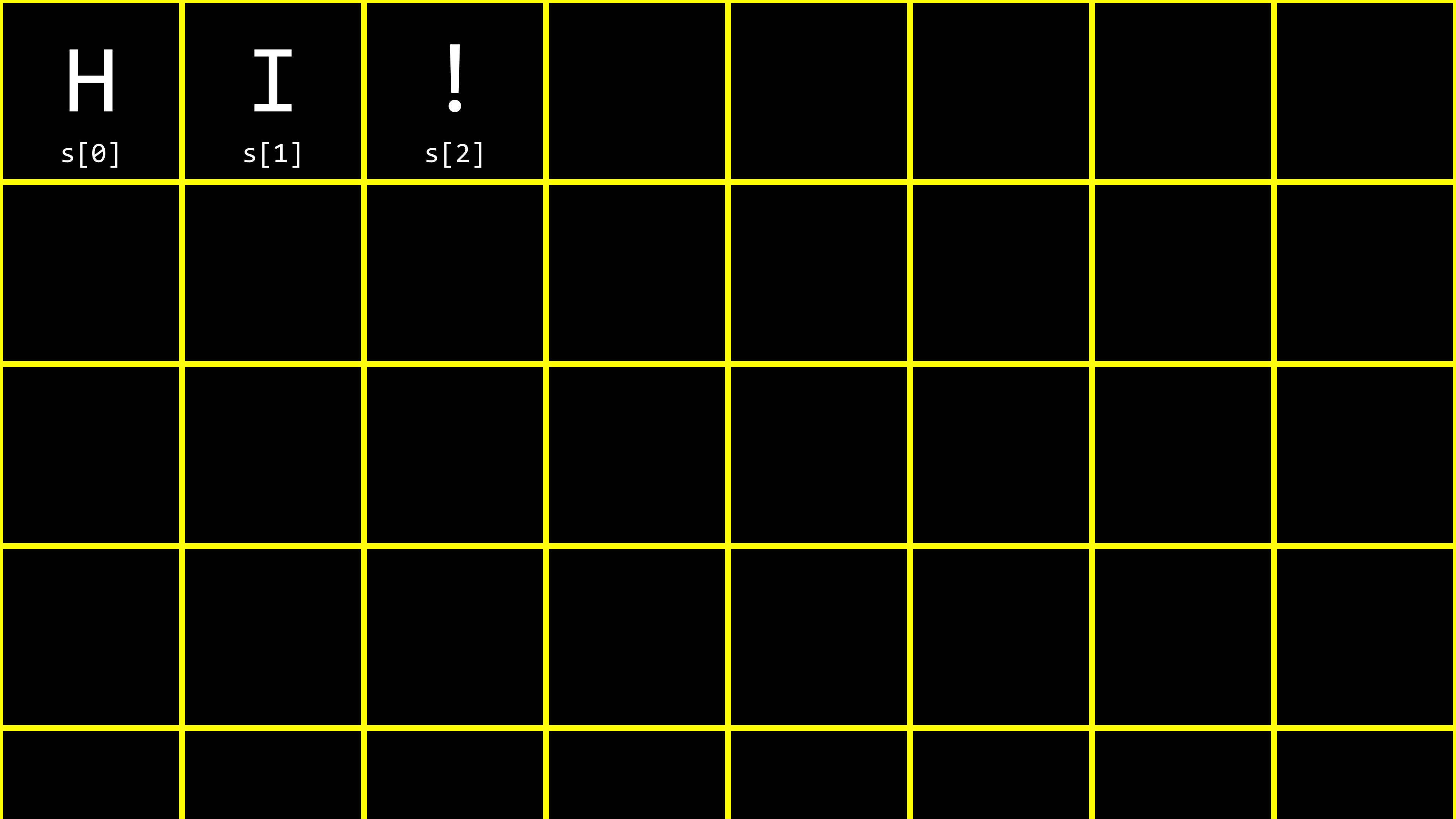
string

string

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

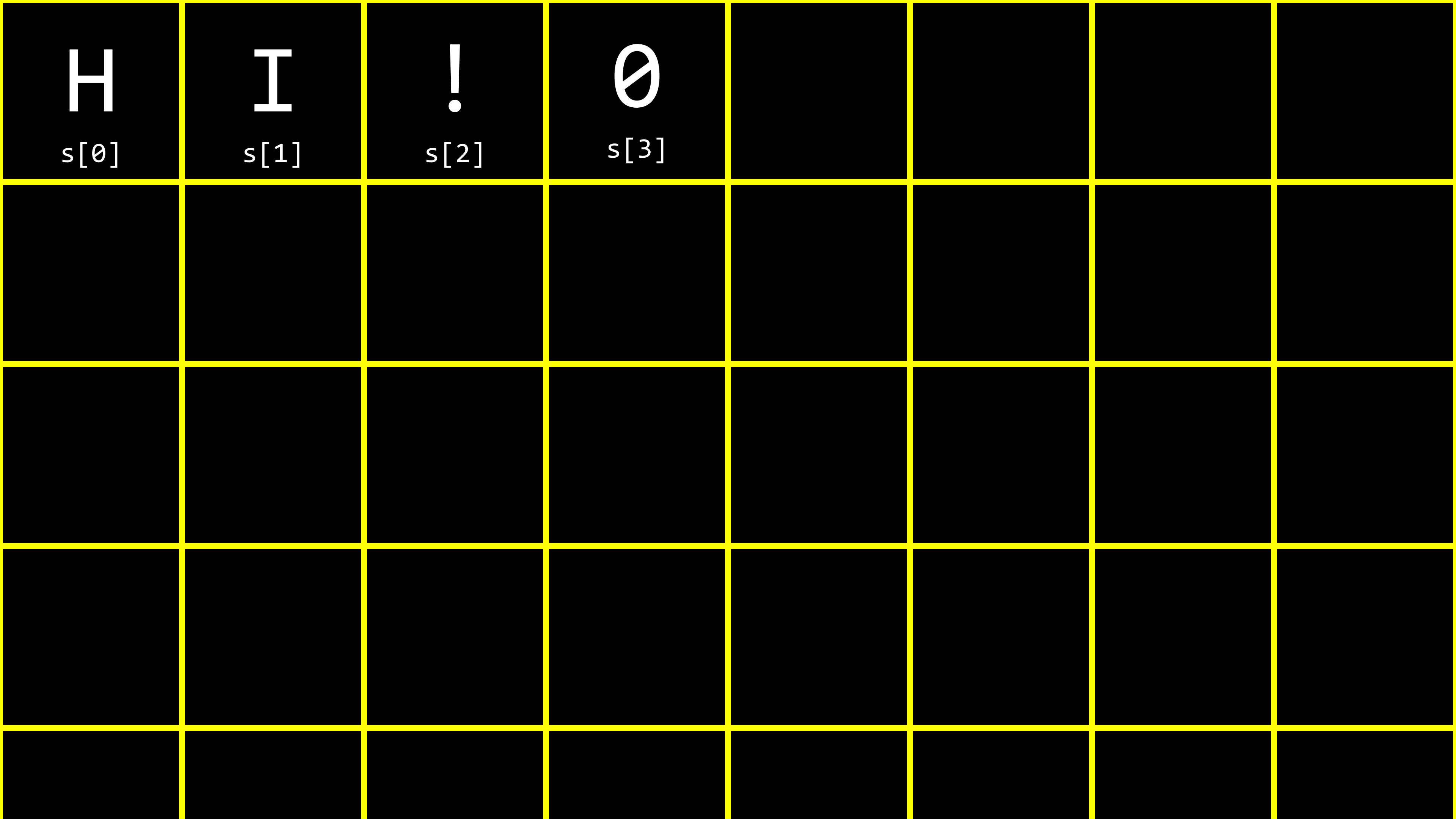
H I !

