Herzlich Willkommen bei Inf-Einf-B Woche 2.

Heute weiter mit C.

Wenn es zu schnell geht, Section Video und Lecture Notes anschauen.

Kurs-Website mit Wochenplan, Videos, Notizen, Aufgaben, FAQ, uvm.: **inf.zone**

Die Vorlesungen werden aufgezeichnet und im Internet veröffentlicht. Die Videos werden allerdings meist erst 3 Wochen später online sein; Folien und Notizen am Tag nach der Vorlesung. Machen Sie sich Notizen!

Ihre Fragen sind in der Aufzeichnung in der Regel nicht zu hören (ich wiederhole sie).

Seien Sie furchtlos und stellen Sie Fragen!
“Publicly not knowing is valuable!”

Nutzen Sie die Zeit, bis es los geht! Stellen Sie sich mindestens zwei anderen Personen in Ihrer Nähe vor, die Sie noch nicht kennen!
(Um das Eis zu brechen, sagen Sie einfach, dass Sie nur diese Anweisung befolgen.)

Wir machen heute **zwei Pausen**.

Dies ist Inf-Einf-B.

lernen wie Werte gespeichert werden

Lesestufen

Ein Fisch. Zwei Fische. Roter Fisch. Blauer Fisch.

Erste Klasse

Im Schatten einer riesigen Lilie wohnt im Herzwald ‘ne Einhornfamilie. Der Herzwald liegt im Land der Träume. Dort sind die Blumen so groß wie Bäume! Alle Steine sind aus Plüsch und rosarot! Zum Essen gibts immer nur Kekse statt Brot.

Dritte Klasse

Es war ein strahlend-kalter Apriltag, und die Uhren schlugen dreizehn. Winston Smith, das Kinn an die Brust gezogen, um dem scheußlichen Wind zu entgehen, schlüpfte rasch durch die Glastüren der Victory Mietskaserne, doch nicht rasch genug, um zu verhindern, daß mit ihm auch ein grießiger Staubwirbel hereinwehte.

Zehnte Klasse

Übung zu Lesestufen

Übung zu Kryptographie

GLHV ZDU LQIHLQIE

Hochgeladene Übungsaufgaben

Woche 0 Scratch	Woche 1 C
88	115+

Selbstkontrollfragen aus letzter Vorlesung
n = 124

KONNTE ICH vs KONNTE ICH NOCH NICHT

Scratch Tracing	Pseudocode Tracing	Parsons Problem
70%	56%	68%
	⚠️	

Selbstkontrollfragen aus letzter Vorlesung
n = 124

alle drei gelöst	zwei von drei	eine von drei	keine
33%	35%	26%	7%

PROBLEM 1

```
x = 1
    sum = 0

    repeat 4 times:
        if x > 5:
            sum = sum + x
        x = x + 3

    print sum
```

Was wird ausgegeben?

PROBLEM 1

Iteration	x	x>5?	sum
Start	1	-	0
1	1	No	0
After 1	4	-	0
2	4	No	0
After 2	7	-	0
3	7	Yes	7
After 3	10	-	7
4	10	Yes	17
After 4	13	-	17

Output: 17

Was wird ausgegeben?

```
A int height = get_int("Height: ");
B }
C {
D #include <stdio.h>
E printf("\n");
F int main(void)
G }
H for (int i = 0; i < length; i++) {
I #include <cs50.h>
J void print_line(int length)
K }
L print_line(row);
M for (int row = 1; row <= height; row++) {
N {
O }
P printf("#");
```

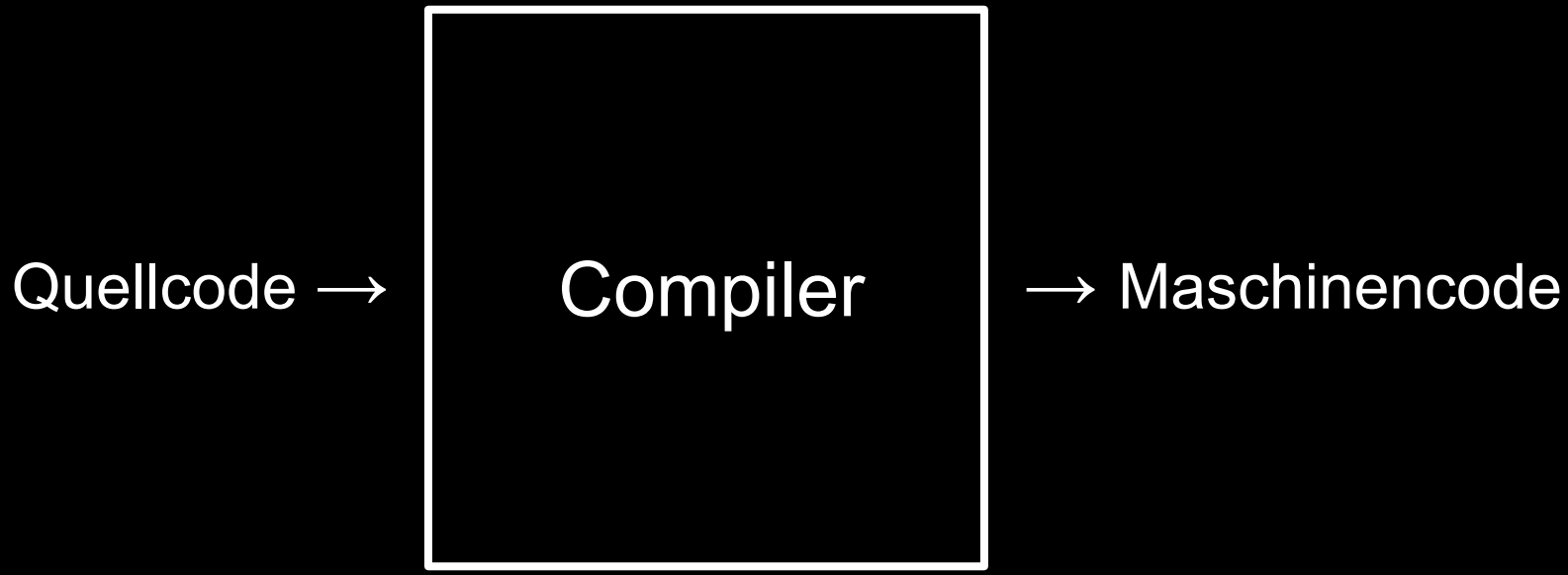
PROBLEM 2

Gib eine Mario-Pyramide aus!

PROBLEM 2

Gib eine Mario-Pyramide aus!

```c
#include <cs50.h>
#include <stdio.h>
void print_line(int length)
{
    for (int i = 0; i < length; i++) {
        printf("#");
    }
    printf("\n");
}
int main(void)
{
    int height = get_int("Height: ");
    for (int row = 1; row <= height; row++) {
        print_line(row);
    }
}
```


Quellcode →
Compiler
→ Maschinencode

```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 <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");
}
```

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

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

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

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

```c
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);
}
```

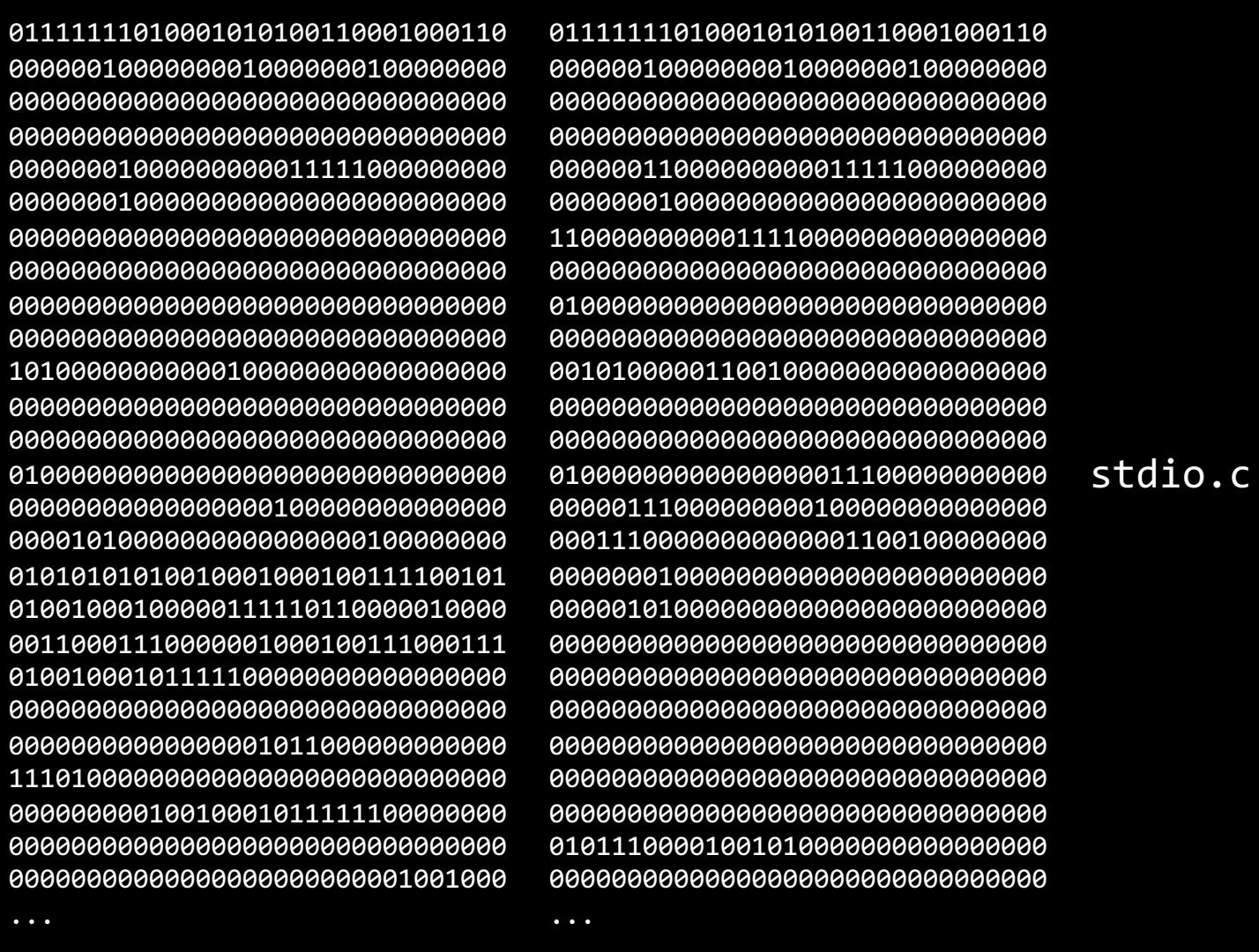
hello.c

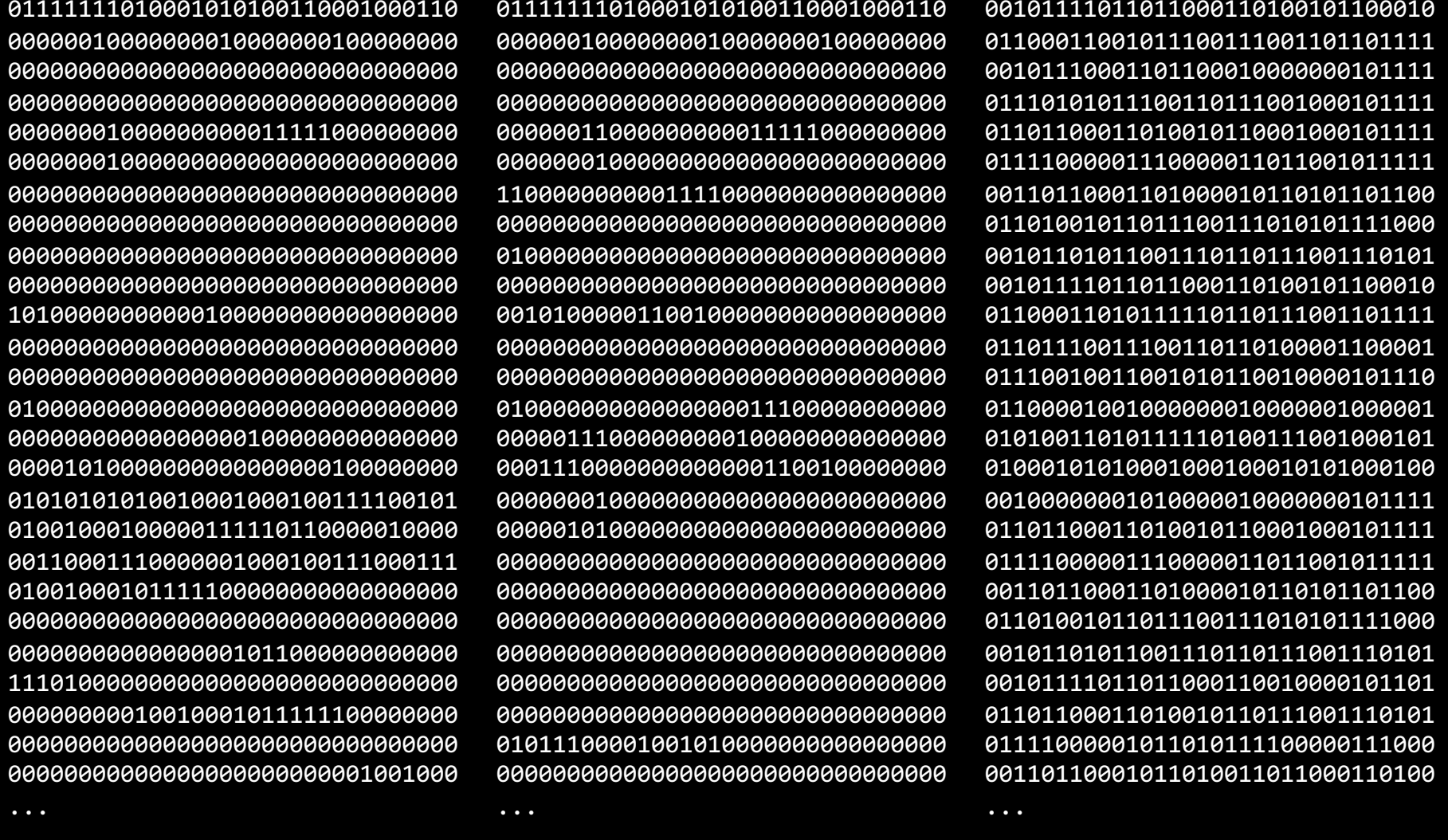
hello.c

cs50.c

hello.c cs50.c stdio.c


cs50.c
stdio.c


stdio.c




01111111010001010100110001000110000001000000010000000100000000000000000000000000000000000000000000
00000000000000000000000000000000010000000001111100000000000000100000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000001010000000000100000000000000000000000000000000000000000000000000000000000000000000
00000000000001000000000000000000000000000000000000000000000000010000000000000000010100000000000010
00000000101010101001000100010011110010101001000100000111110110000010000001100011100000010001001110001
11010010001011111000000000000000000000000000000000000000000000000000000000000000000000000000000010111000000000000111
01000000000000000000000000000000000001001000101111110000000000000000000000000000000000000000000000000000
00000000000000001001000...01111111010001010100110001000110000001000000010000000100000001000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000110000000000111110000000000000000010000000
00000000000000001100000000001111000000000000000000000000000000000000000000000000000000000000000100000000000000000
00000000000000000000000000000000000000000000000010100000110010000000000000000000000000000000000000000000
00000000000000000000000000000000000000010000000000000000111000000000000000111000000000100000000000000
00000111000000000000110010000000000000010000000000000000000000000000000010100000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000010111000010010
10100000000000000000000000000000000000000000000...00101111011011000110100101100010011000110010111
00111001101101111001011100011011000100000010111101110101011100110111001000101111011011000110100101100
00100010111101111000001110000011011001011111001101100011010000101110101101100011010010110111001110101010
11110000010110101100111011011100111010100101111011011000110100101100010011000110101111101101110011011
11011011100111001101101000011000010111001001100101011001000010111001100001001000000100000010000010100
10011010111110100111001000101010001010100010001000101010001000010000000101000001000000010111101101100
01101001011000100010111101111000001110000011011001011111001101100011010000101101011011000110100101101
11001110101011110000010110101100111011011100111010100101111011011000110010000101101011011000110100101
10111001110101011110000010110101111000001110000011011000101101001101100011010...

