

IC 2022.1

```
#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

```
#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);
}
```

```
...
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
0000000000000000000000000000000000
000000000000000000000000000000000000
0000000010000000000111110000000000
000000010000000000000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000000000
101000000000000100000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000000000
010000000000000000000000000000000000
000000000000000100000000000000000000
0000101000000000000000000000000000000
01010101010010001000100111100101
01001000100000111110110000010000
00110001110000001000100111000111
0100100010111110000000000000000000000
00000000000000000000000000000000000000
00000000000000000000000000000000000000
111010000000000000000000000000000000000
000000000000000000000000000000000000000
000000000000000000000000000000000000000
000000000000000000000000000000000000000

1

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);
}
```

```
#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);
}
```

```
#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

011111110100010100100110001000110

0000001000000001000000010000000

00000000000000000000000000000000

000000010000000000011111000000000

00000000000000000000000000000000

000

10100000000000100000000000000000

0101010100100010001001111100101

01001000100000111110110000010000

00110001110000001000100111000111

010010001011111000000000000000000000000

0000000000000001011000000000000

00000000010010001011111100000000

Digitized by srujanika@gmail.com

cs50.c

stdio.c

1

stdio.c

| | | |
|----------------------------------|----------------------------------|-----------------------------------|
| 01111111010001010100110001000110 | 0111111010001010100110001000110 | 00101111011011000110100101100010 |
| 000000100000001000000010000000 | 000000100000001000000010000000 | 01100011001011100111001101101111 |
| 00000000000000000000000000000000 | 00000000000000000000000000000000 | 0010111000110110001000000101111 |
| 00000000000000000000000000000000 | 00000000000000000000000000000000 | 01110101011100110111001000101111 |
| 000000010000000001111000000000 | 000000110000000001111000000000 | 01101100011010010110001000101111 |
| 00000001000000000000000000000000 | 00000010000000000000000000000000 | 0111000001110000011011001011111 |
| 00000000000000000000000000000000 | 11000000000111100000000000000000 | 0011011000110100010110101101100 |
| 00000000000000000000000000000000 | 00000000000000000000000000000000 | 01101001011011100111010101111000 |
| 00000000000000000000000000000000 | 01000000000000000000000000000000 | 0010110101100111010111001110101 |
| 00000000000000000000000000000000 | 00000000000000000000000000000000 | 0010111011011000110100101100010 |
| 10100000000001000000000000000000 | 00101000011001000000000000000000 | 0110001101011110110111001101111 |
| 00000000000000000000000000000000 | 00000000000000000000000000000000 | 0110110011100110110100001100001 |
| 00000000000000000000000000000000 | 00000000000000000000000000000000 | 011100100110010101100010000101110 |
| 01000000000000000000000000000000 | 01000000000000000000000000000000 | 0110000100100000010000001000001 |
| 00000000000000001000000000000000 | 00000111000000000100000000000000 | 0101001101011110100111001000101 |
| 00001010000000000000000000000000 | 00011100000000000000000000000000 | 01000101010001000100010101000100 |
| 01010101010010001000100111100101 | 00000001000000000000000000000000 | 0010000001010000010000000101111 |
| 0100100010000011111011000010000 | 00000101000000000000000000000000 | 01101100011010010110001000101111 |
| 00110001110000001000100111000111 | 00000000000000000000000000000000 | 0111000001110000011011001011111 |
| 01001000101111100000000000000000 | 00000000000000000000000000000000 | 0011011000110100010110101101100 |
| 00000000000000000000000000000000 | 00000000000000000000000000000000 | 0110100101101110011101010111100 |
| 00000000000000000000000000000000 | 00000000000000000000000000000000 | 0010110101100111010111001110101 |
| 11101000000000000000000000000000 | 00000000000000000000000000000000 | 00101111011011000110010000101101 |
| 00000000100100010111110000000000 | 00000000000000000000000000000000 | 01101100011010010110111001110101 |
| 00000000000000000000000000000000 | 01011100001001010000000000000000 | 0111000001011010111100000111000 |
| 00000000000000000000000000000000 | 00000000000000000000000000000000 | 0011011000101101011100001110100 |

...

...

...

preprocessing

compiling

assembling

linking

compiling

debugging

92

9/9

0800 Antran started
 1000 . stopped - antran ✓
 13° UC (032) MP - MC
 (033) PRO 2
 convt
 2.130476415
 2.130676415

{ 1.2700 9.037847025
 9.037846995 const
~~1.982677000~~
~~2.130776415~~ (23) 4.615925059 (-)

Relays 6-2 in 033 failed special speed test
 in relay 10.000 test.

Relay 2145
Relay 337

1100 Started Cosine Tape (Sine check)
 1525 Started Multi+ Adder Test.

1545



Relay #70 Panel F
 (moth) in relay.