s



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

H s[0]	I s[1]	! s[2]	00000000 s[3]				

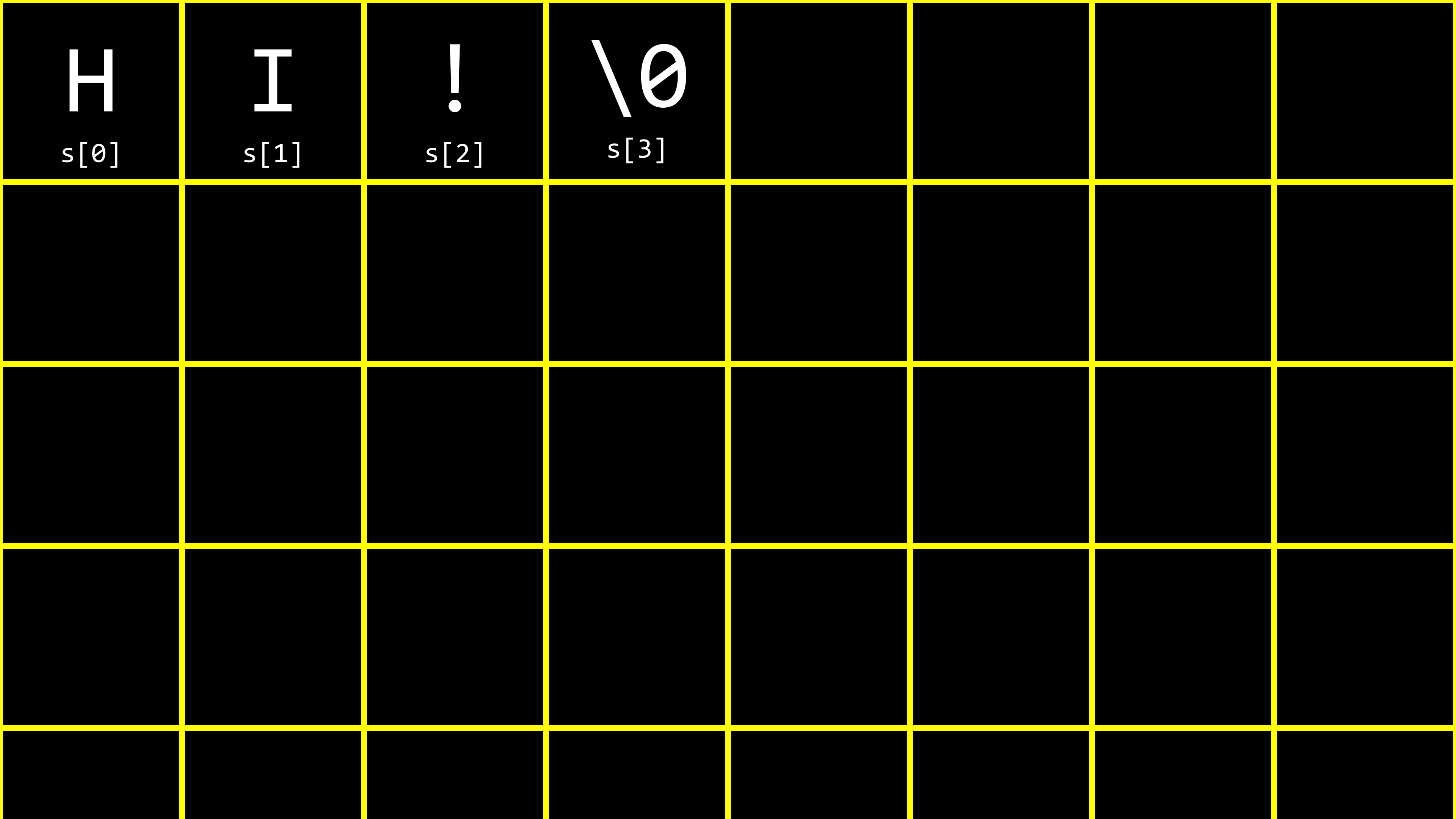


H
s[0]

I
s[1]