preprocessing

compiling

assembling

linking

compiling

decompiling

reverse engineering


01111111010001010100110001000110000001000000010000000100000000000000000000000000000000000000000000
00000000000000000000000000000000010000000001111100000000000000100000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000001010000000000100000000000000000000000000000000000000000000000000000000000000000000
00000000000001000000000000000000000000000000000000000000000000010000000000000000010100000000000010
00000000101010101001000100010011110010101001000100000111110110000010000001100011100000010001001110001
11010010001011111000000000000000000000000000000000000000000000000000000000000000000000000000000010111000000000000111
01000000000000000000000000000000000001001000101111110000000000000000000000000000000000000000000000000000
00000000000000001001000...01111111010001010100110001000110000001000000010000000100000001000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000110000000000111110000000000000000010000000
00000000000000001100000000001111000000000000000000000000000000000000000000000000000000000000000100000000000000000
00000000000000000000000000000000000000000000000010100000110010000000000000000000000000000000000000000000
00000000000000000000000000000000000000010000000000000000111000000000000000111000000000100000000000000
00000111000000000000110010000000000000010000000000000000000000000000000010100000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000010111000010010
10100000000000000000000000000000000000000000000...00101111011011000110100101100010011000110010111
00111001101101111001011100011011000100000010111101110101011100110111001000101111011011000110100101100
00100010111101111000001110000011011001011111001101100011010000101110101101100011010010110111001110101010
11110000010110101100111011011100111010100101111011011000110100101100010011000110101111101101110011011
11011011100111001101101000011000010111001001100101011001000010111001100001001000000100000010000010100
10011010111110100111001000101010001010100010001000101010001000010000000101000001000000010111101101100
01101001011000100010111101111000001110000011011001011111001101100011010000101101011011000110100101101
11001110101011110000010110101100111011011100111010100101111011011000110010000101101011011000110100101
10111001110101011110000010110101111000001110000011011000101101001101100011010...

debugging


COMO HOPPL

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

9/9

0800 Arctan started { 1.2700 9.037 847 025
1000 " stopped - arctan ✓ 9.037 846 ?95 correct
13"uc (032) MP - MC ~~1.5821[illegible]000~~ ~~2.13047641[illegible]~~) 4.615925059(-2)
(033) PRO 2 2.13047641[illegible]
correct 2.13067641[illegible]
Relays 6-2 in 033 failed special speed test
tm 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.

~~1645~~ 1700 Arctangent started.
1700 closed down.

In relay . 11.00 test.

Relays changed

1100 Started Cosine Tape (Sine check)

1525 Started Multy Adder Test.

1545 

Relay #70 Panel F
(moth) in relay.

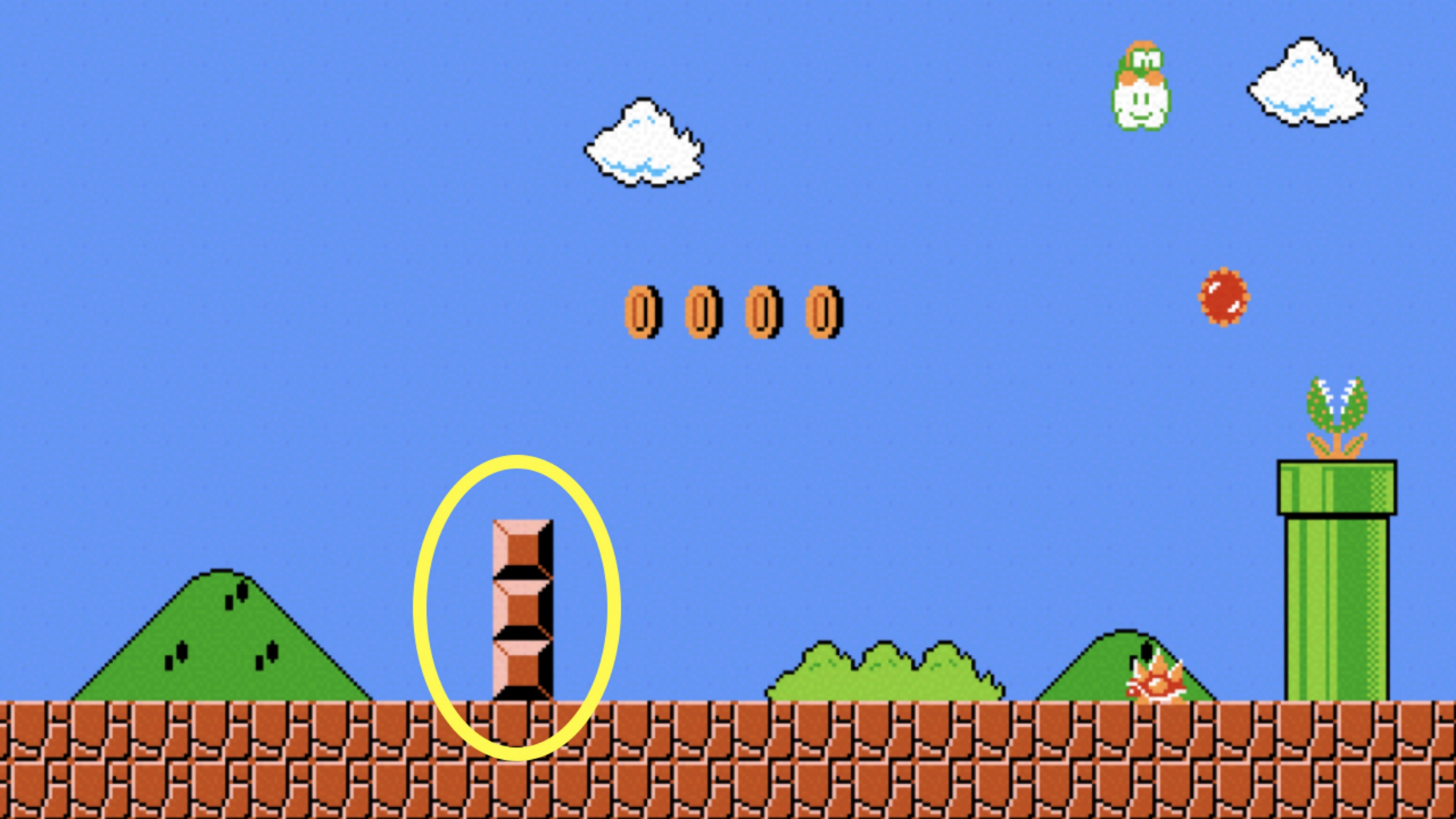
First actual case of bug being found.

1630 Antanna started.

1700 closed down.

debugging

Vier Methoden



printf

printf

debug50

`printf`

`debug50`

rubber duck

rubber duck debugging

`printf`

`debug50`

rubber duck

cs50.ai (und andere Chatbots)