First actual case of bug being found.
 1600 Antran started.

1700 closed down.

In relay 11.00 test.

(Relays changed)

1100 Started Cosine Tape (Sine check)
1525 Started Multi Adder Test.

1545

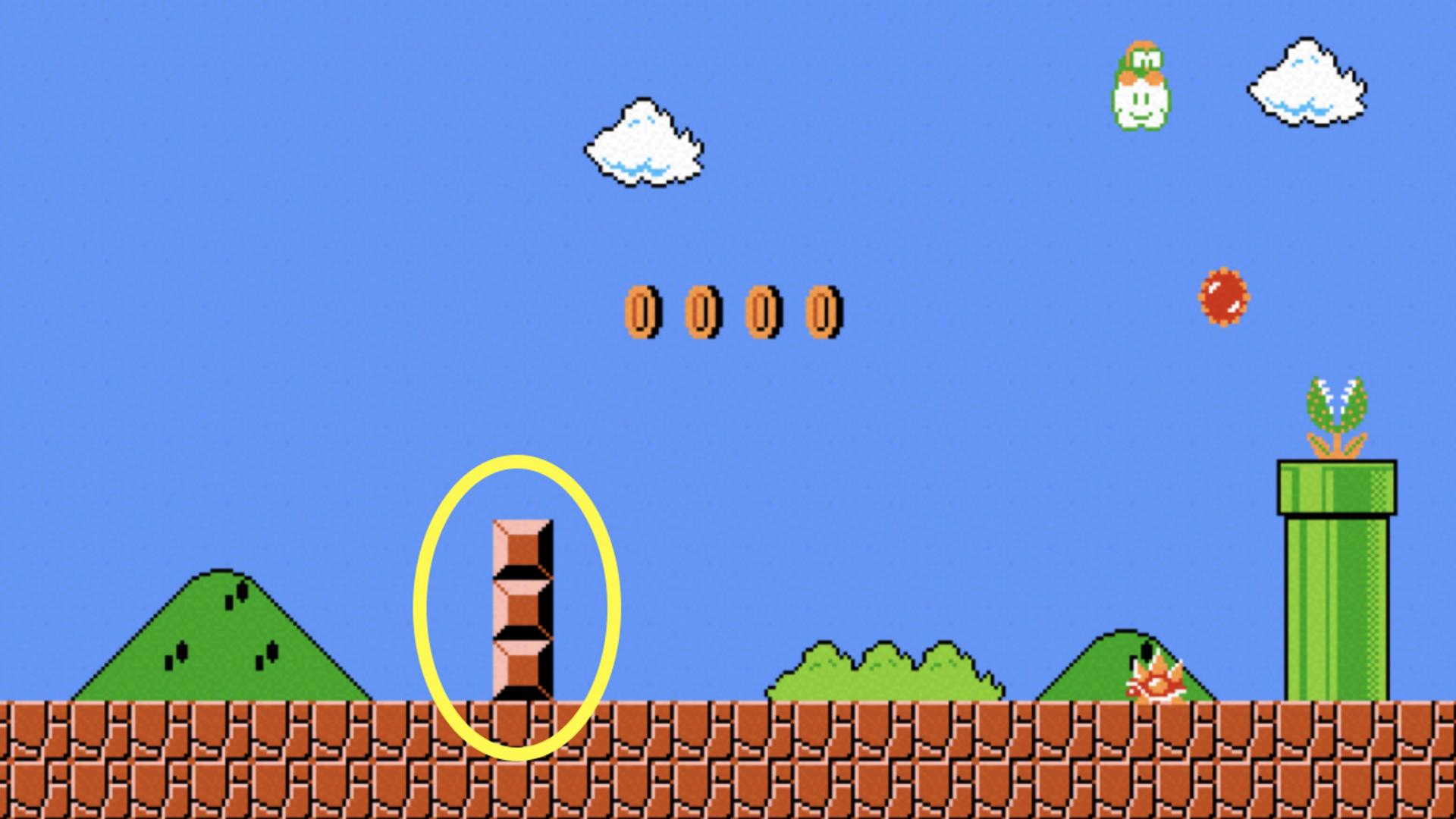


Relay #70 Panel F
(moth) in relay.

First actual case of bug being found.

1630 Arctangent started.

1700 Closed down.



0 0 0 0

```
printf
```

printf

debugger

printf

debugger

rubber duck

types

bool 1 byte

char 1 byte

double 8 bytes

float 4 bytes

int 4 bytes

long 8 bytes

string ? bytes

...