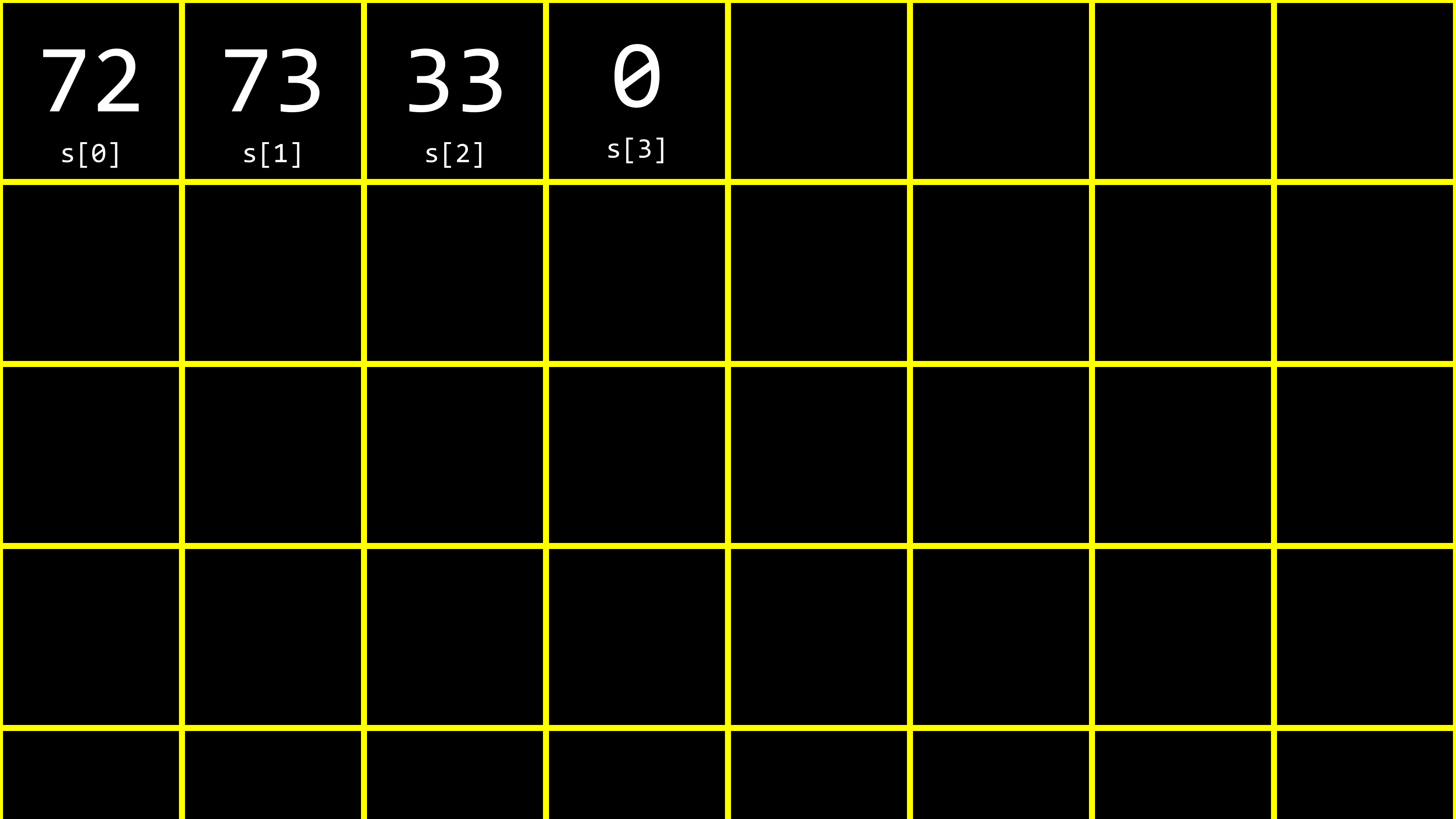
!
s[2]

0
s[3]

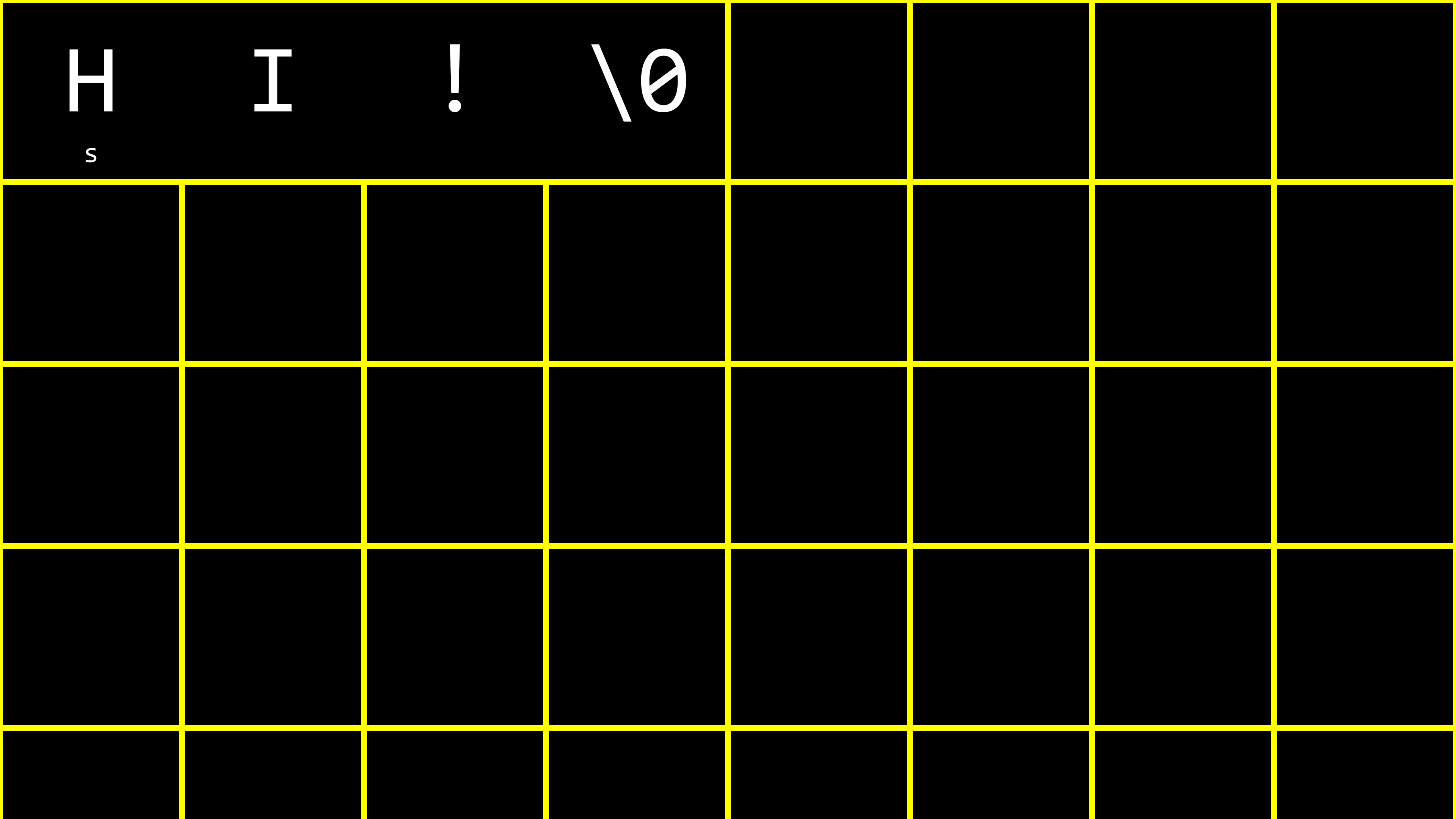


H I ! \0

s[0] s[1] s[2] s[3]



72 s[0]	73 s[1]	33 s[2]	0 s[3]				



H I ! \0

s

NUL

0	NUL	16	DLE	32	SP	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	-	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

0	NUL	16	DLE	32	SP	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	-	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

```
string s = "HI!";

string t = "BYE!";
```

H s	I	!	\0				

H	I	!	\0	B	Y	E	!
s				t			
\0							

H s[0]	I s[1]	! s[2]	\0 s[3]	B t[0]	Y t[1]	E t[2]	! t[3]
\0 t[4]							

```
string words[2];

words[0] = "HI!";

words[1] = "BYE!";
```