CS50

PAUSE

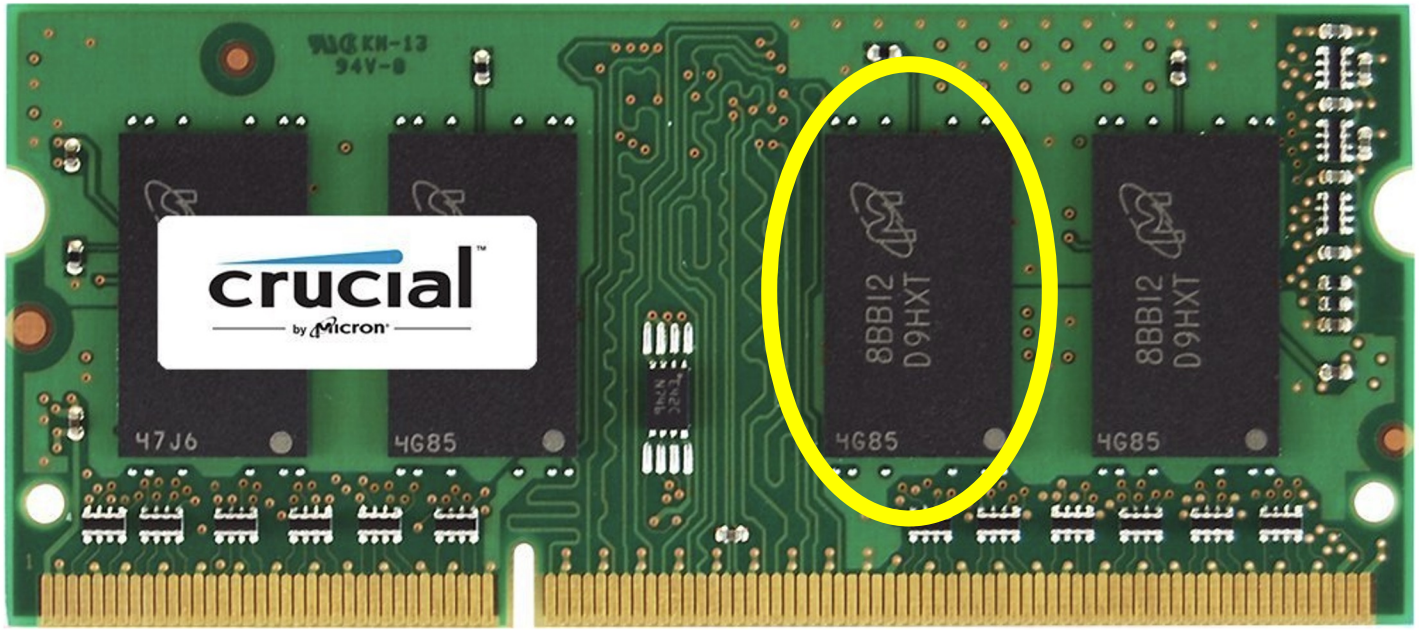
15 min

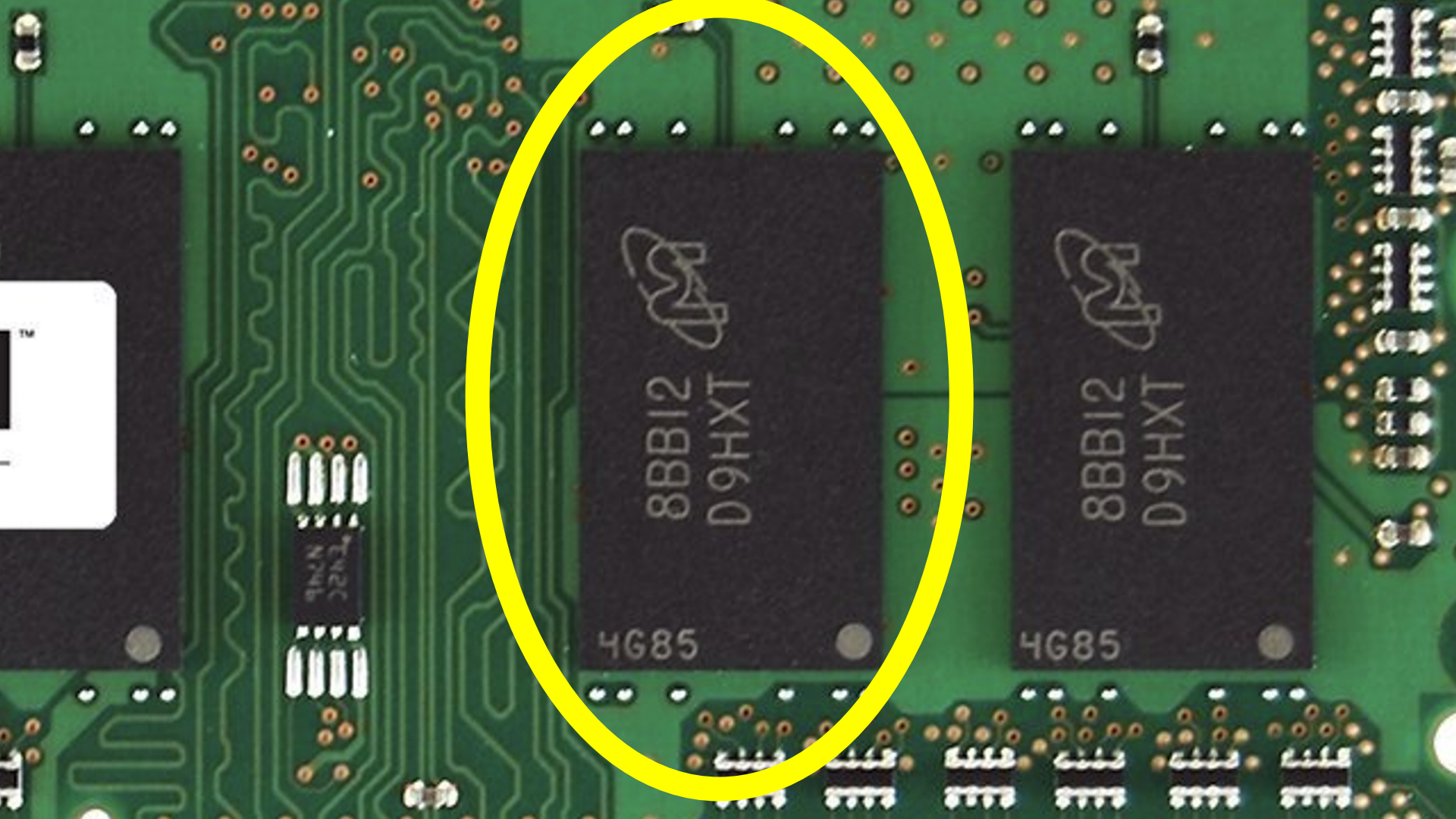
Datentypen

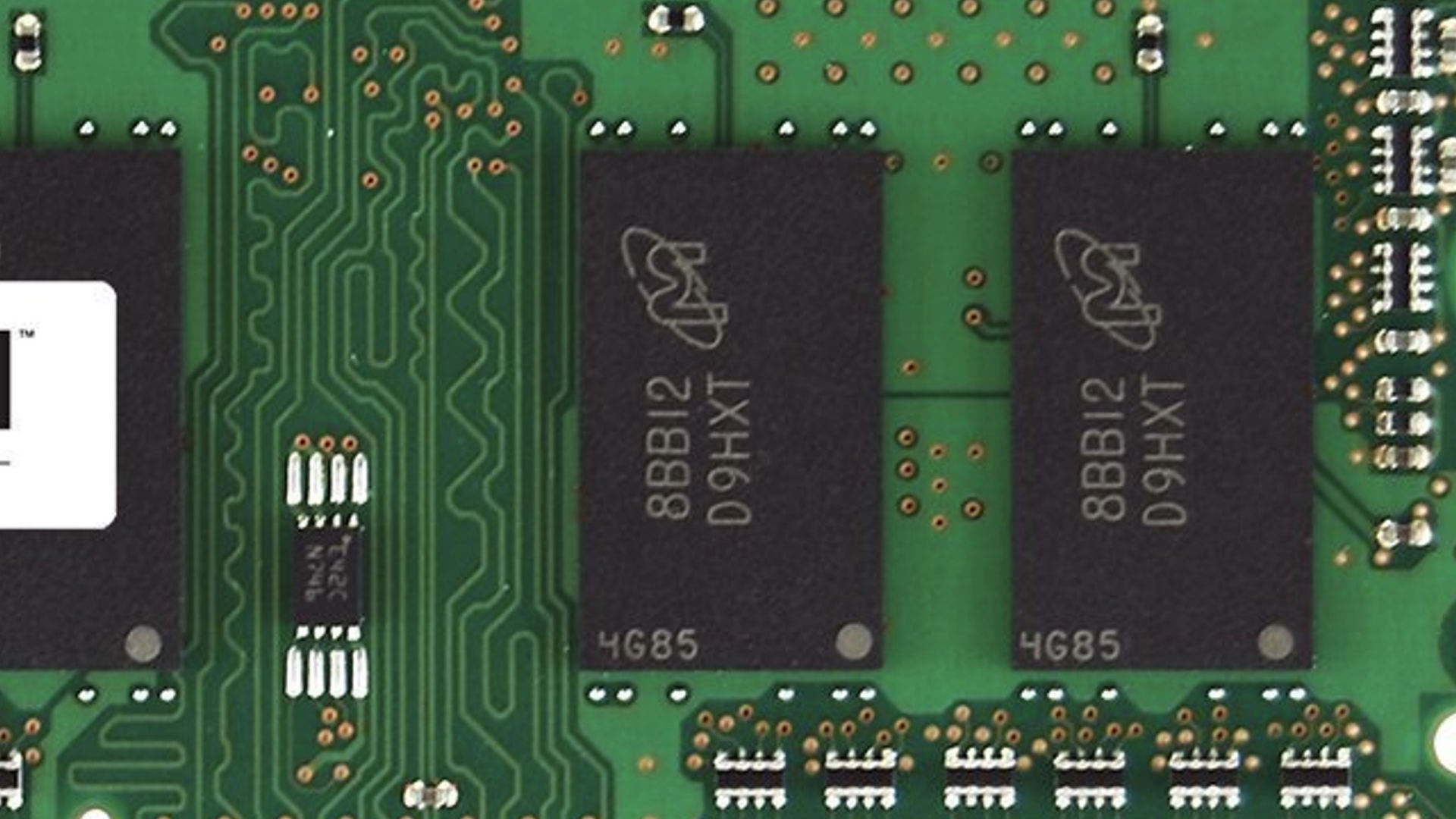
```
bool

int

long

float

double

char

string

...
```

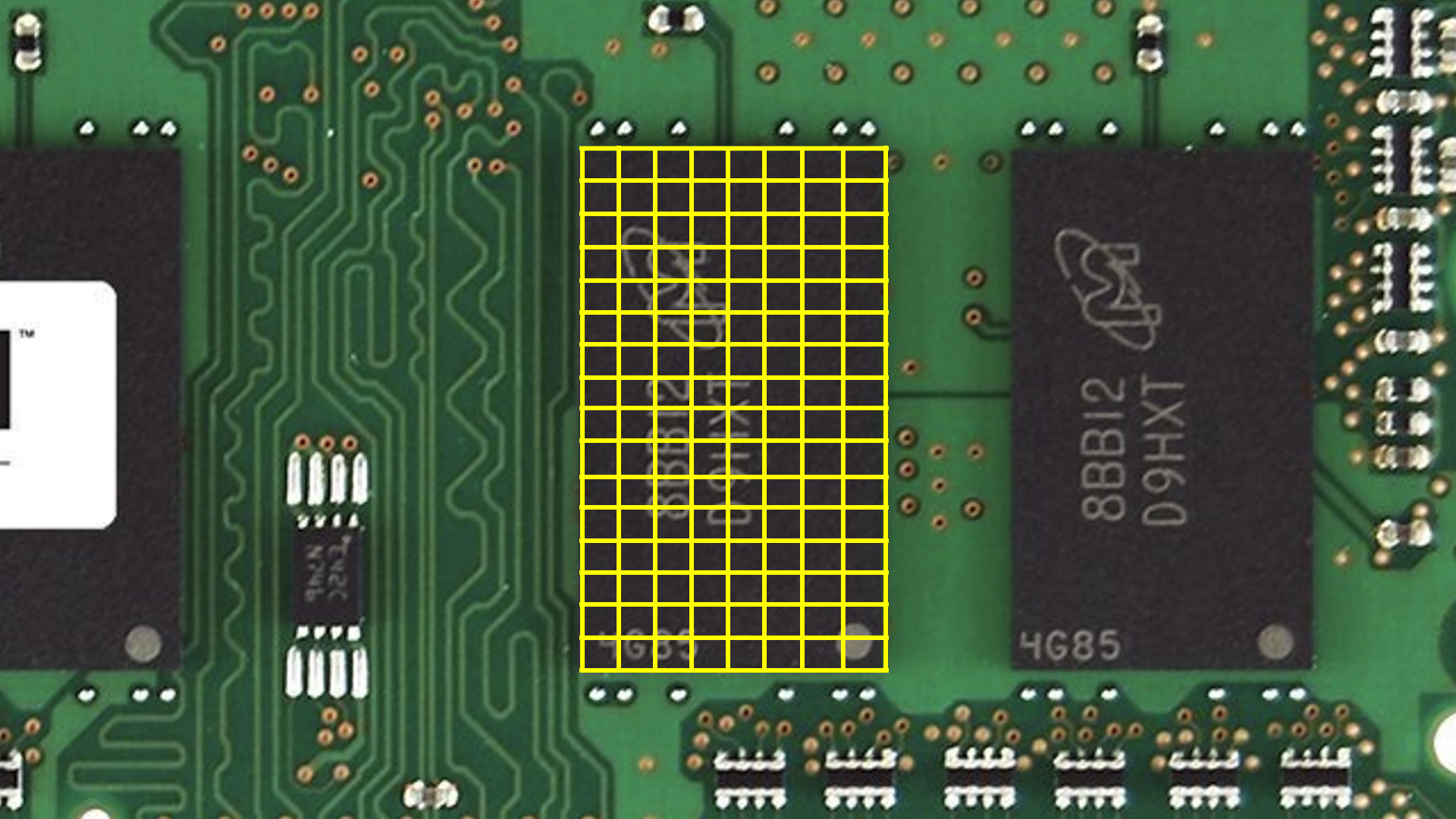
`bool`	1 byte
`int`	4 bytes
`long`	8 bytes
`float`	4 bytes
`double`	8 bytes
`char`	1 byte
`string`	? bytes
...	