8BB12
D9HXT

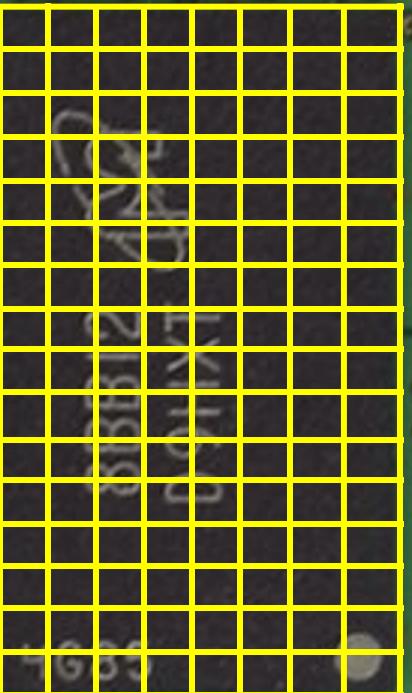
4G85

8BB12
D9HXT

4G85

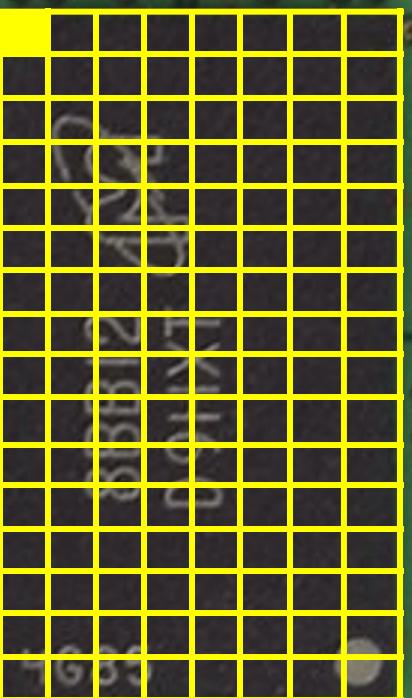
8BB12
D9HXT

4G85



8BB12
D9HXT

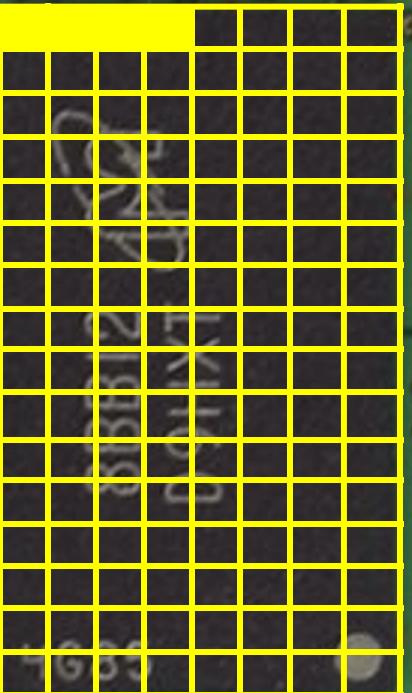
4G85

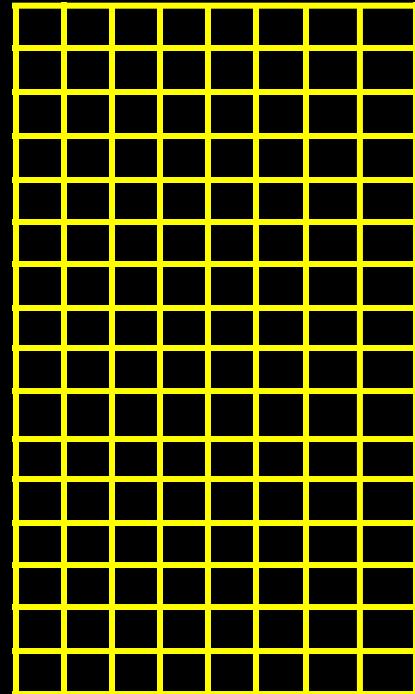


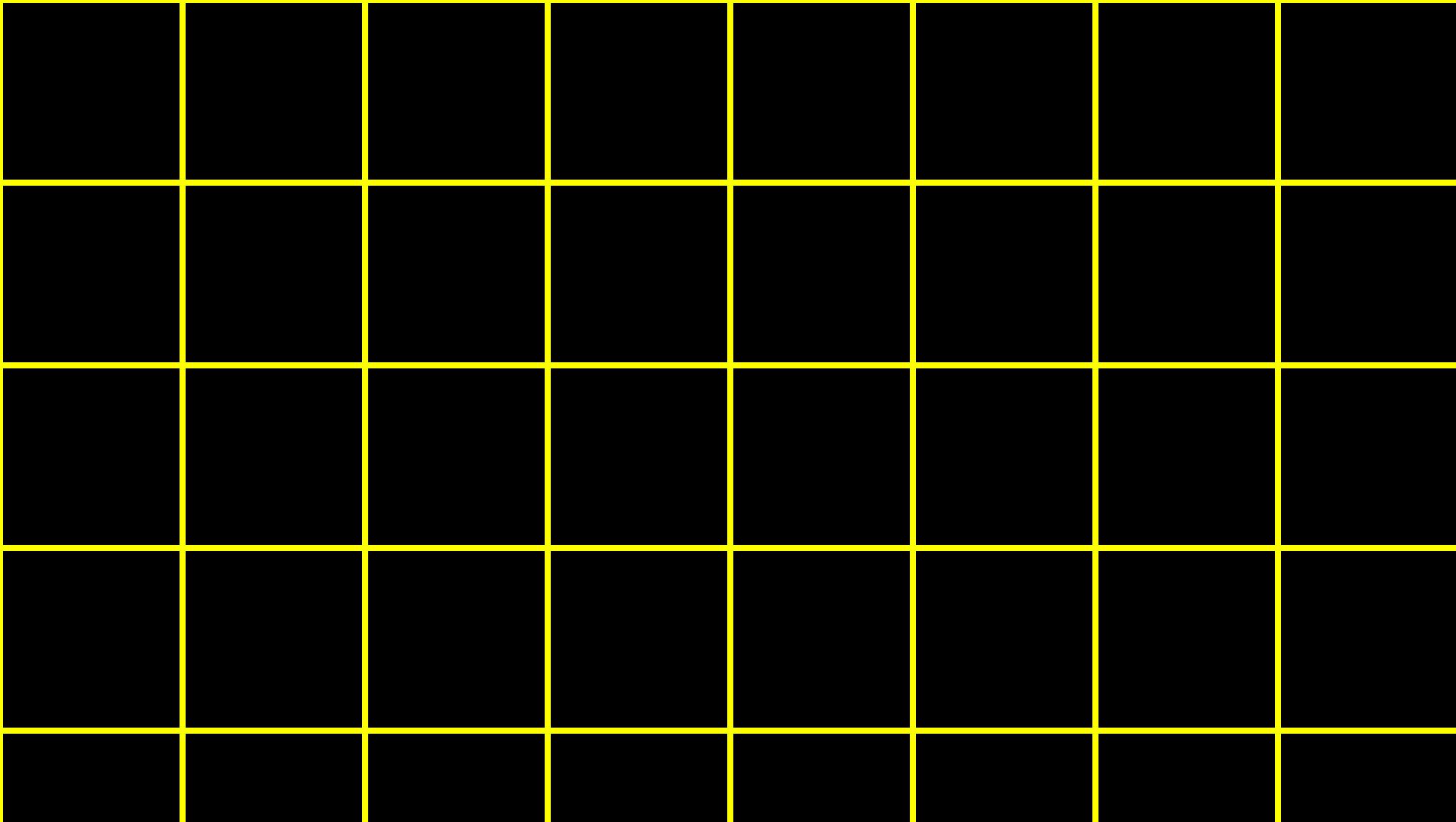
4G85
D9HXT

8BB12
D9HXT

4G85







```
int score1 = 72;
```

```
int score2 = 73;
```

```
int score3 = 33;
```

72

score1

72

score1

73

score2

72

score1

73

score2

33

score3

000000000000000000000000000000001001000

score1

score2

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]