H	I	!	\0	B	Y	E	!
words[0]				words[1]			
\0							

H words[0][0]	I words[0][1]	! words[0][2]	\0 words[0][3]	B words[1][0]	Y words[1][1]	E words[1][2]	! words[1][3]
\0 words[1][4]							

string

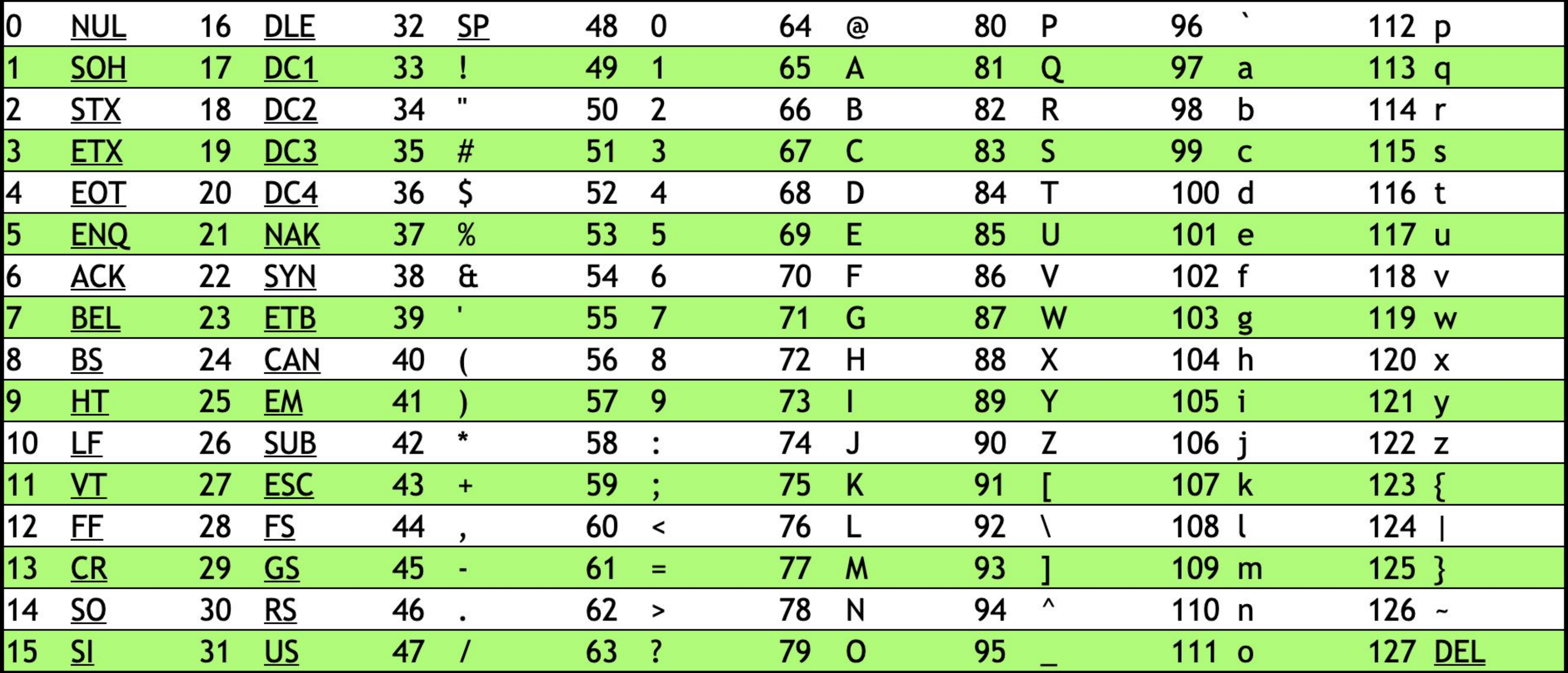
string.h

manual.cs50.io/#string.h

strlen

ctype.h

manual.cs50.io/#ctype.h



0	NUL	16	DLE	32	SP	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	-	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

0	NUL	16	DLE	32	SP	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	-	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

command-line arguments

```
#include <stdio.h>

int main(void)
{
    ...
}
```

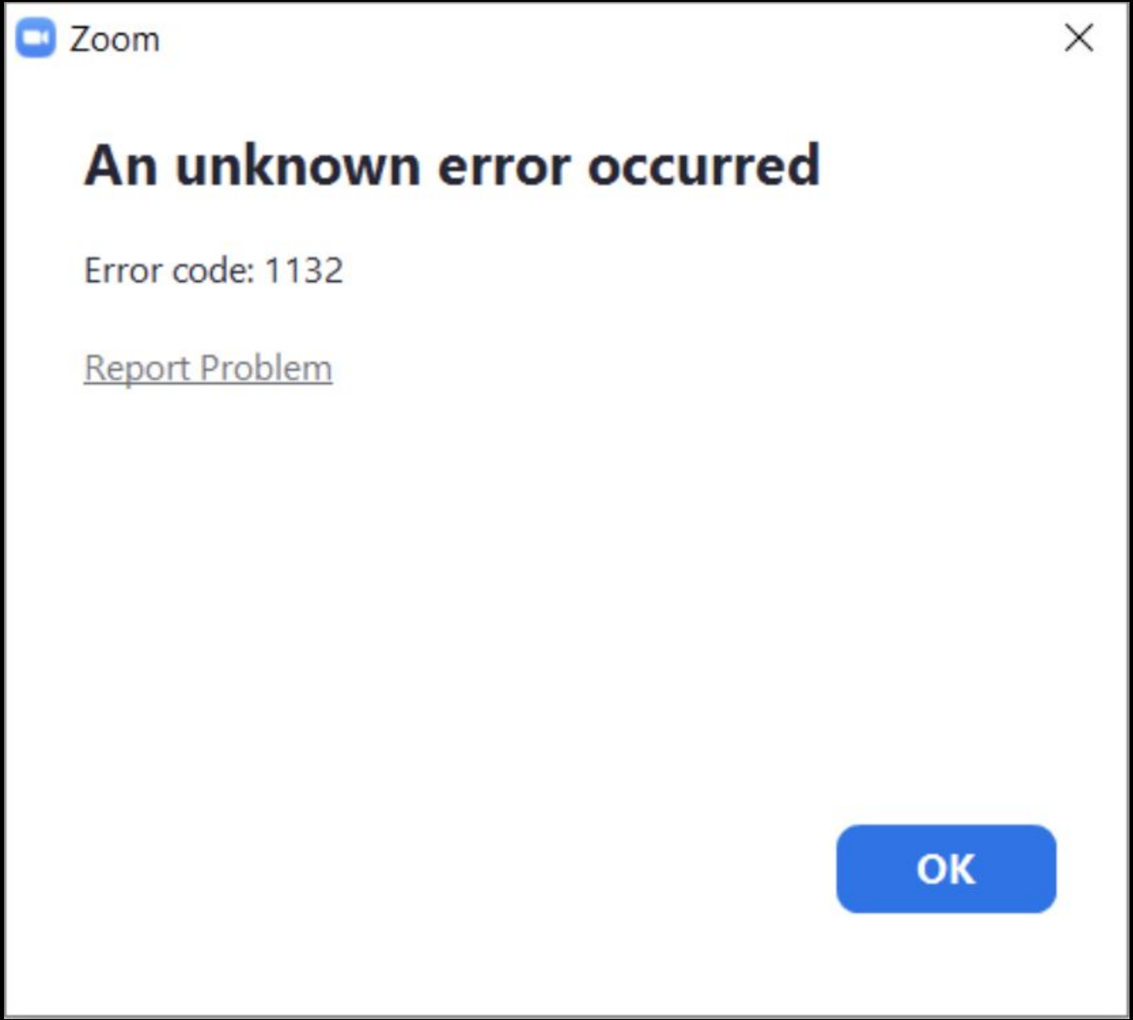
```
#include <stdio.h>

int main(void)
{
    ...
}
```

```
#include <stdio.h>

int main(int argc, string argv[])
{
    ...
}
```