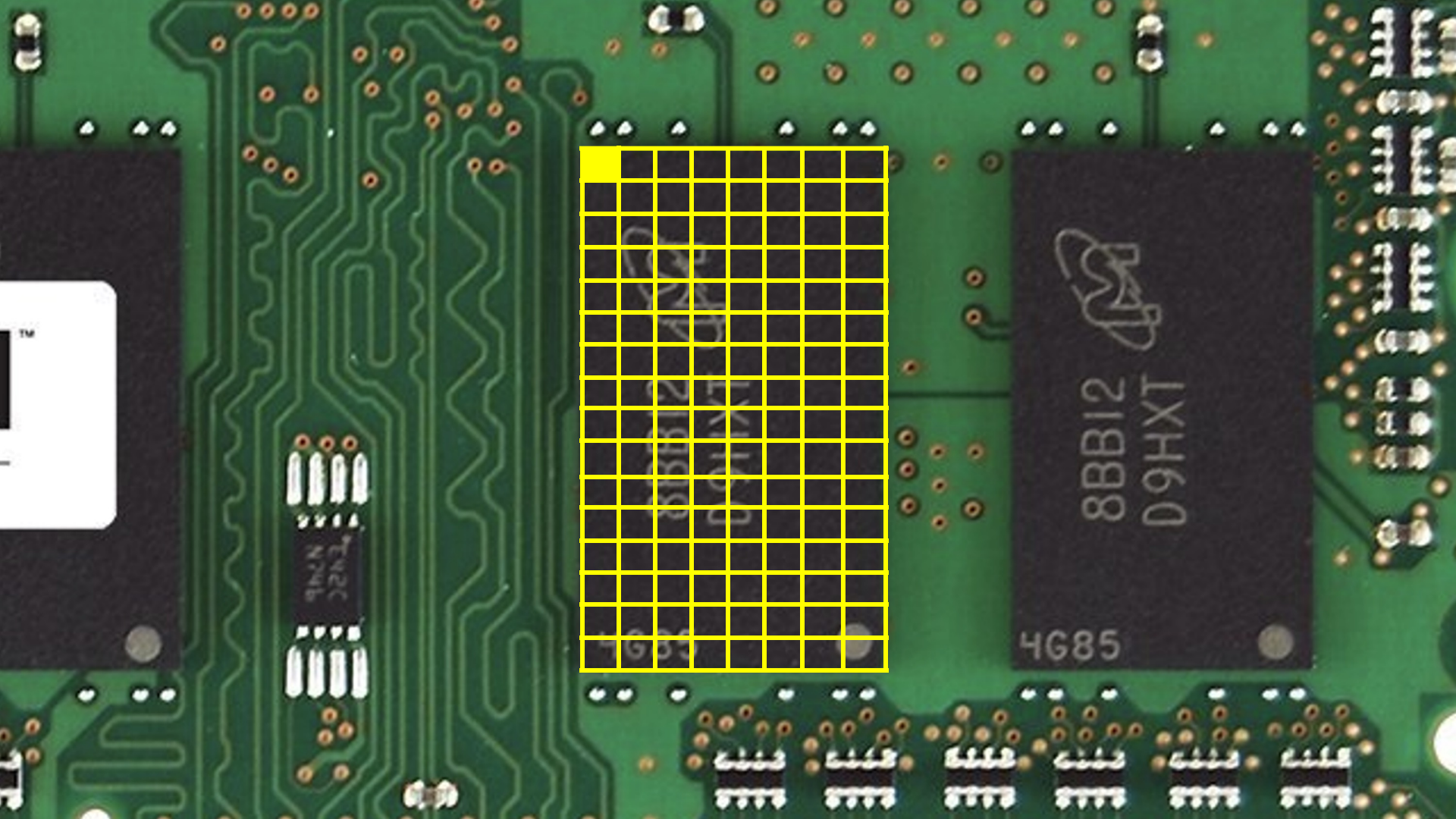

crucial
by Micron
8BBI2
D9HXT
8BBI2
D9HXT
4G85
4G85
4G85
47J6

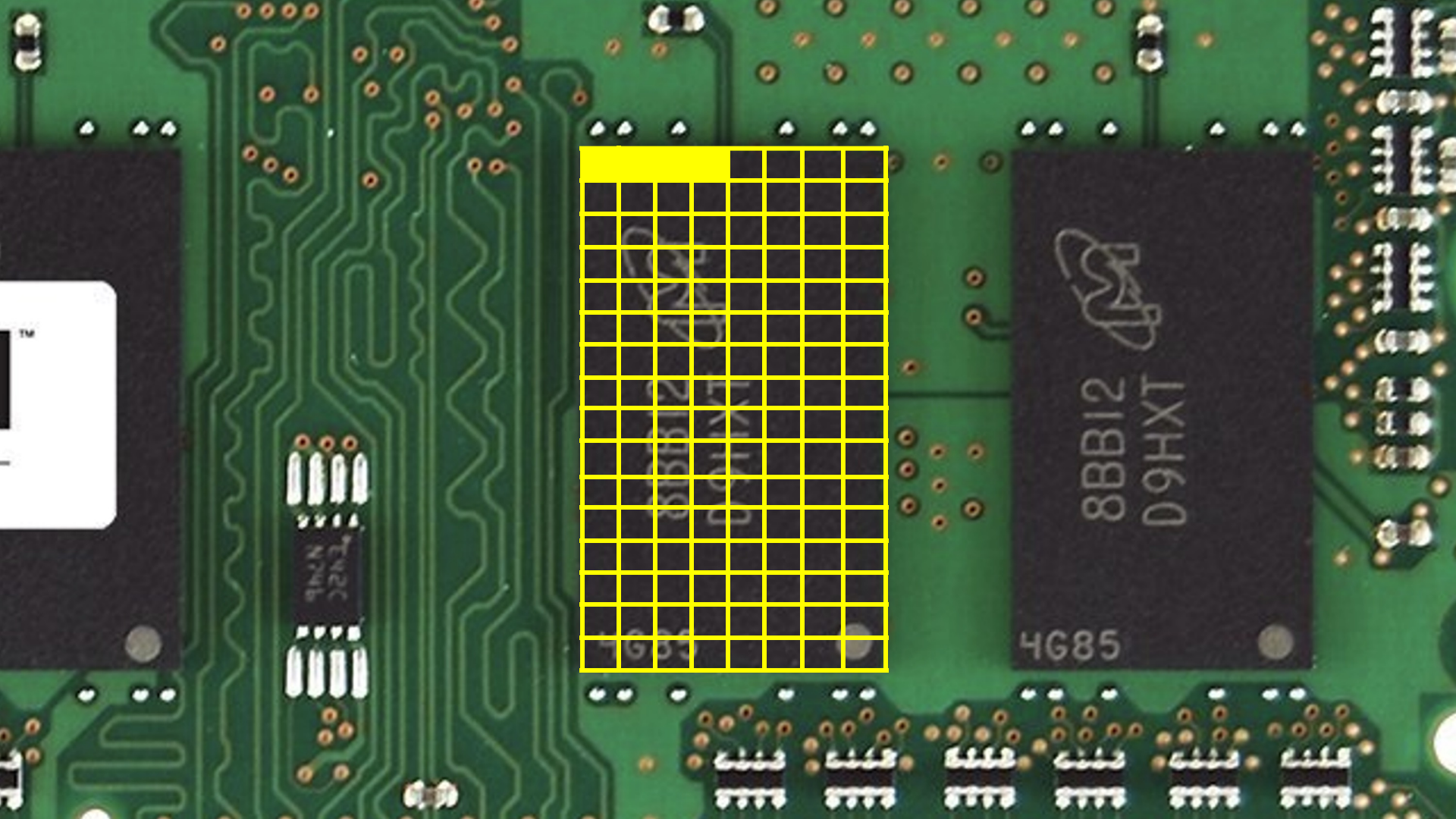

crucial
by Micron
8BBI2
D9HXT
8BBI2
D9HXT
4G85
4G85
4G85
47J6

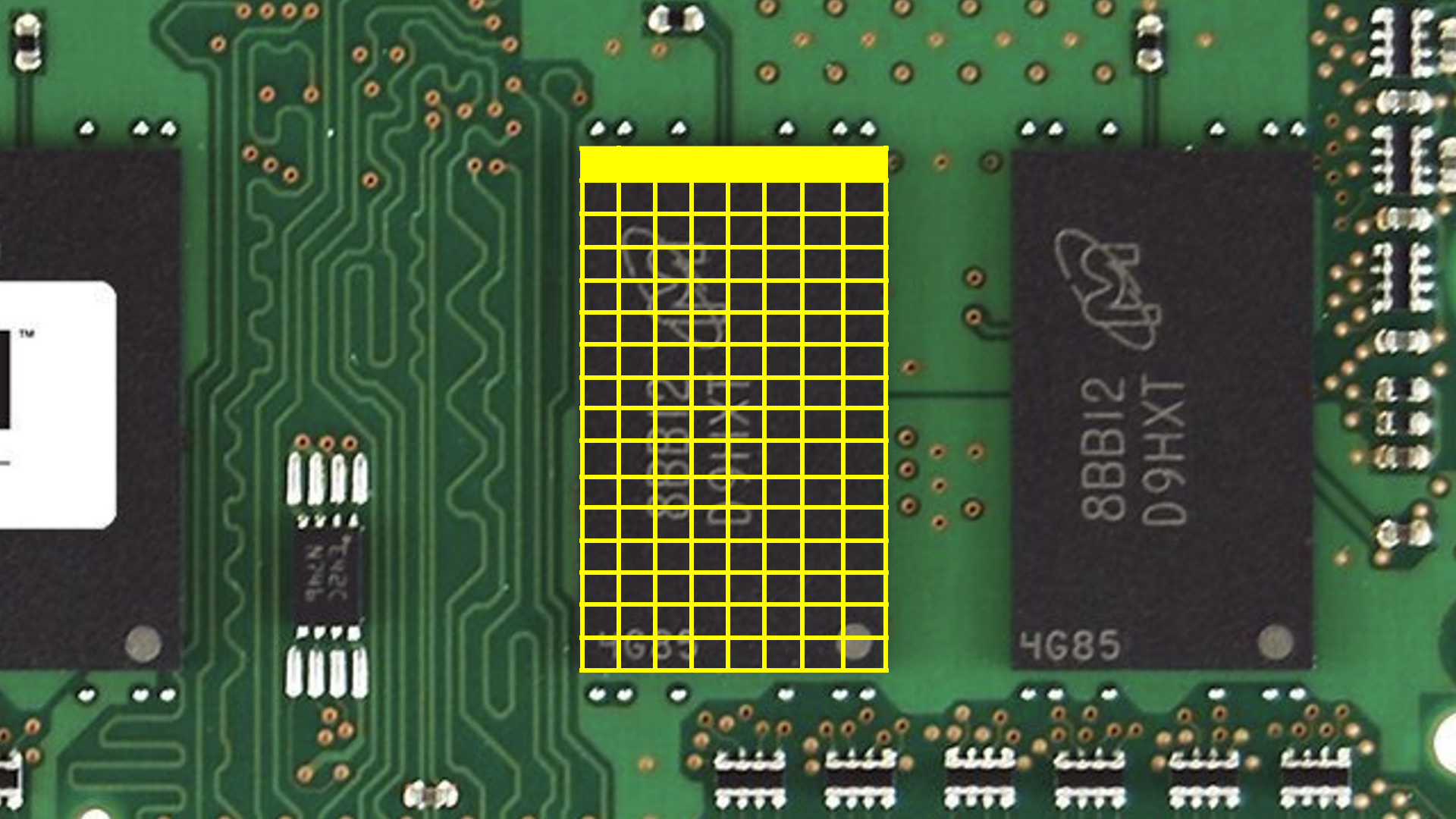

8BBI2
D9HXT
4G85
8BBI2
D9HXT
4G85

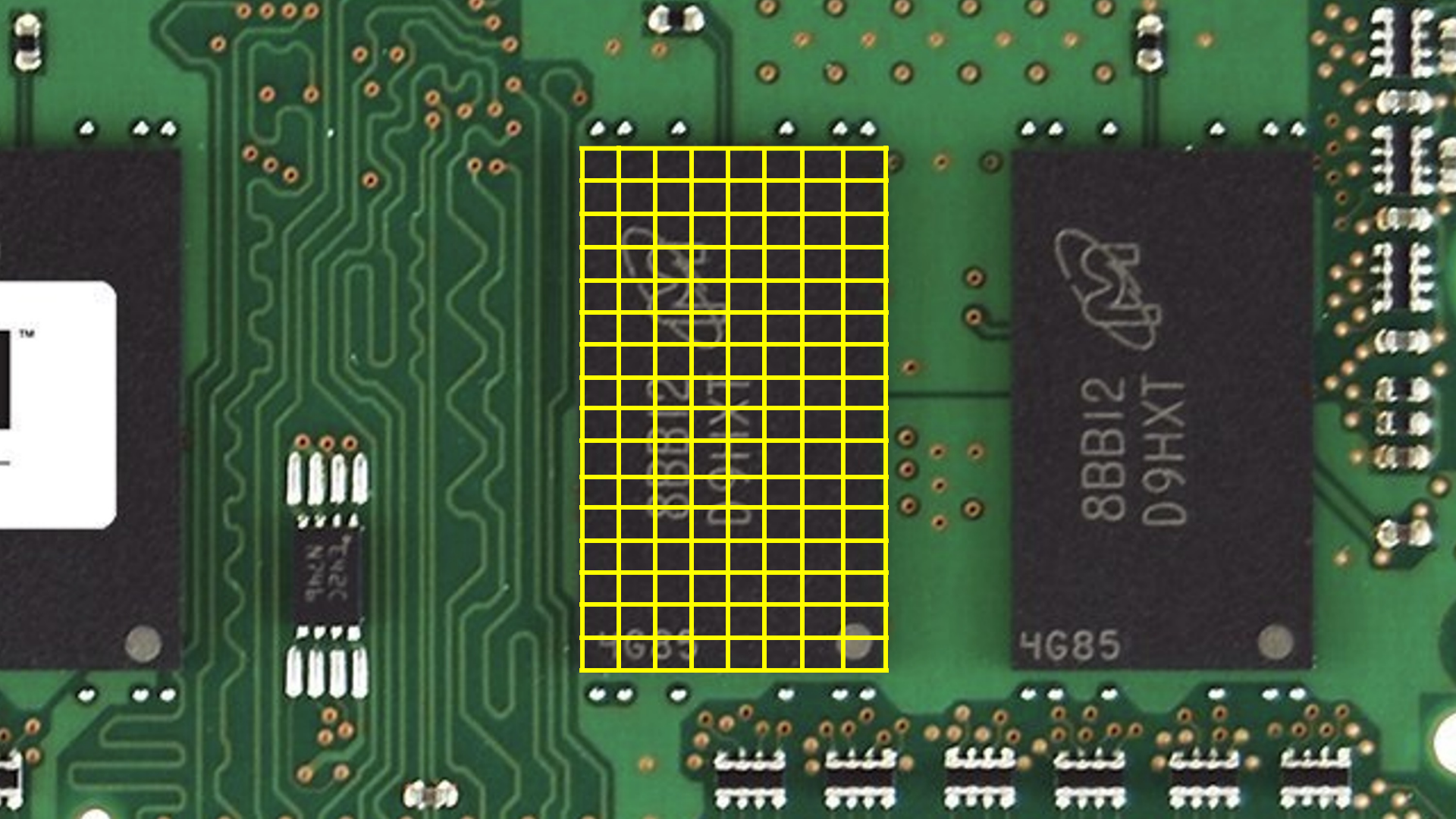

8BBI2
D9HXT
4G85
8BBI2
D9HXT
4G85

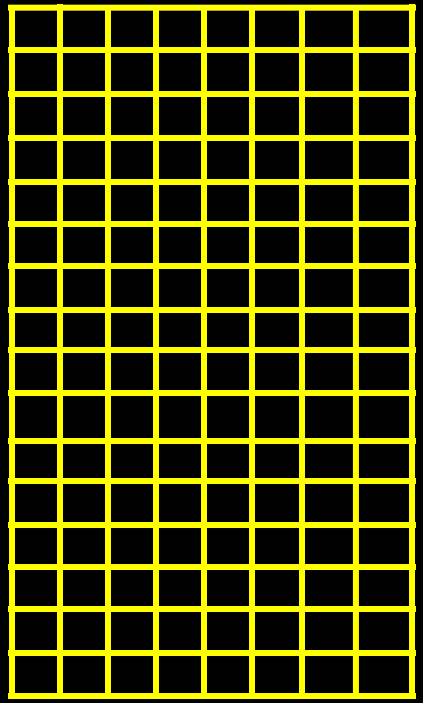

8BBI2
D9HXT
4G85


8BBI2
D9HXT
4G85


8BBI2
D9HXT
4G85


8BBI2
D9HXT
4G85


8BBI2
D9HXT
4G85



```
int score1 = 72;

int score2 = 73;

int score3 = 33;
```

72

score1

72

score1

73

score2

72

score1

73

score2

33

score3

00000000000000000000001001000

score1

00000000000000000000001001001

score2

00000000000000000000100001

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]

```
int scheine[] = {100, 50, 20, 10, 5};
int betrag = 85;
int count = 0;

for (int i = 0; i < 5; i++)
{
    if (betrag >= scheine[i])
    {
        betrag = betrag - scheine[i];
        count++;
    }
}

printf("Scheine: %d, Rest: %d\n", count, betrag);
```

PROBLEM 3

Gibt dieses Programm immer die Anzahl der mindestens benötigten Scheine korrekt aus?

Warum, warum nicht?

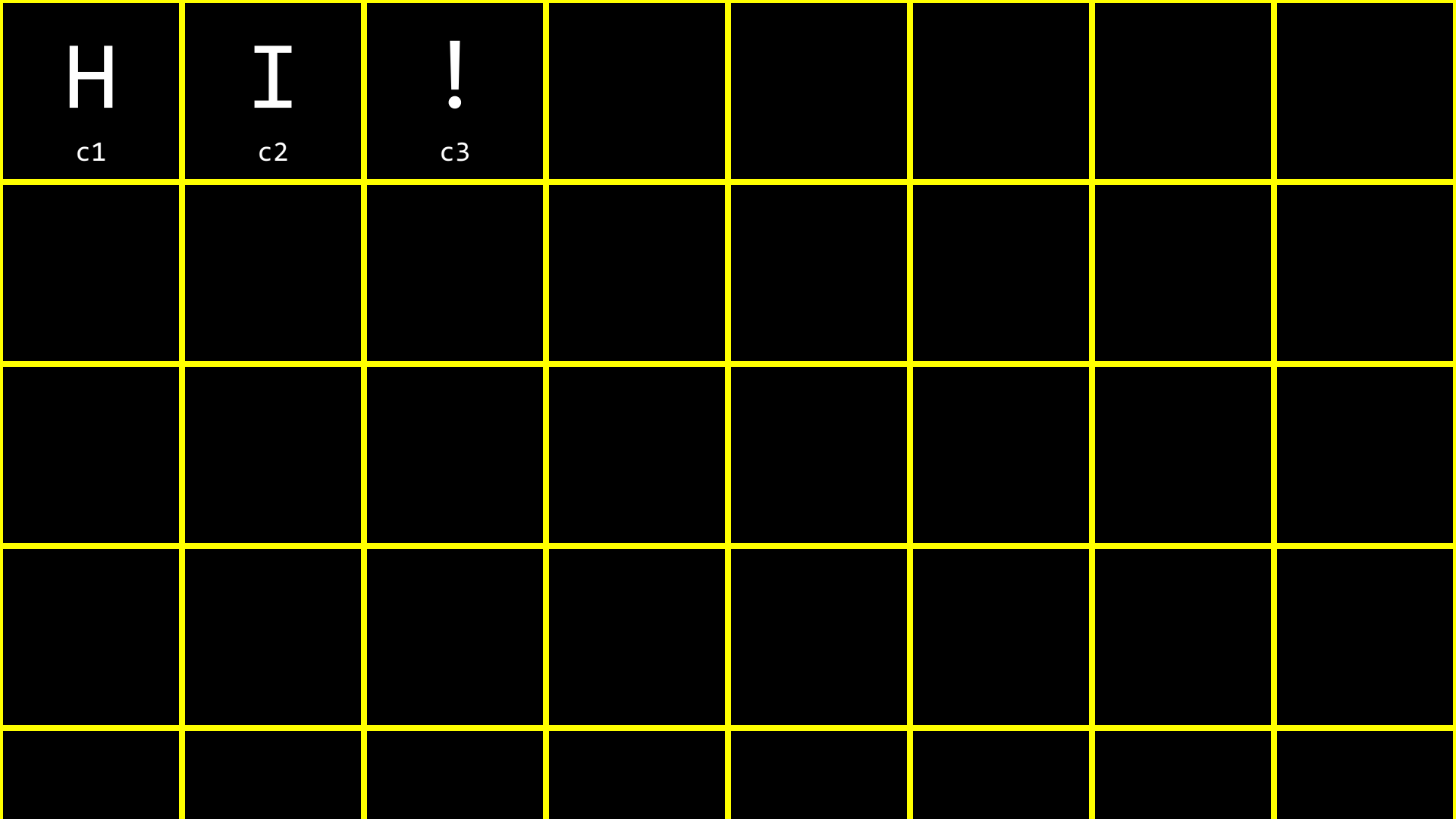
```
int scheine[] = {100, 50, 20, 10, 5};
int betrag = 85;
int count = 0;

for (int i = 0; i < 5; i++)
{
    while (betrag >= scheine[i]) // !
    {
        betrag = betrag - scheine[i];
        count++;
    }
}

printf("Scheine: %d, Rest: %d\n", count, betrag);
```

PROBLEM 3

Gibt dieses Programm immer die Anzahl der mindestens benötigten Scheine korrekt aus?