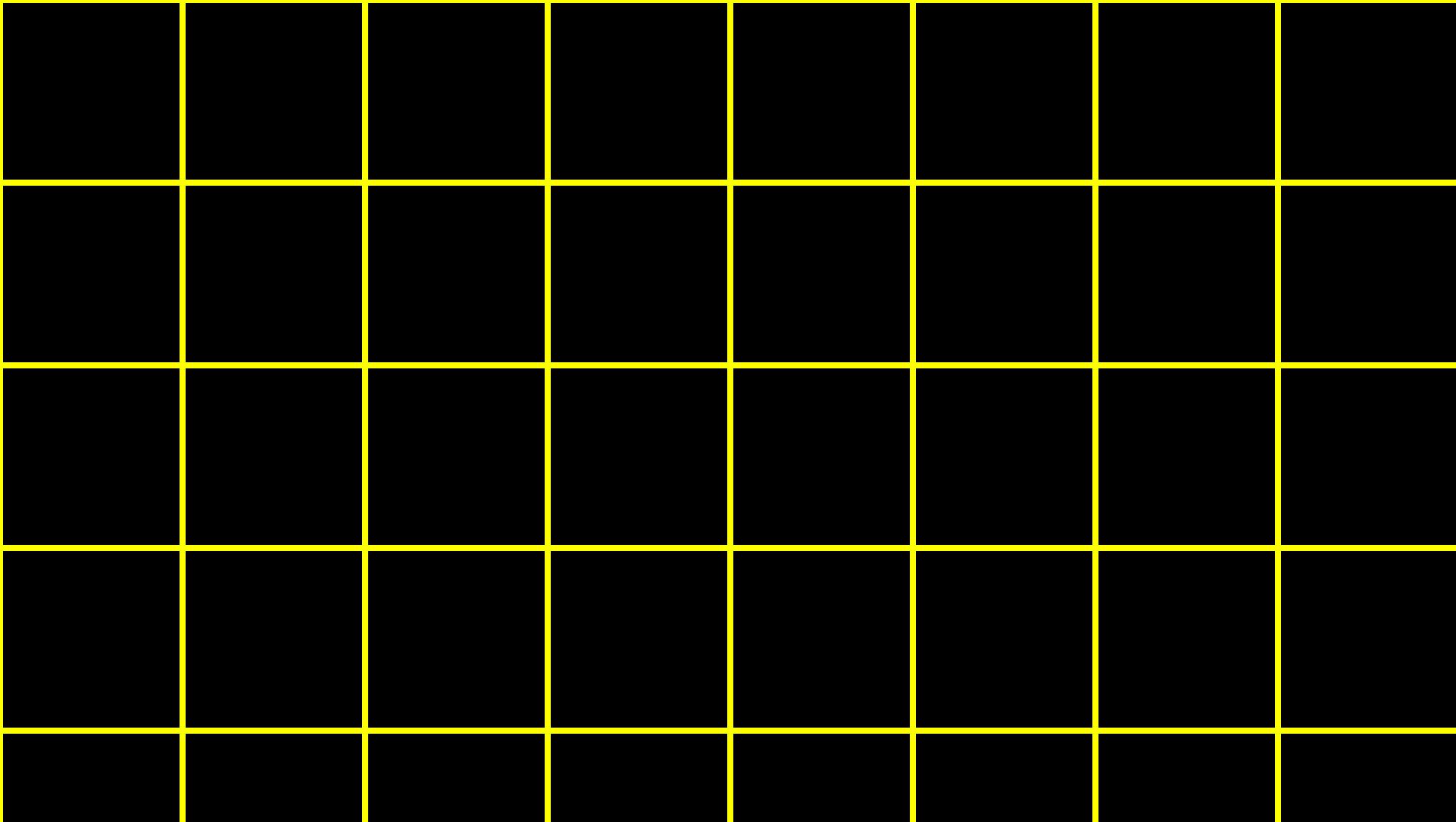
33

scores[2]

```
char c1 = 'H';
```

```
char c2 = 'I';
```

```
char c3 = '!';
```



H

c1

I

c2

!