ASCII art

cowsay

exit status



An unknown error occurred

Error code: 1132

404

This is not the web page you are looking for.

```
#include <stdio.h>

int main(int argc, string argv[])
{
    ...
}
```

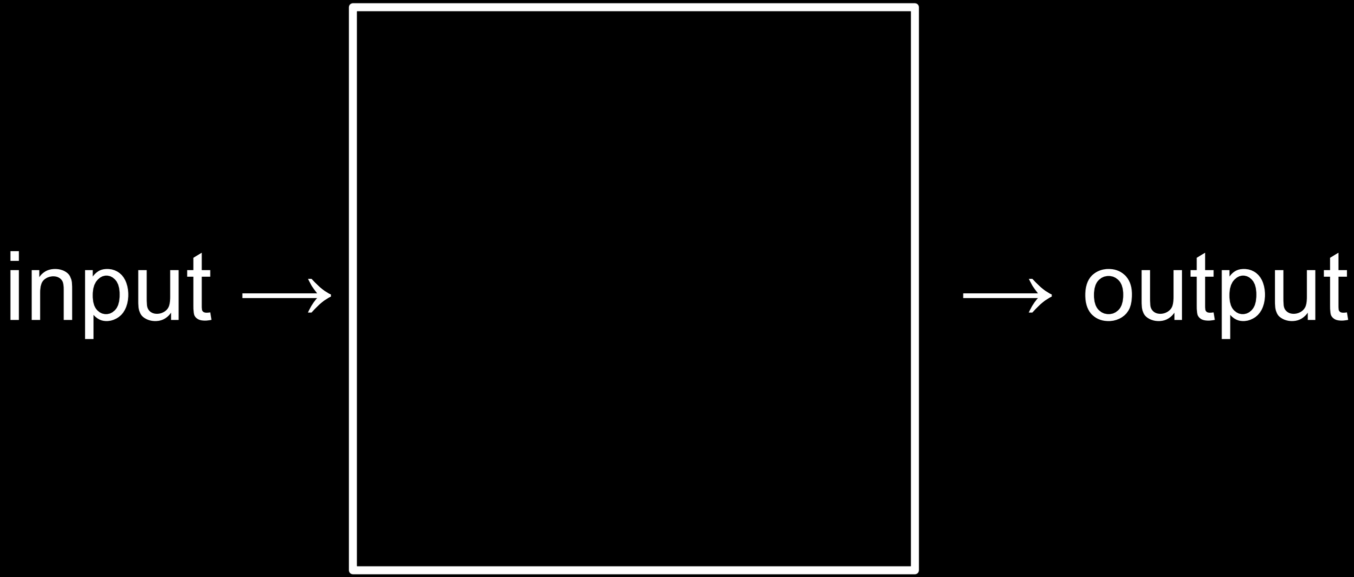
```
#include <stdio.h>

int main(int argc, string argv[])
{
    ...
}
```

```
#include <stdio.h>

int main(void)
{
    ...
}
```