Warum, warum nicht?

```
char c1 = 'H';

char c2 = 'I';

char c3 = '!';
```


H
I
!
c1
c2
c3

72 c1	73 c2	33 c3					

01001000 c1	01001001 c2	00100001 c3					

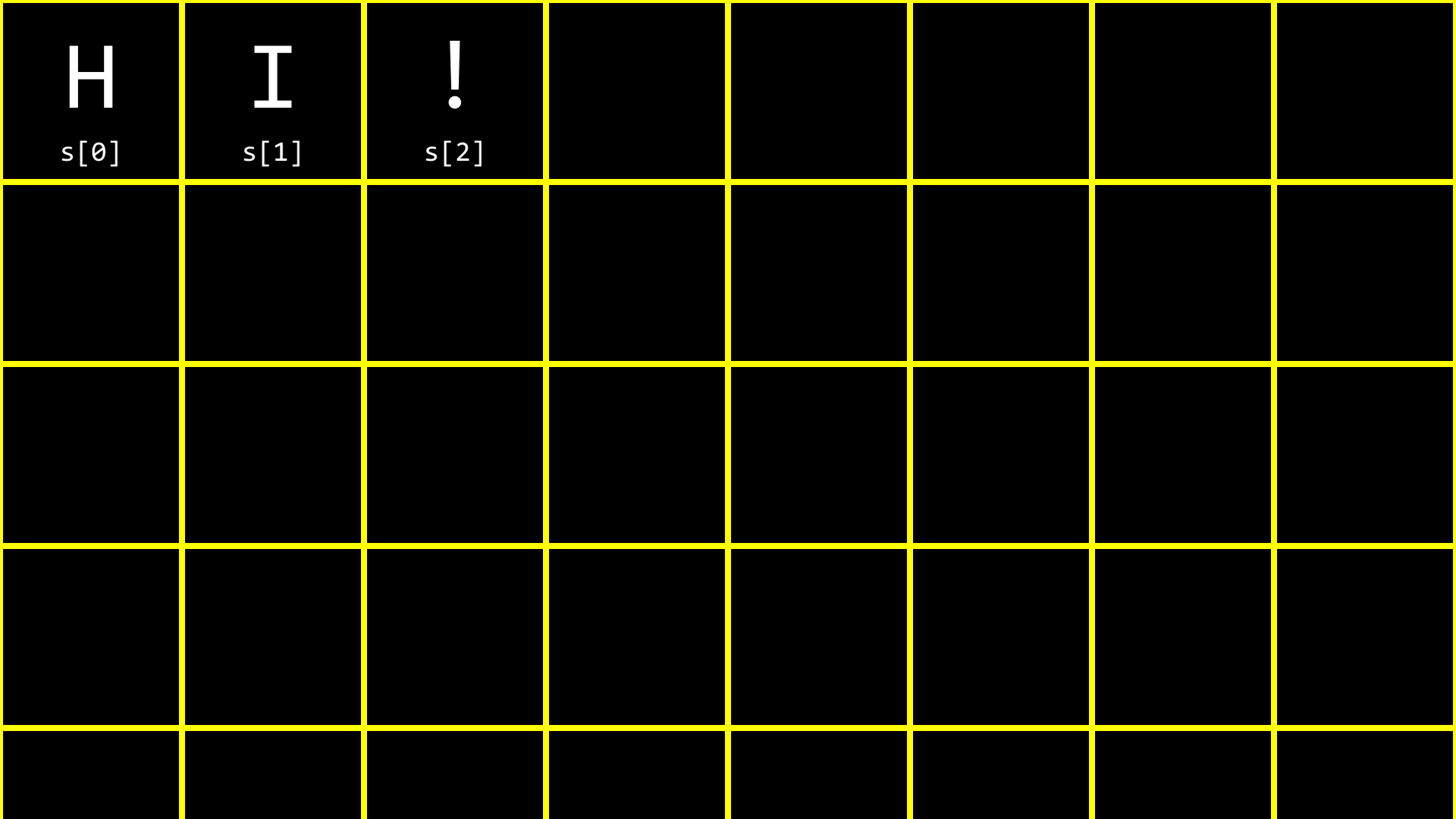
string

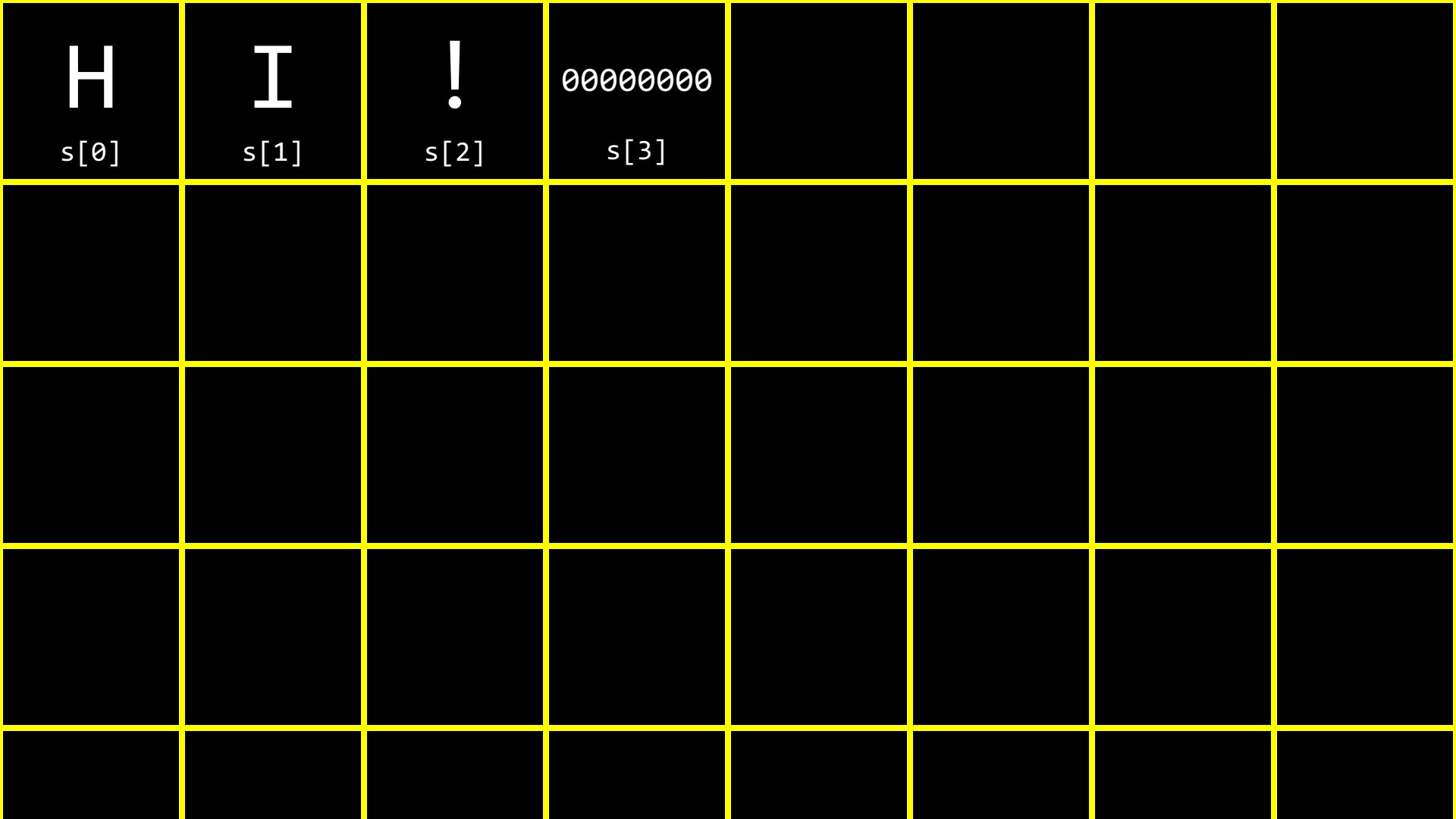
string

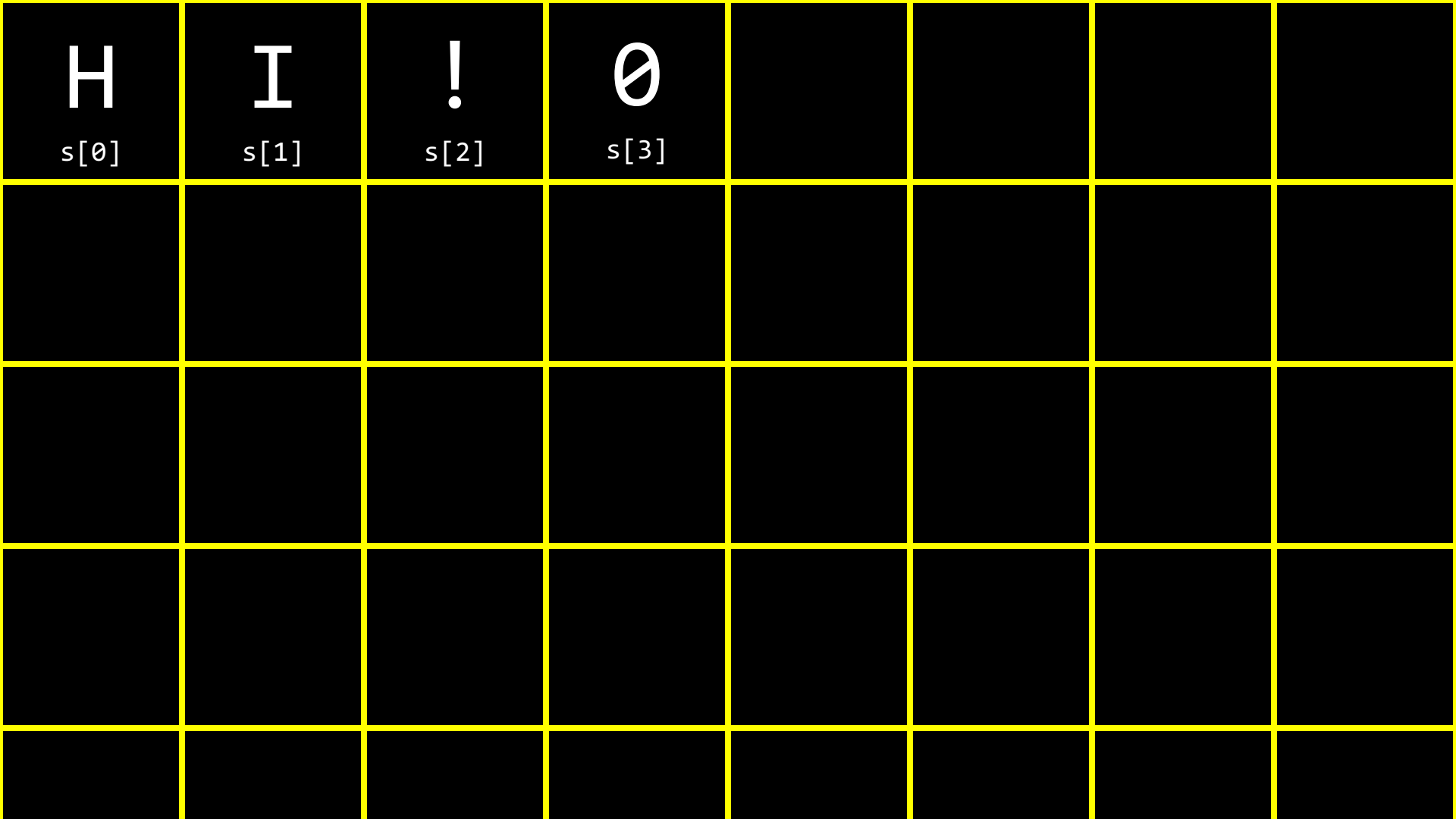
```
string s = "HI!";
```

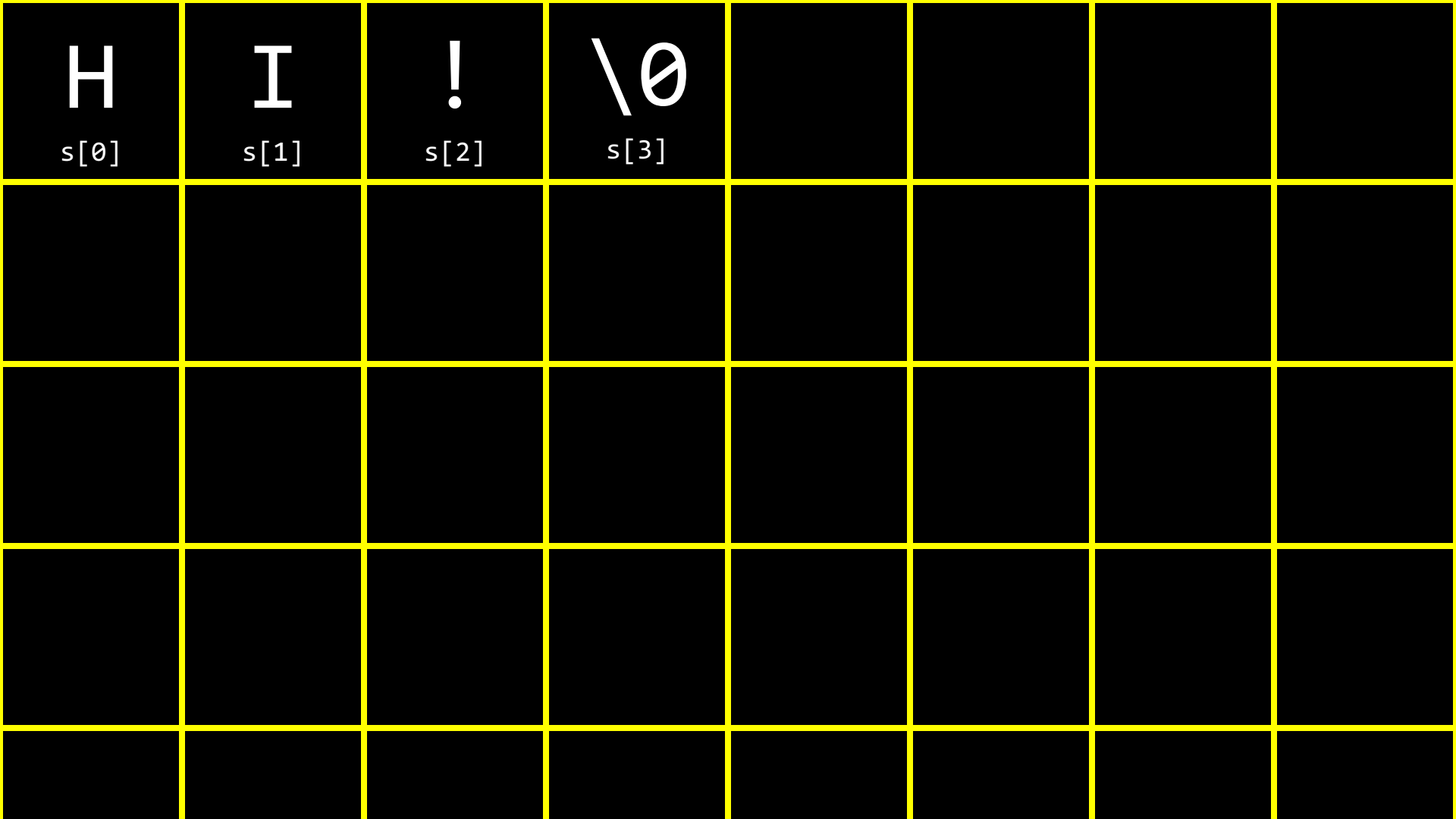
H I !