c3

72

c1

73

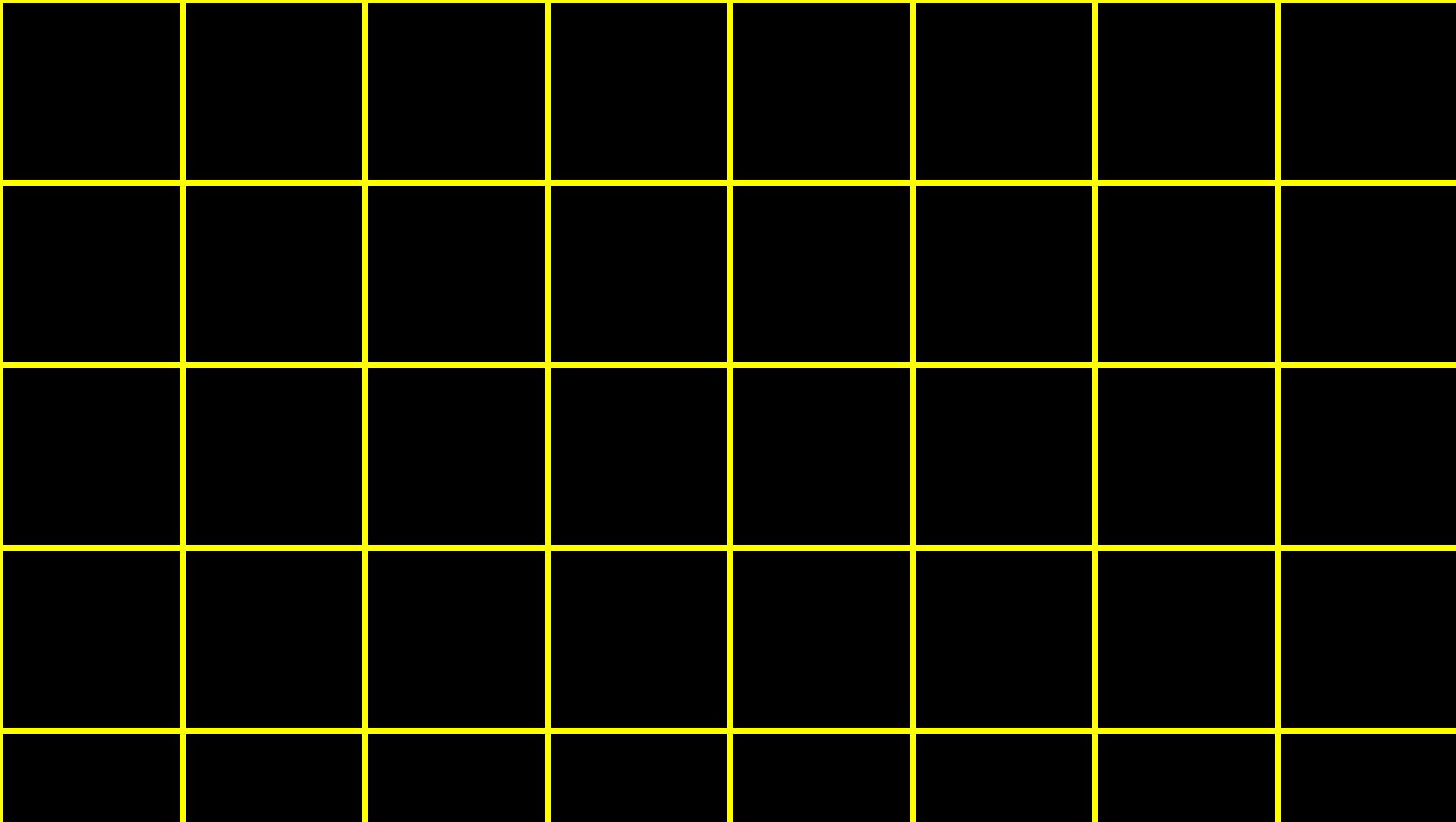
c2

33

c3

| | | | | | | | |
|----------|----------|----------|--|--|--|--|--|
| 01001000 | 01001001 | 00100001 | | | | | |
| c1 | c2 | c3 | | | | | |
| | | | | | | | |
| | | | | | | | |

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



H

I

!

s

H

$s[0]$

I

$s[1]$

!

$s[2]$

H

$s[0]$

I

$s[1]$

!

$s[2]$

\theta

$s[3]$

72

$s[0]$

73

$s[1]$

33

$s[2]$

0

$s[3]$

H

I

!