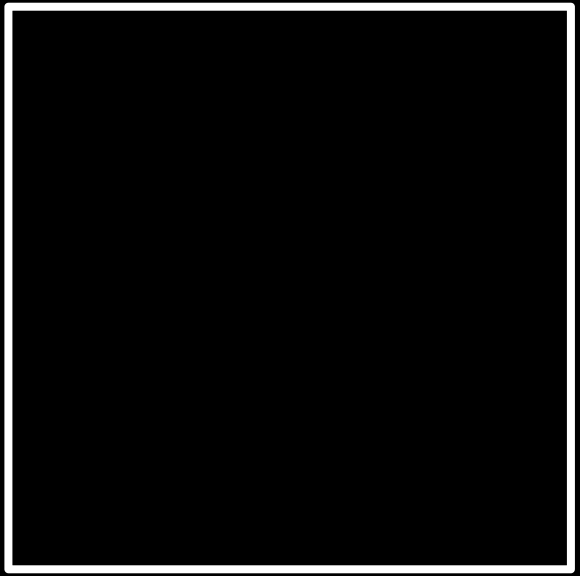
```
echo $?
```

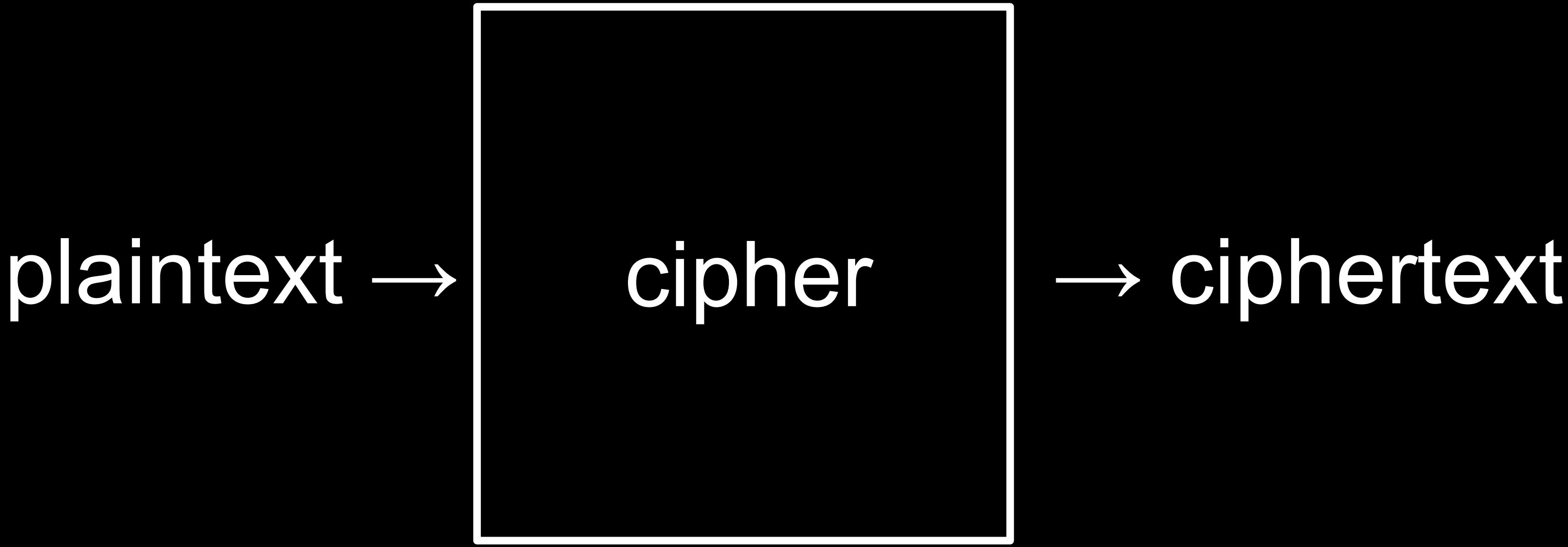
cryptography

encryption

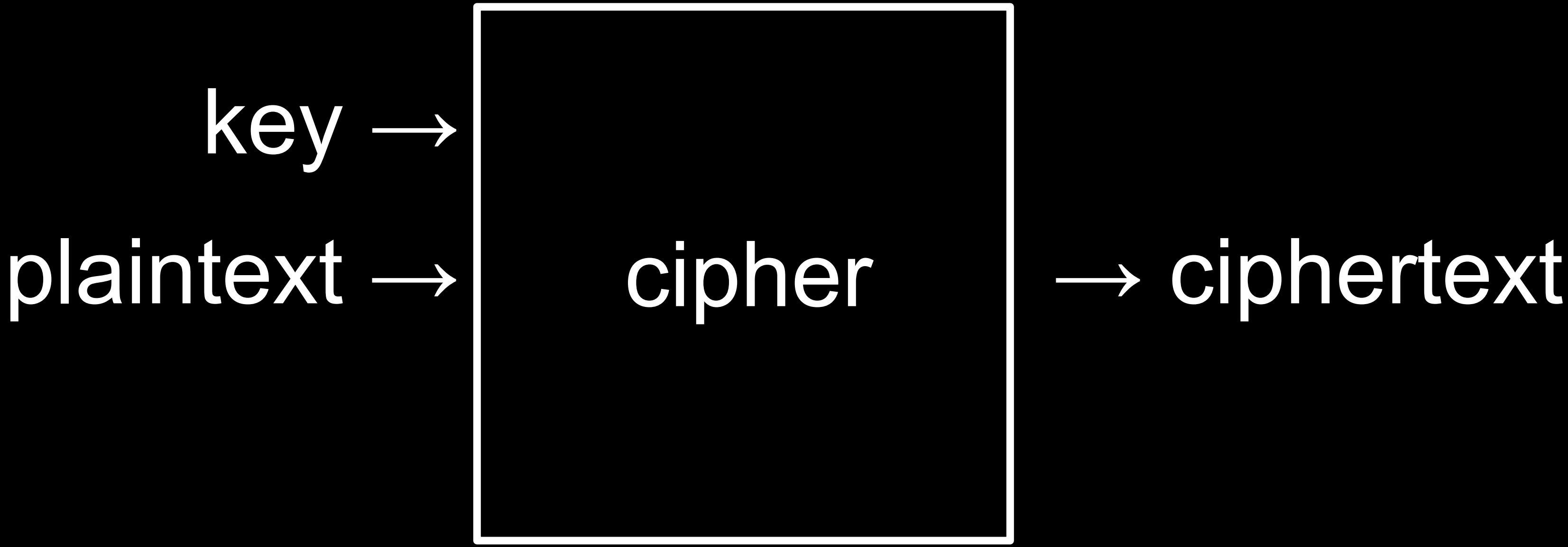


input → → output

plaintext →  → ciphertext



plaintext → cipher → ciphertext