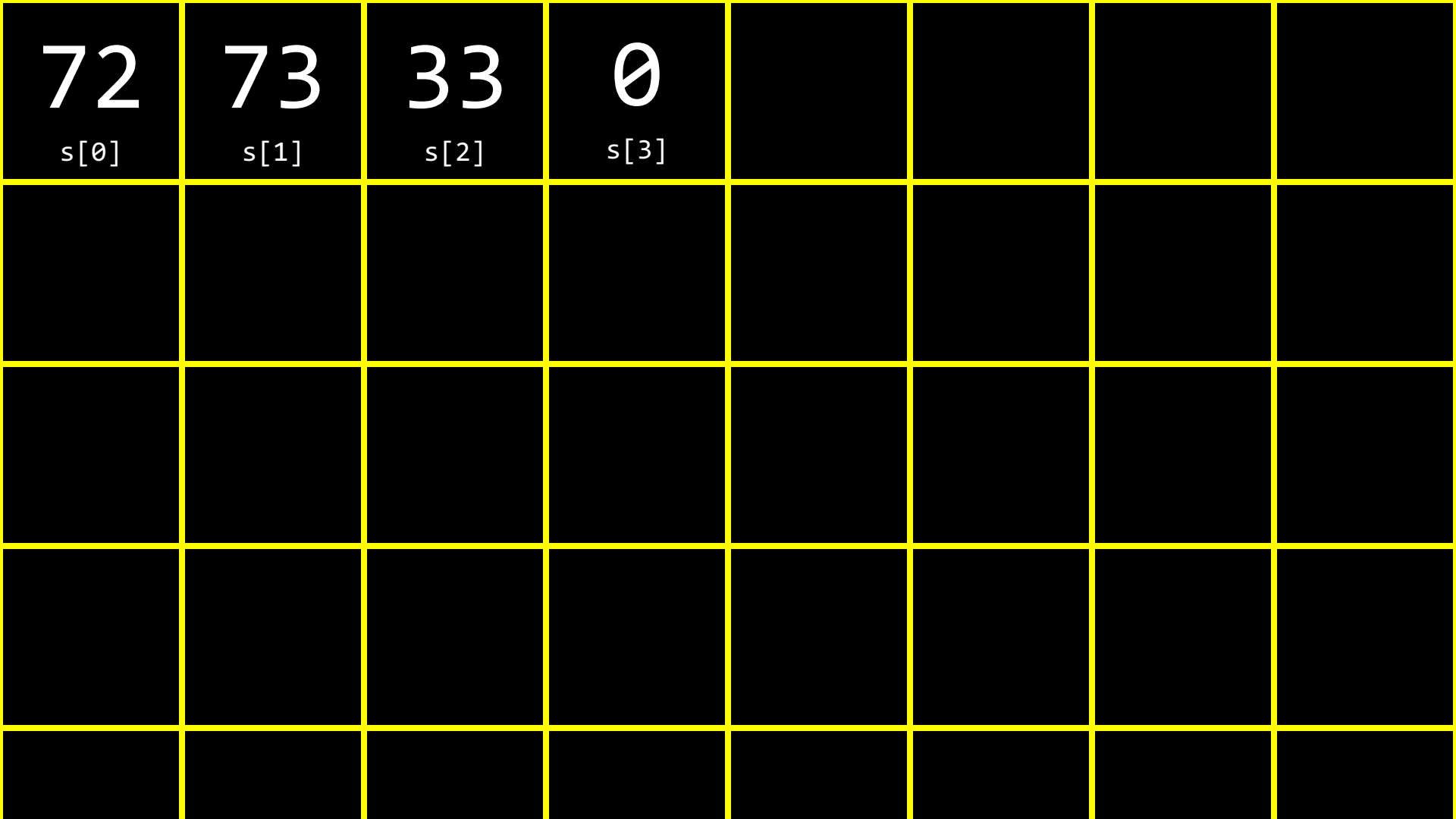
s

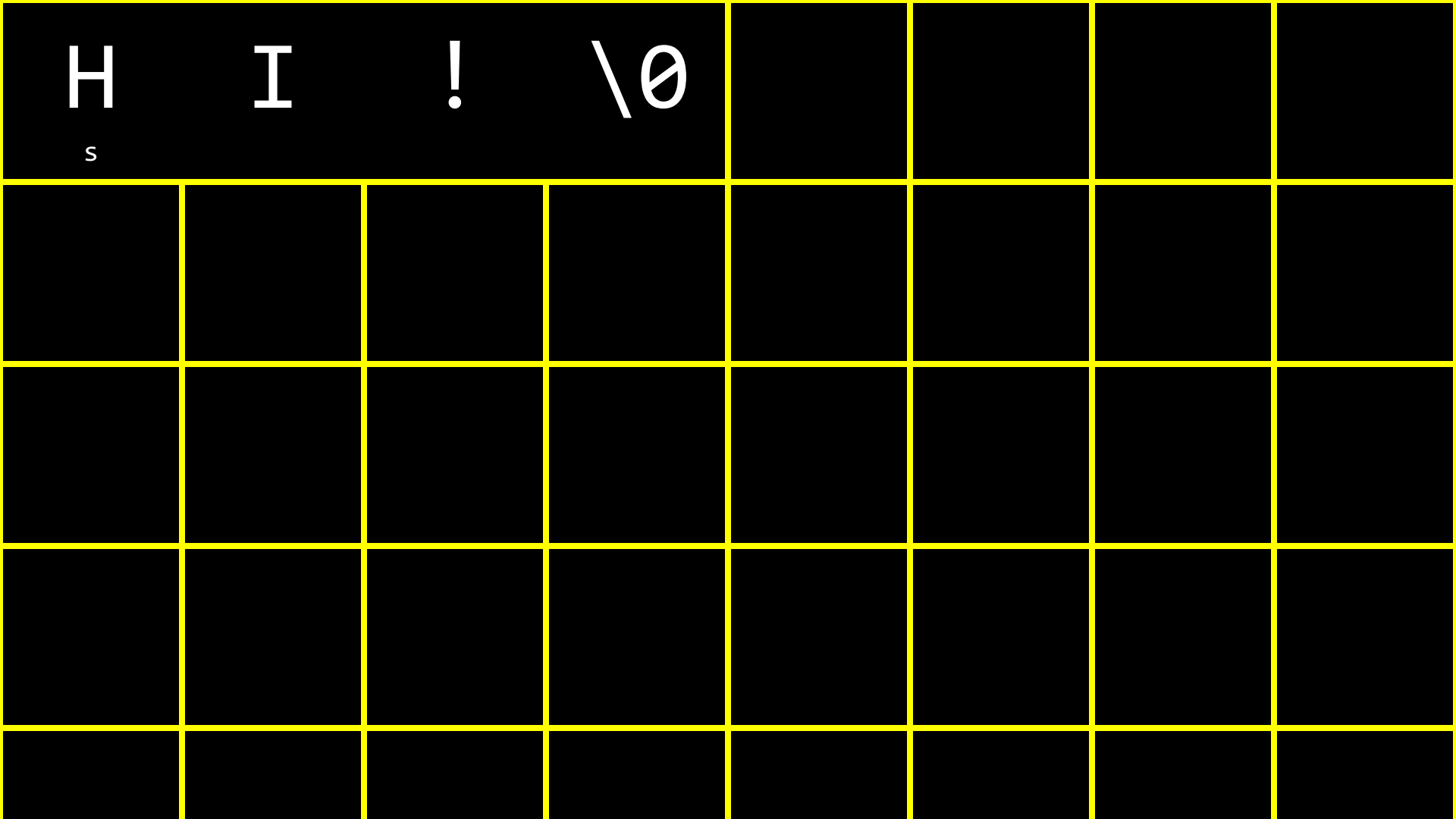

H
I
!
s[0]
s[1]
s[2]


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


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

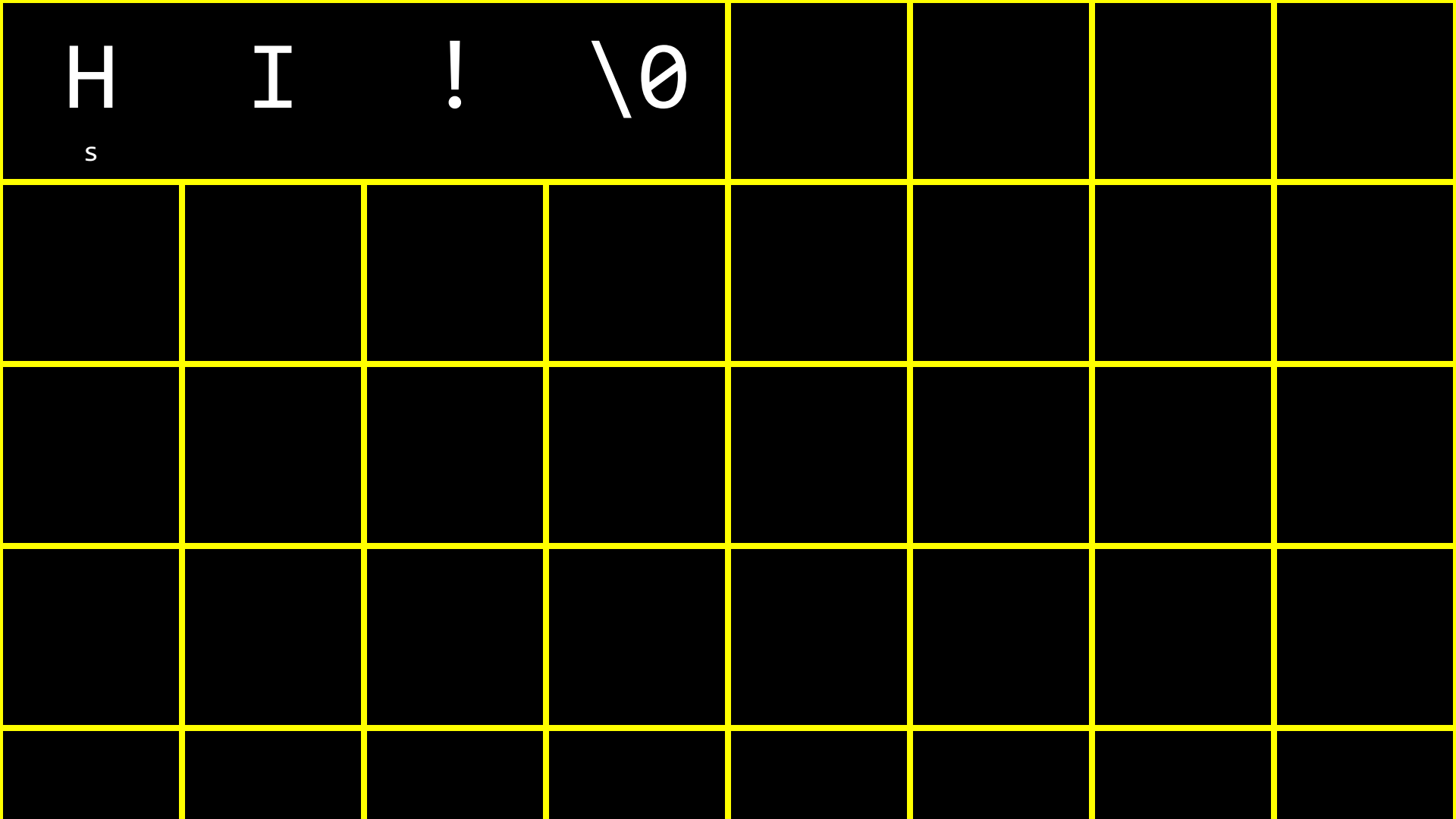

H I ! \0
s

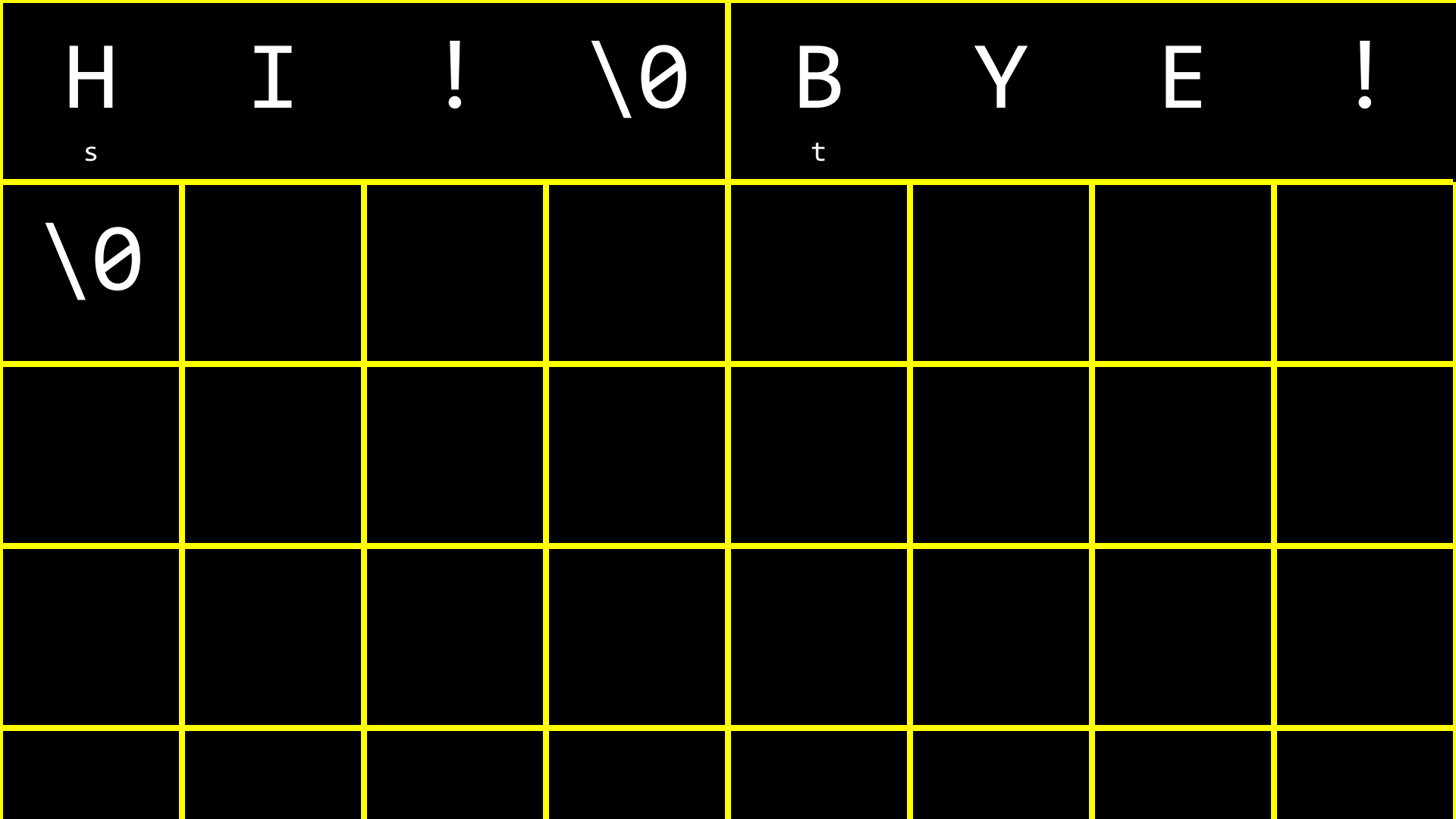
NUL

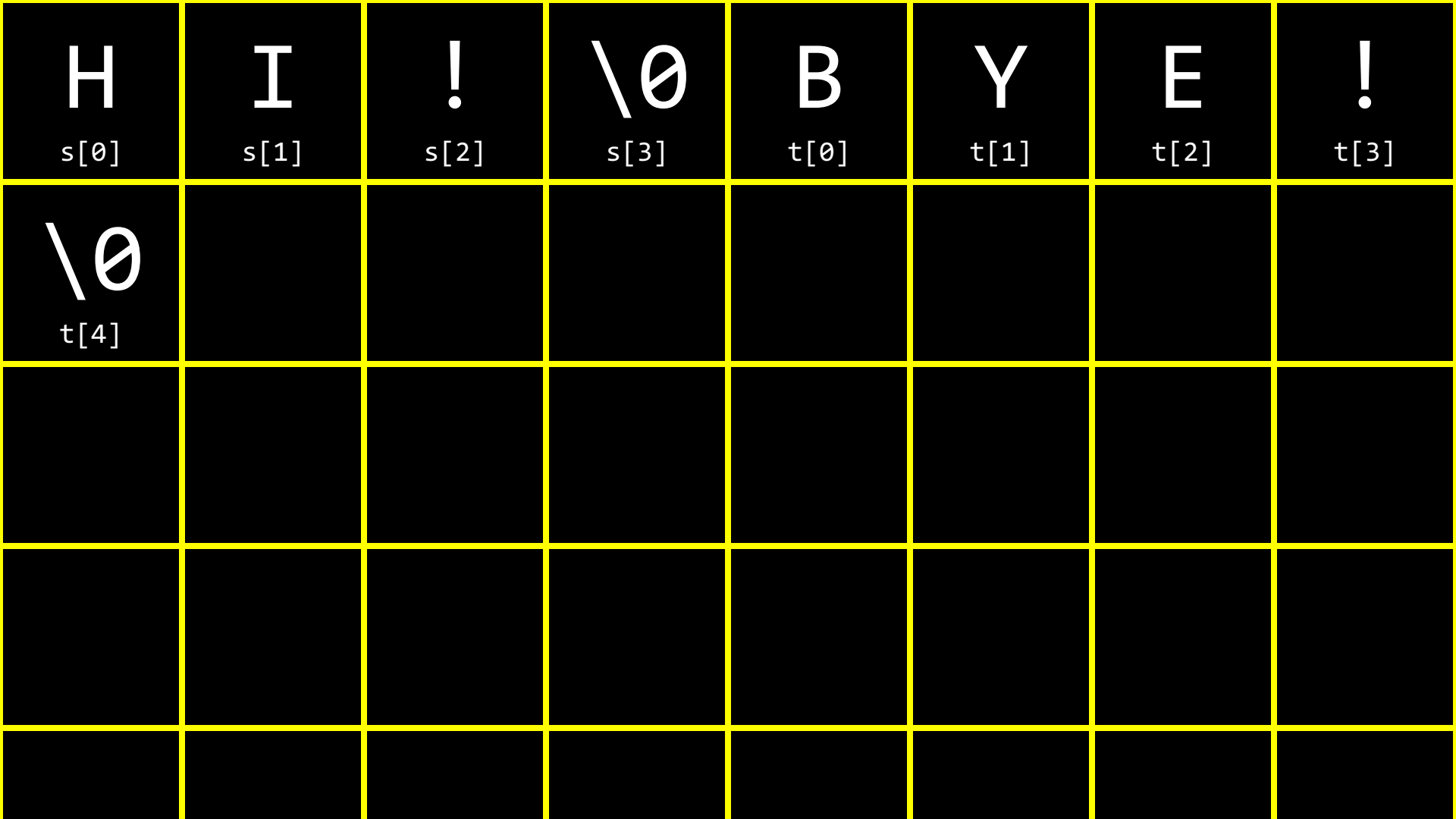
0	NUL	16	DLE	32	SP	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	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