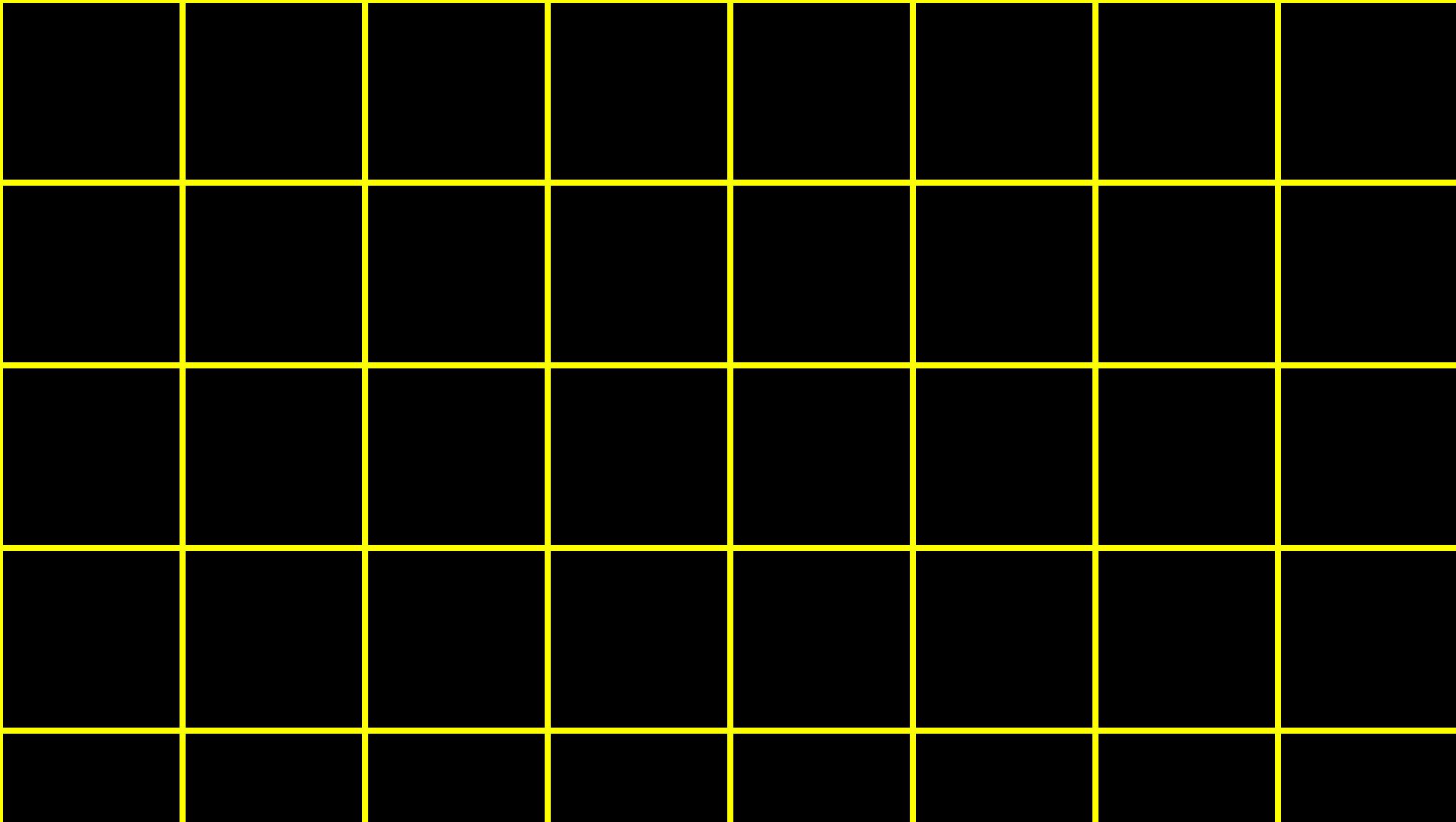
\theta

s

NUL

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

```
string t = "BYE!";
```



H

I

!

\theta

s

H

I

!

\theta

s

B

Y

E

!

t

\theta

H

s[0]

I

s[1]

!

s[2]

\theta

s[3]

B

t[0]

Y

t[1]

E

t[2]

!

t[3]

\theta

t[4]

```
string words[2];
```

```
words[0] = "HI!";
```

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

string

manual pages

command-line arguments

```
#include <stdio.h>

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

```
#include <stdio.h>

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

```
#include <stdio.h>

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

exit status