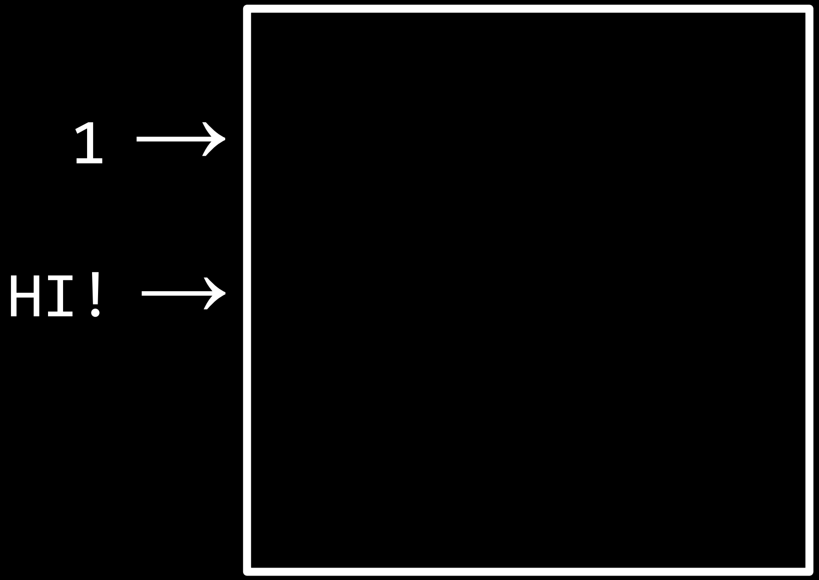


key →

plaintext →

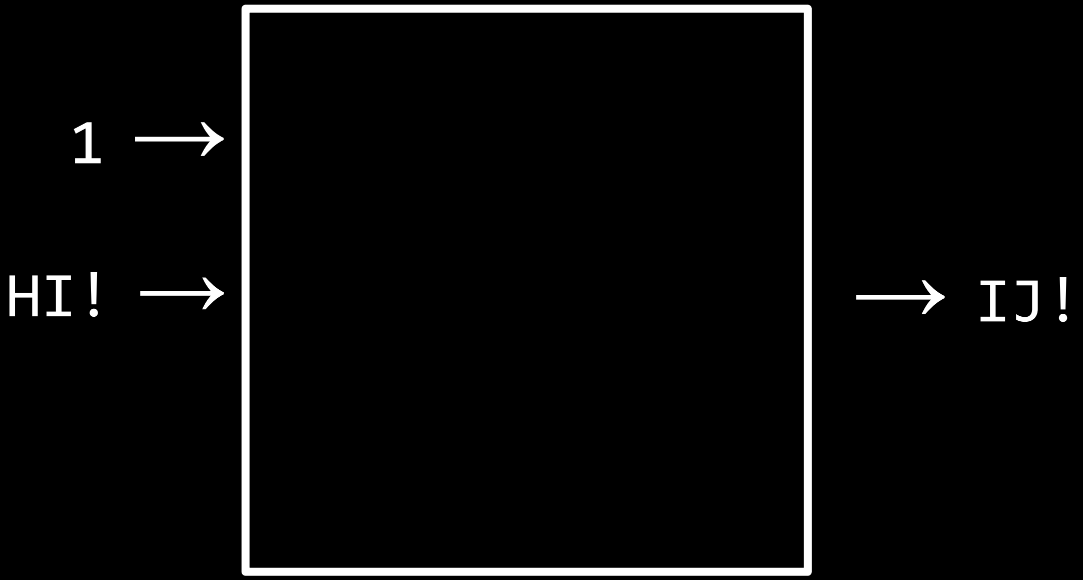
cipher

→ ciphertext



1

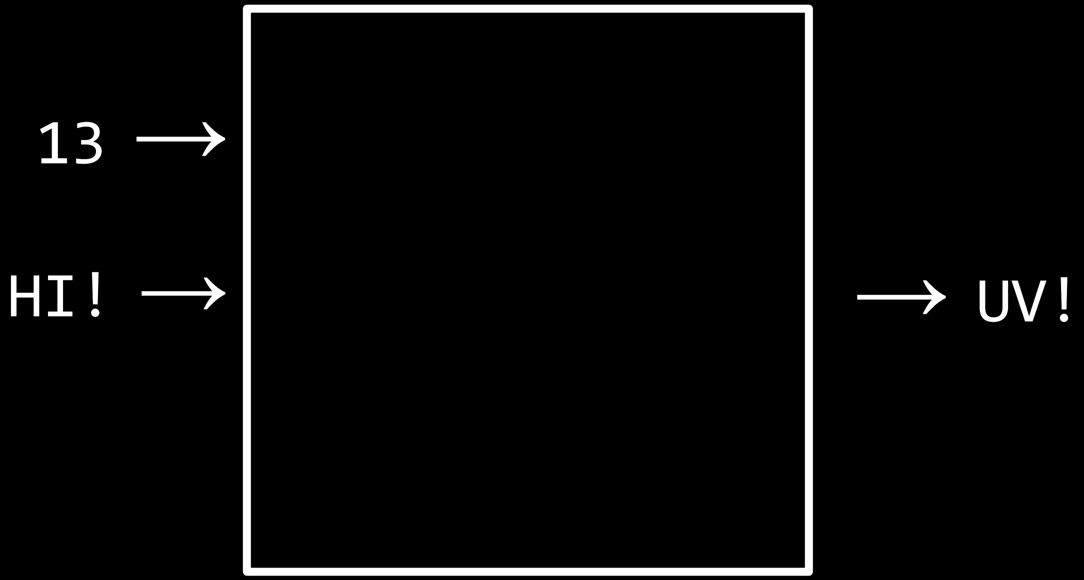
HI!



1

HI!

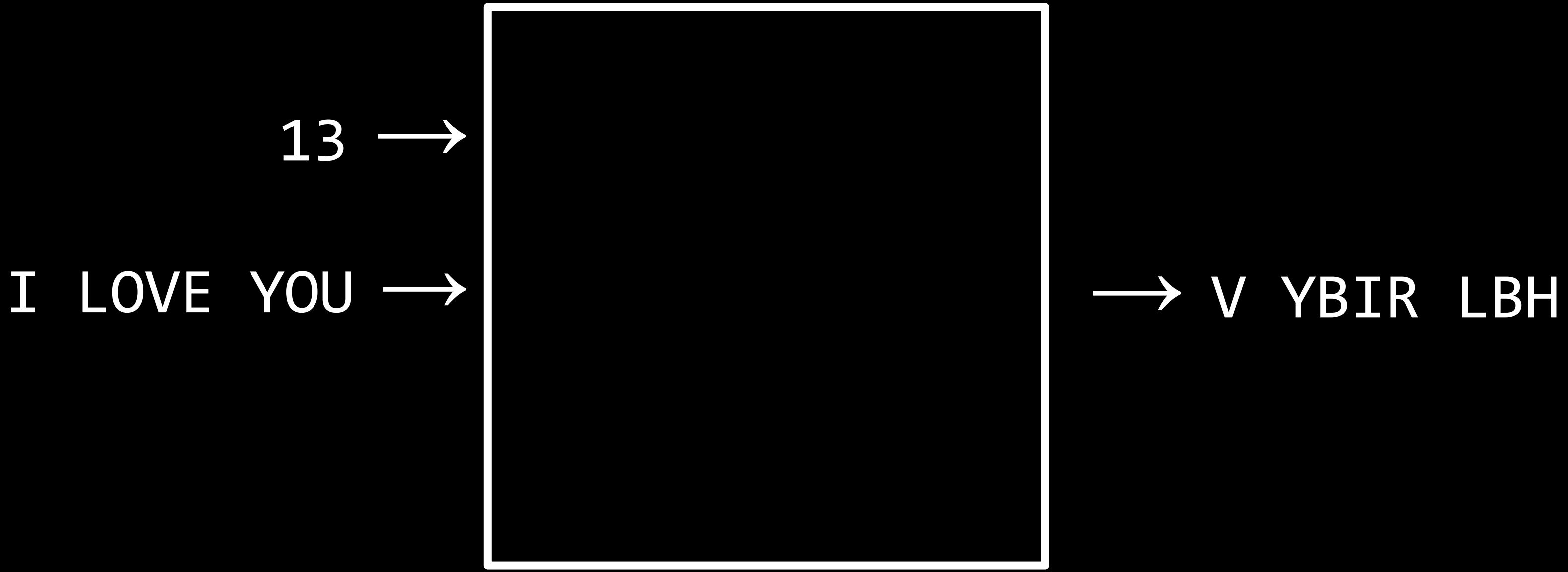
IJ!