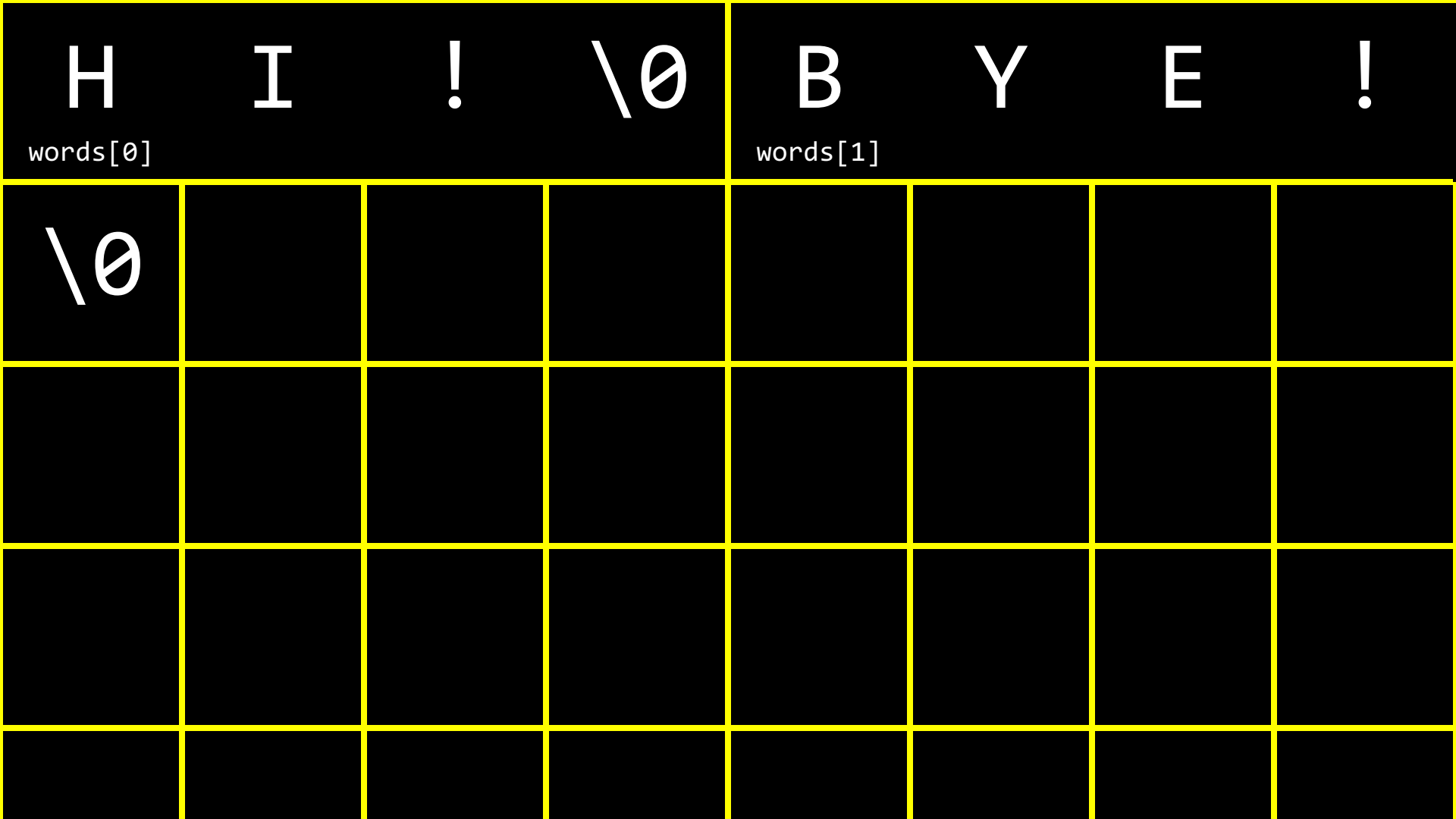
```
string s = "HI!";

string t = "BYE!";
```

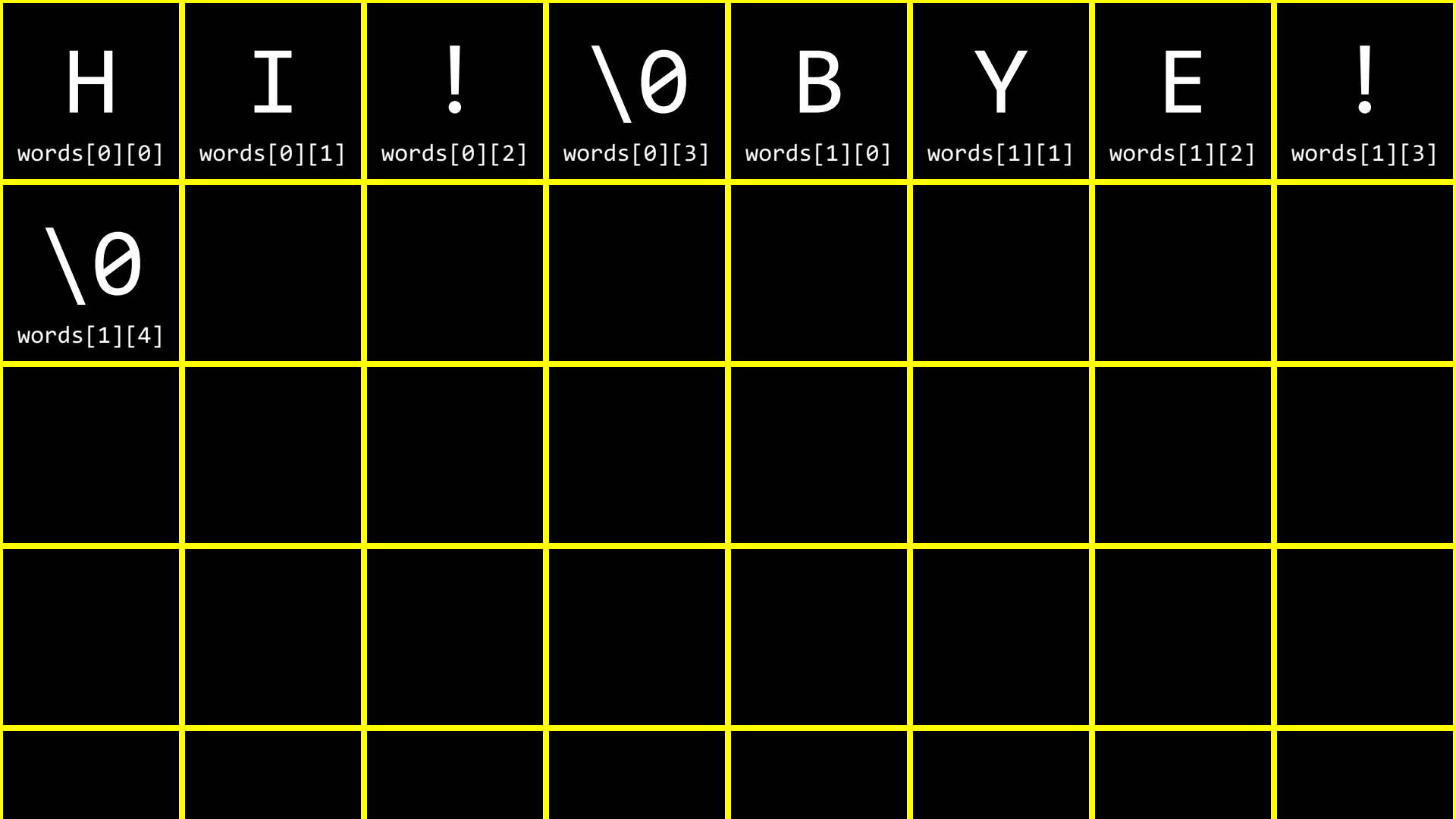

H I ! \0
s


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

PAUSE

10 min

Kommandozeilenparameter

```c
#include <stdio.h>

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

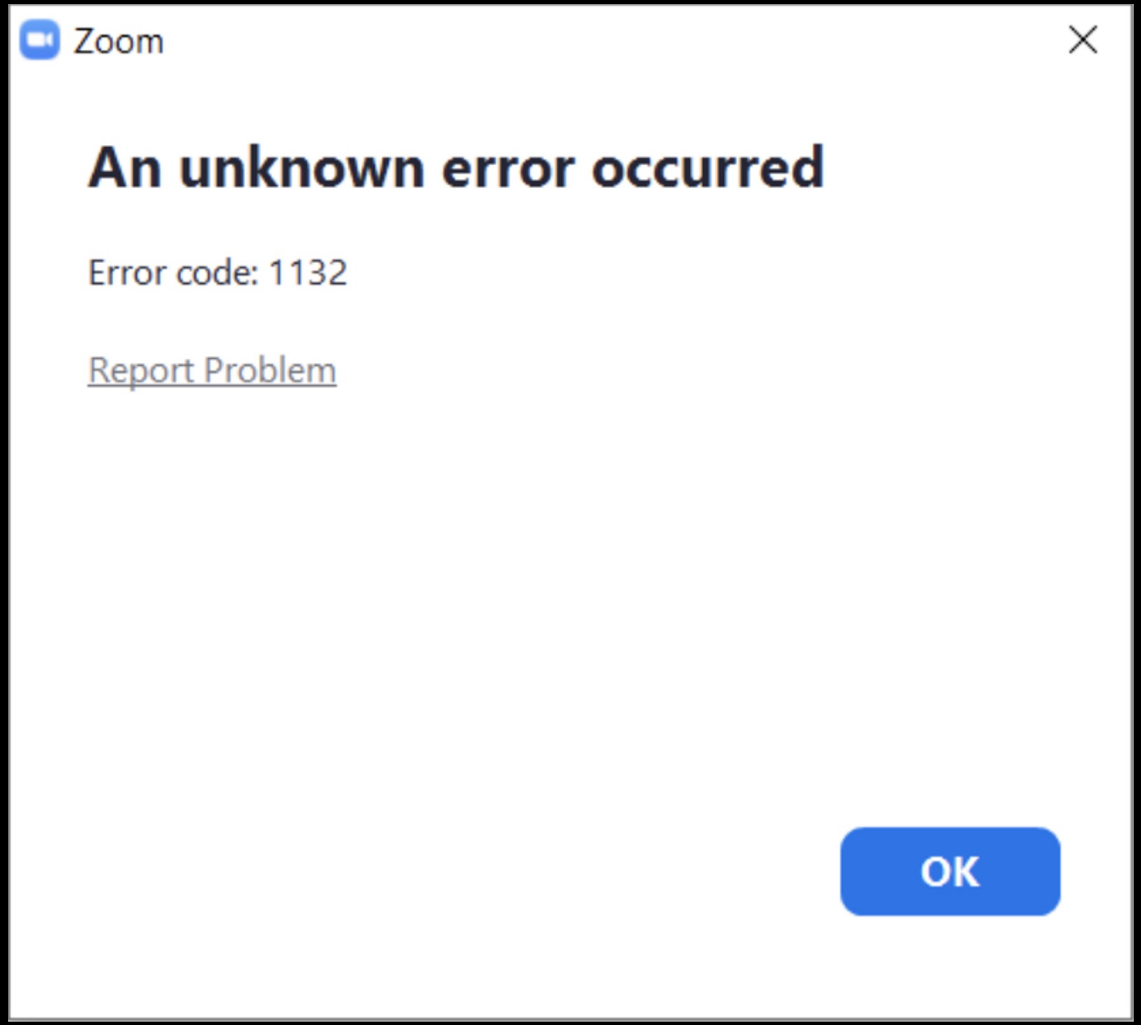
```c
#include <stdio.h>

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

```
#include <stdio.h>

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

ASCII Art

cowsay

exit status


Zoom
An unknown error occurred
Error code: 1132
Report Problem
OK


404
This is not the web page you are looking for.