```
#include <stdio.h>

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

```
#include <stdio.h>

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

readability

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

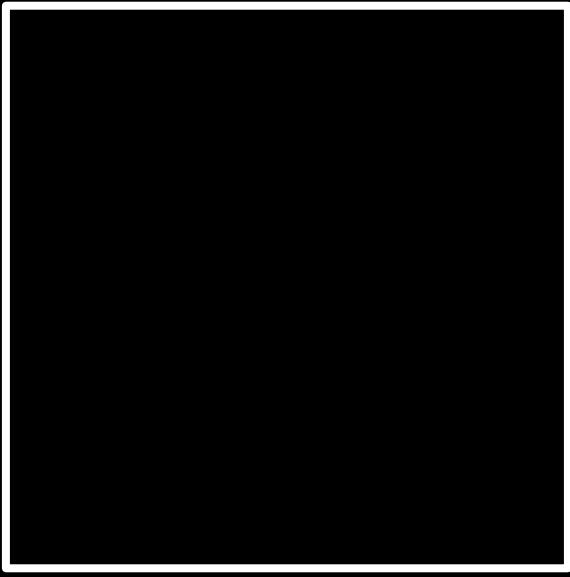
Before Grade 1

Mr. and Mrs. Dursley, of number four, Privet Drive, were proud to say that they were perfectly normal, thank you very much. They were the last people you'd expect to be involved in anything strange or mysterious, because they just didn't hold with such nonsense...

Grade 7

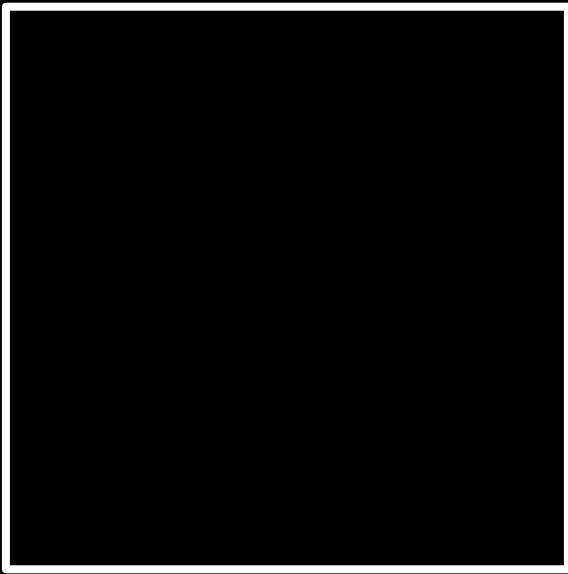
cryptography

input →



→ output

plaintext →

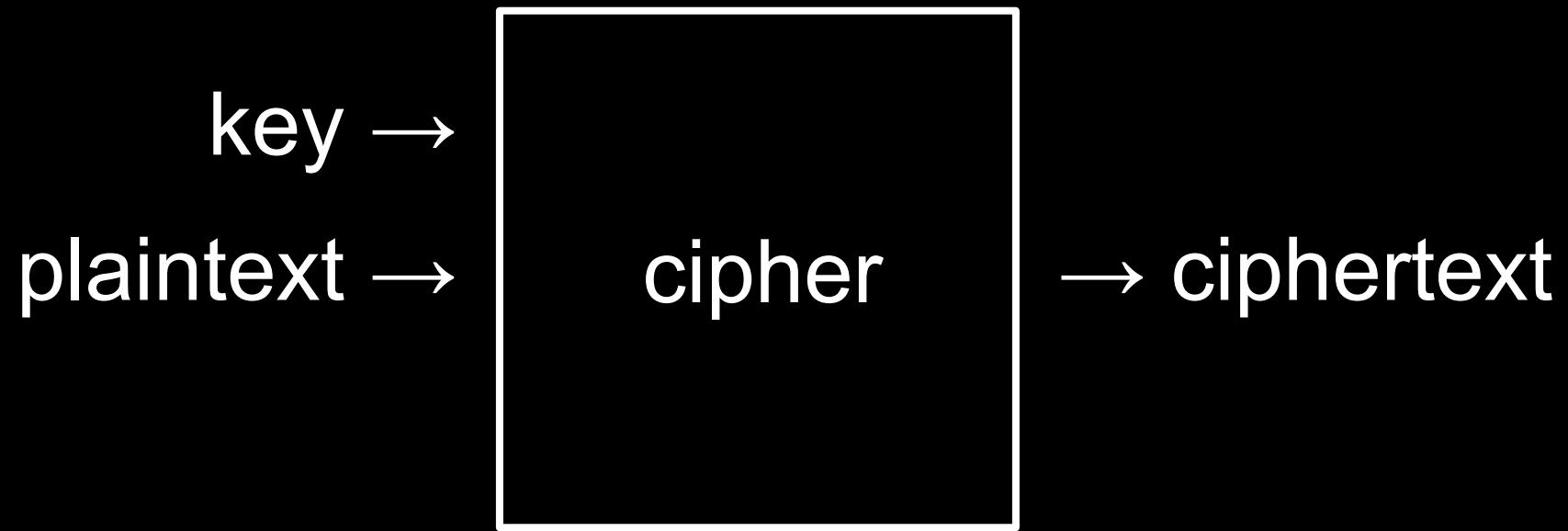


→ ciphertext

plaintext →

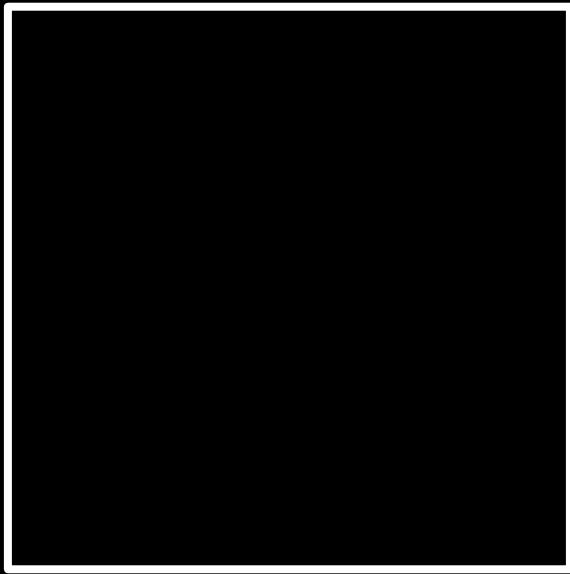
cipher

→ ciphertext



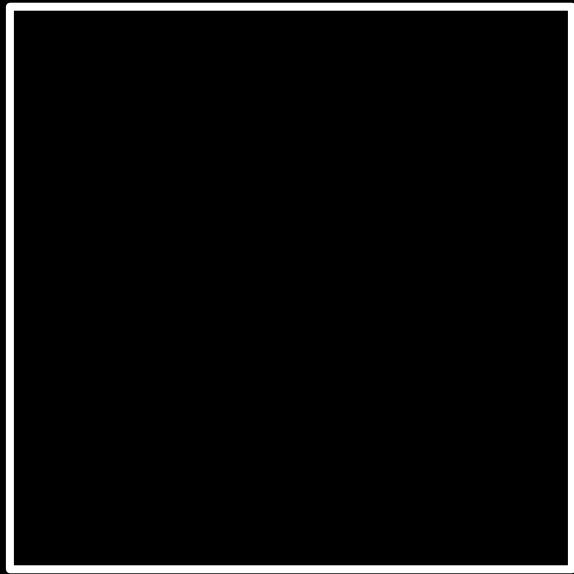
1 →

HI! →



$1 \rightarrow$

$H! \rightarrow$



$\rightarrow IJ!$

-1 →

UIJT XBT DT50 →