13 →

HI! →

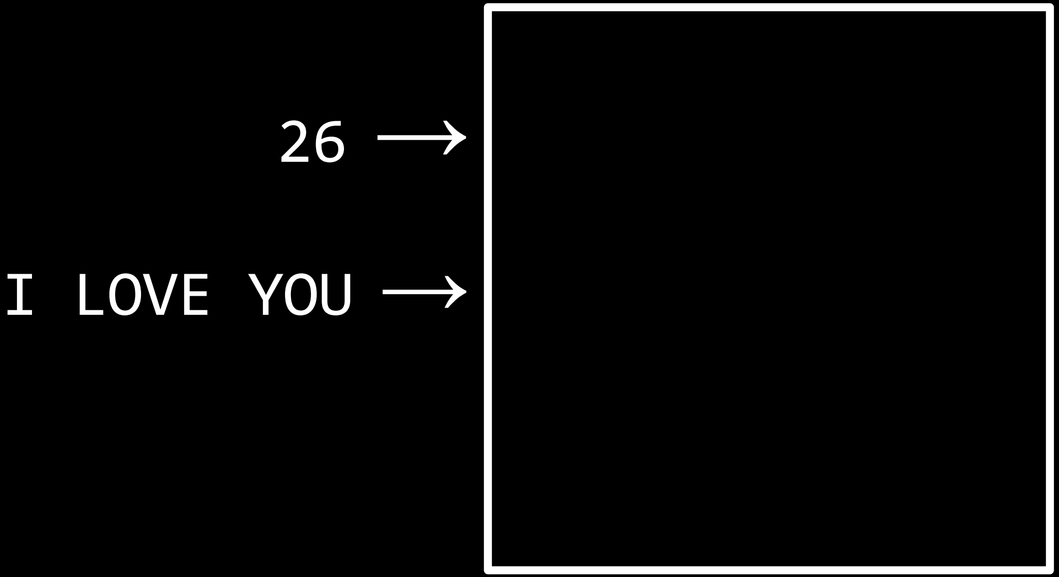
→ UV!



13 →

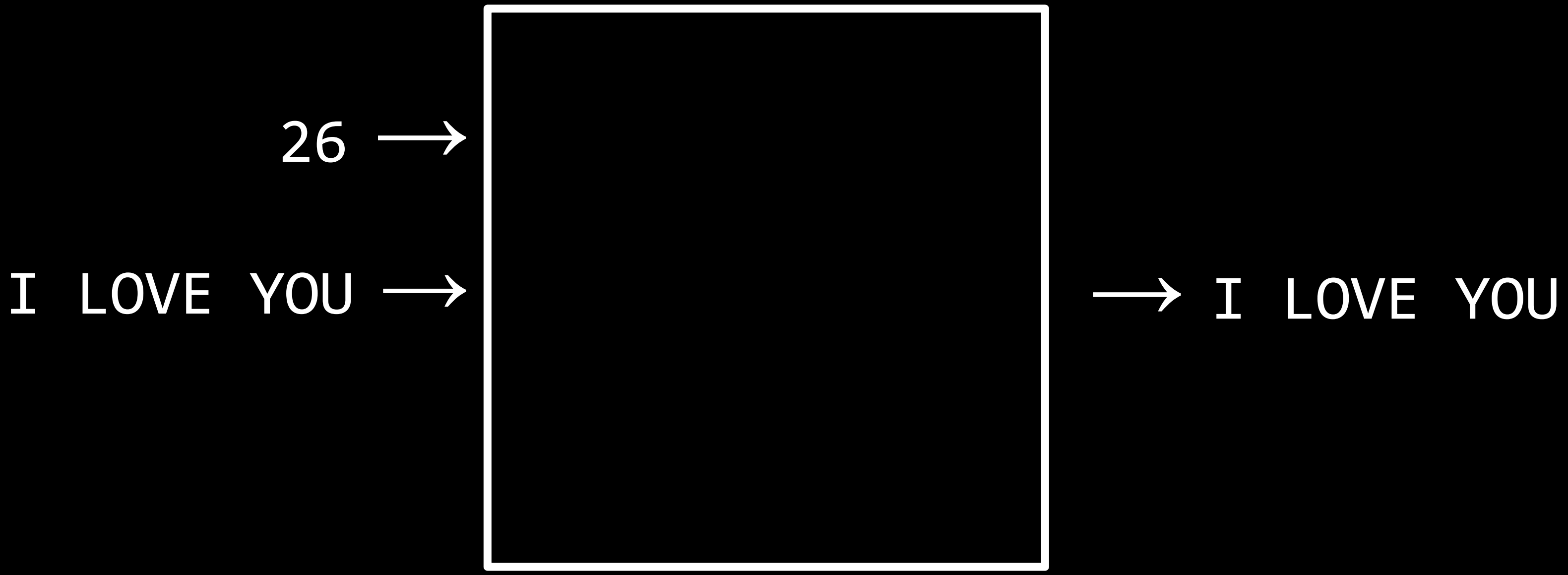
I LOVE YOU →

→ V YBIR LBH



26 →

I LOVE YOU →

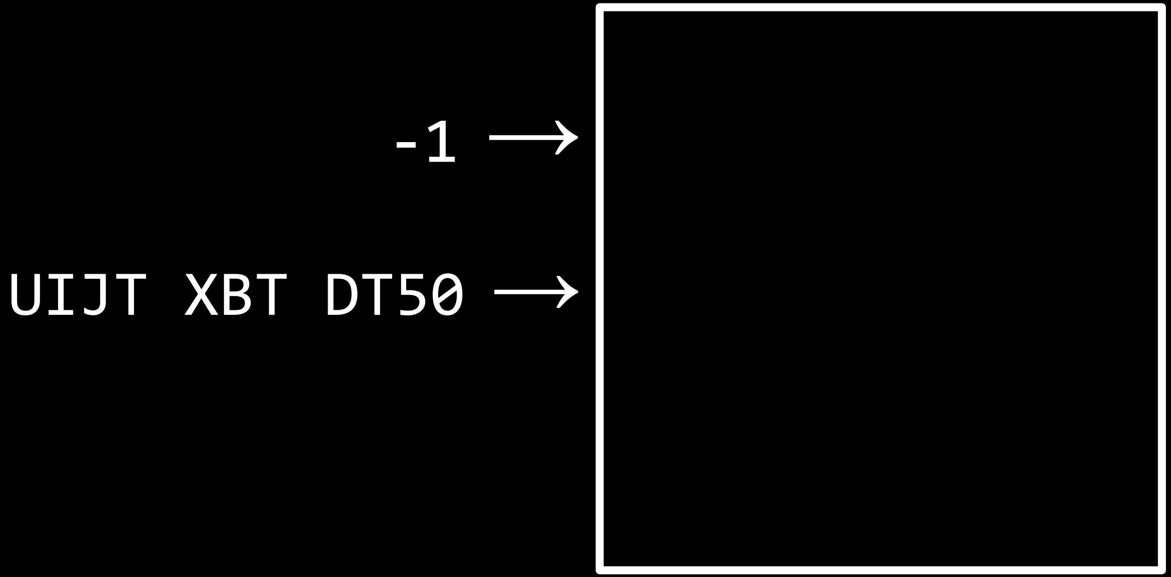


26

I LOVE YOU

I LOVE YOU

decryption



-1 →

UIJT XBT DT50 →

UIJT XBT DT50

TIJT XBT DT50

THJT XBT DT50

THIT XBT DT50

THIS XBT DT50

THIS WBT DT50

THIS WAT DT50

THIS WAS DT50

THIS WAS CS50

THIS WAS CS50

CS50
ddb

This is CS50