```
#include <stdio.h>

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

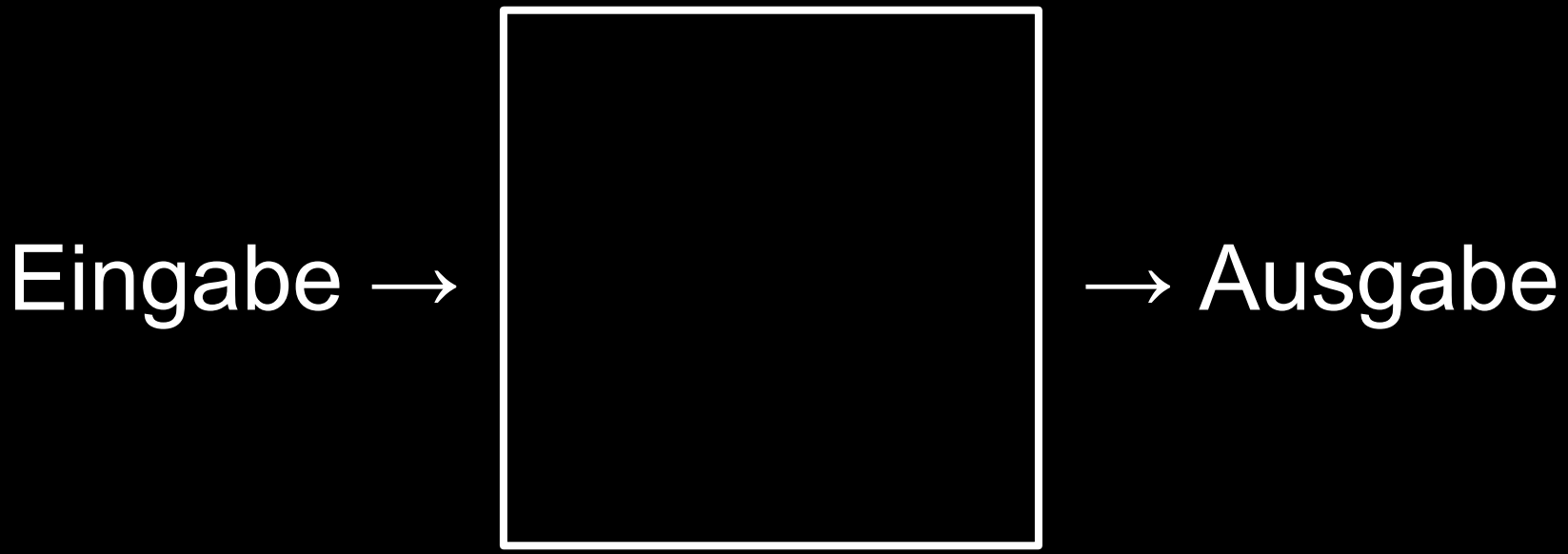
```
#include <stdio.h>

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

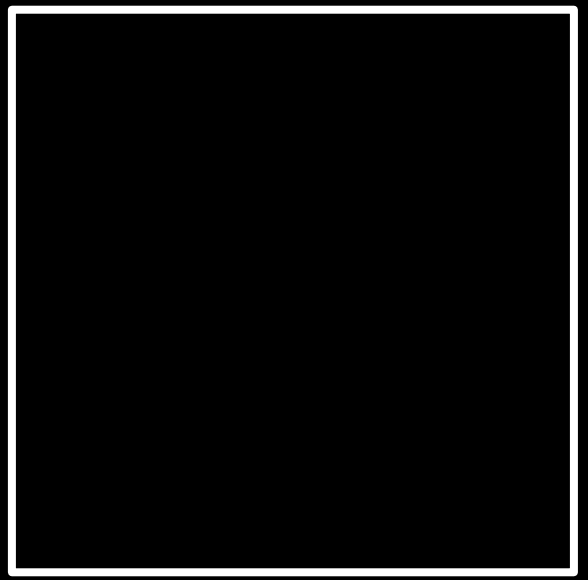
```c
#include <stdio.h>

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

```
echo $?
```

Kryptographie

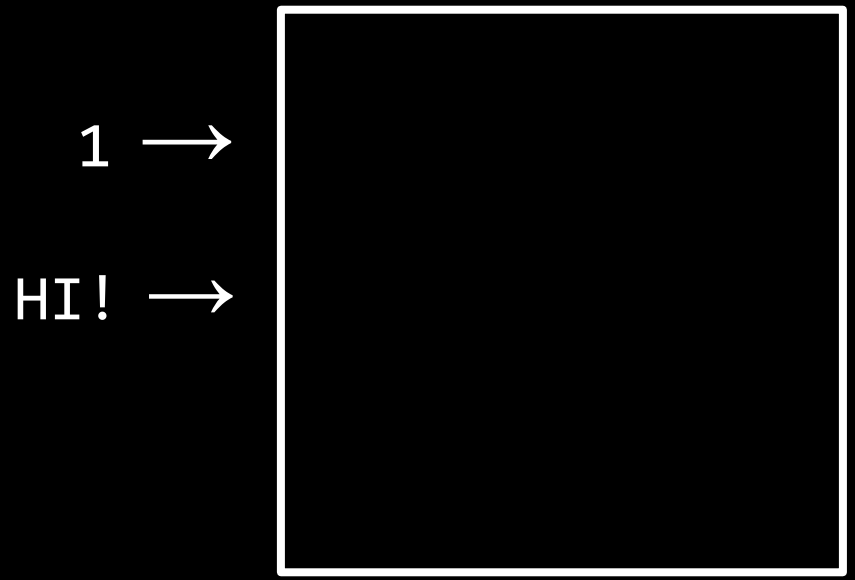
Verschlüsselung (encryption)

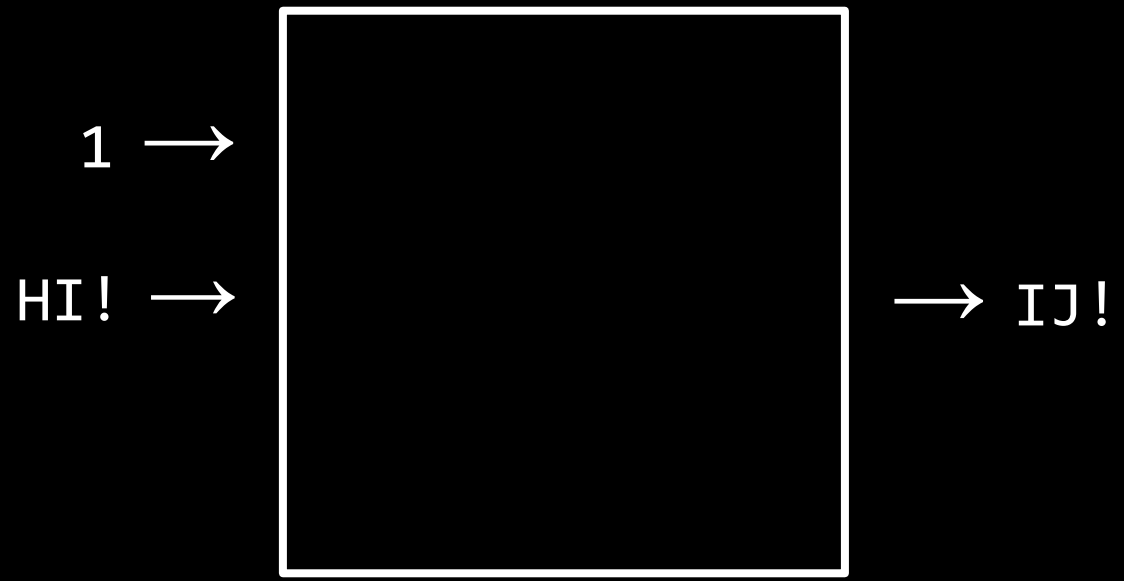

Eingabe →
→ Ausgabe

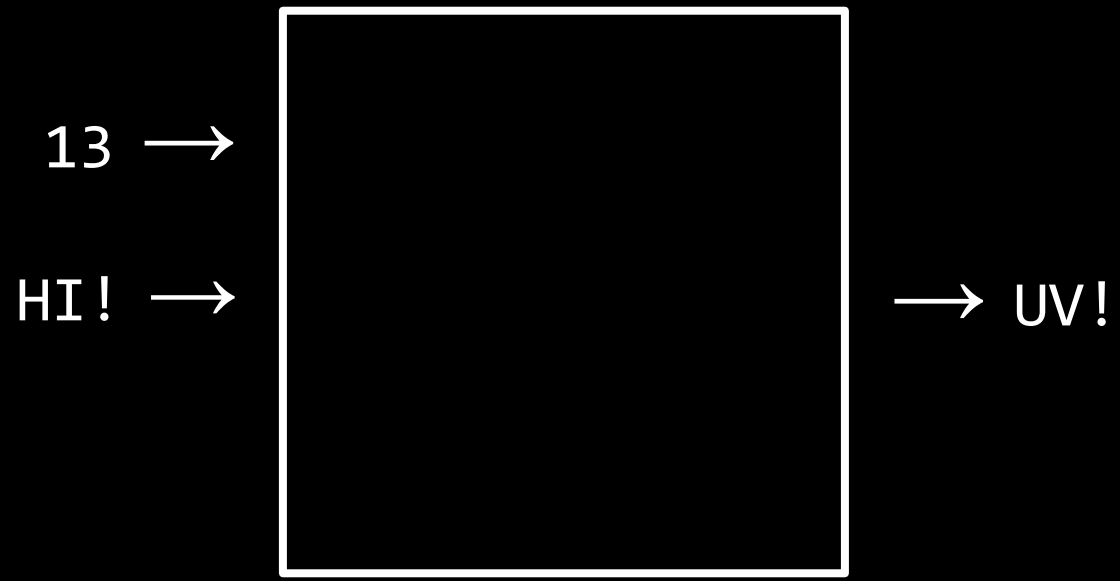
Klartext →  → Chiffretext


Klartext →
Chiffre
→ Chiffretext


Schlüssel →
Klartext →
Chiffre
→ Chiffretext

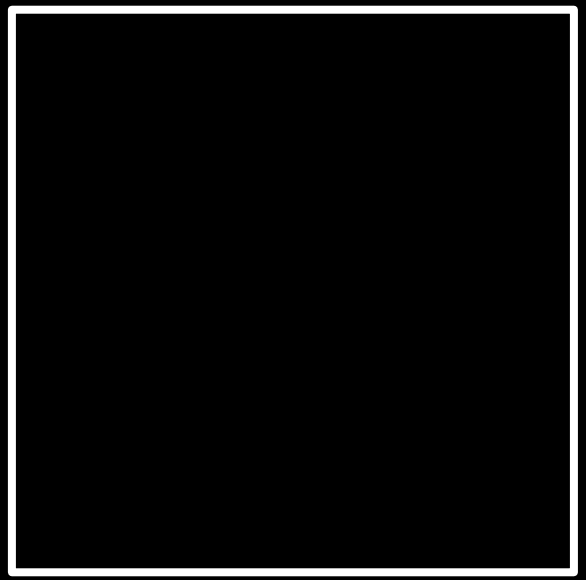

1
HI!


1
HI!
IJ!


13
HI!
UV!

13 ⟶

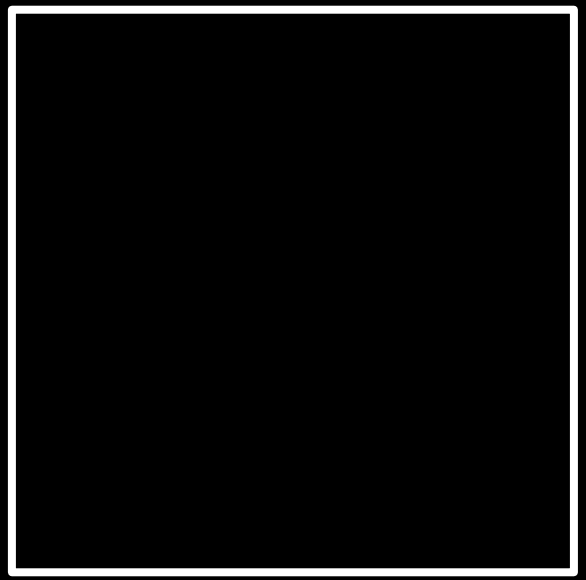
I LOVE YOU ⟶

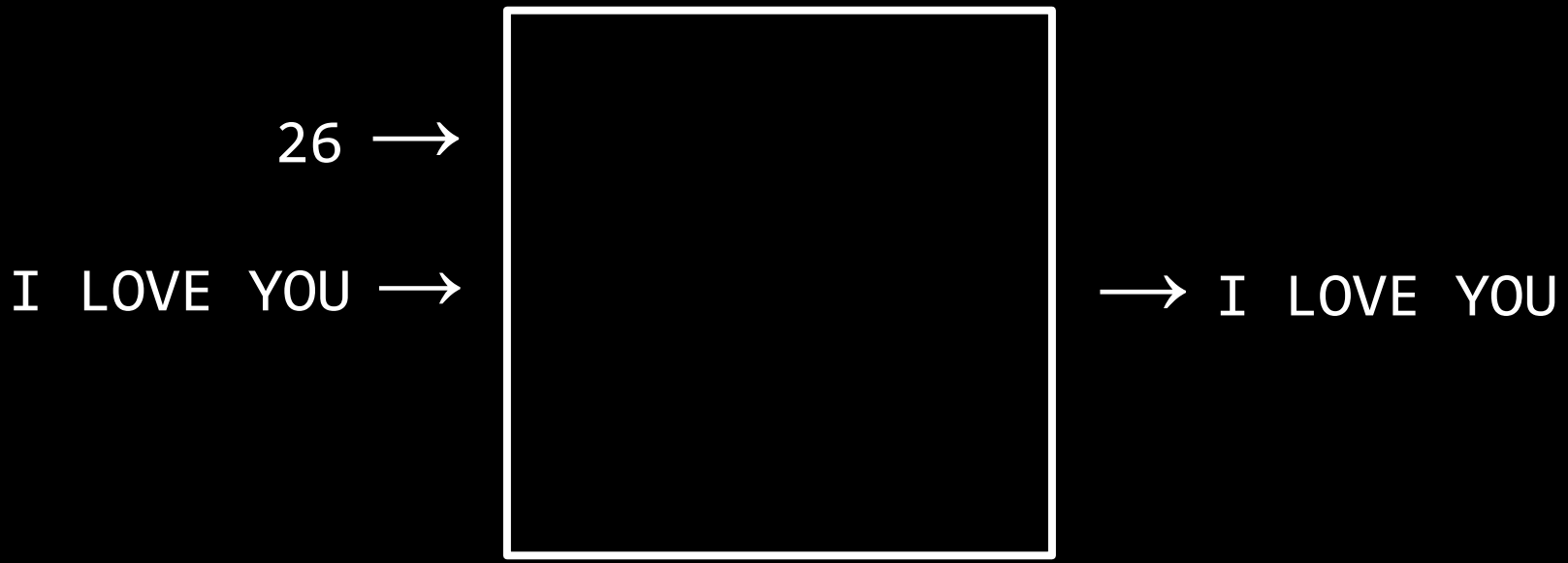


⟶ V YBIR LBH

26 ⟶

I LOVE YOU ⟶

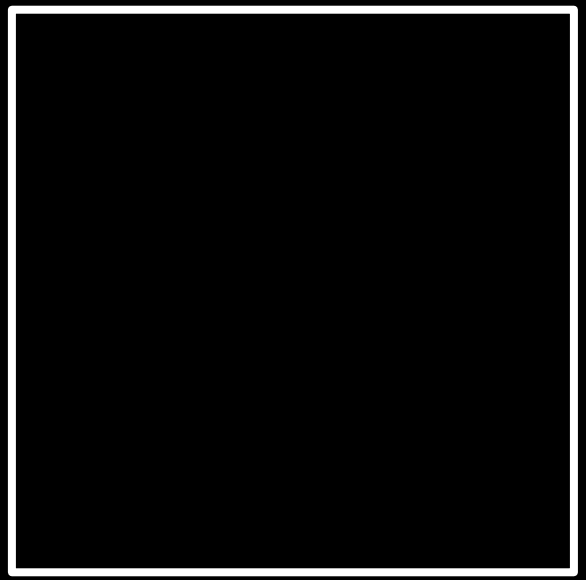



26 ⟶
I LOVE YOU ⟶
⟶ I LOVE YOU

Entschlüsselung (decryption)

-3 ⟶

GLHV ZDU LQIHLQIE ⟶



GLHV ZDU LQIHLQIE

DLHV ZDU LQIHLQIE

DIHV ZDU LQIHLQIE

DIEV ZDU LQIHLQIE

DIES ZDU LQIHLQIE

DIES WDU LQIHLQIE

DIES WAU LQIHLQIE

DIES WAR LQIHLQIE

DIES WAR IQIHLQIE

DIES WAR INIHLQIE

DIES WAR INFHLQIE

DIES WAR INFELQIE

DIES WAR INFEIQIE

DIES WAR INFEINIE

DIES WAR INFEINFE

DIES WAR INFEINFB

Dies war Inf-Einf-B.