This is CS50


when clicked
say
hello, world

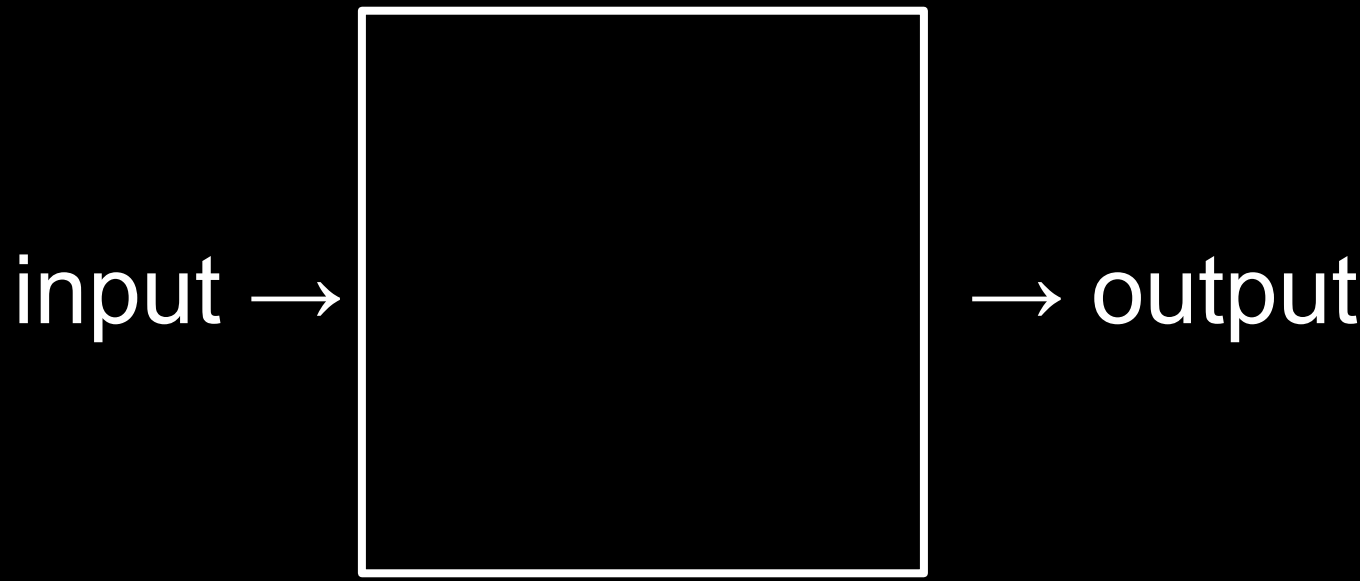
```c
#include <stdio.h>

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

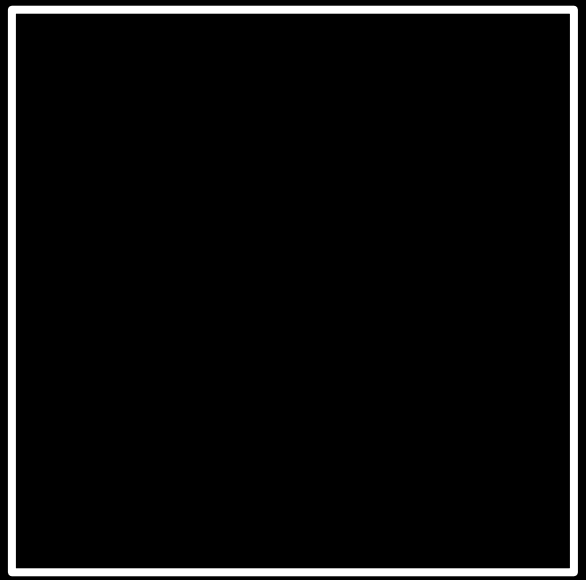
```c
#include <stdio.h>

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

source code

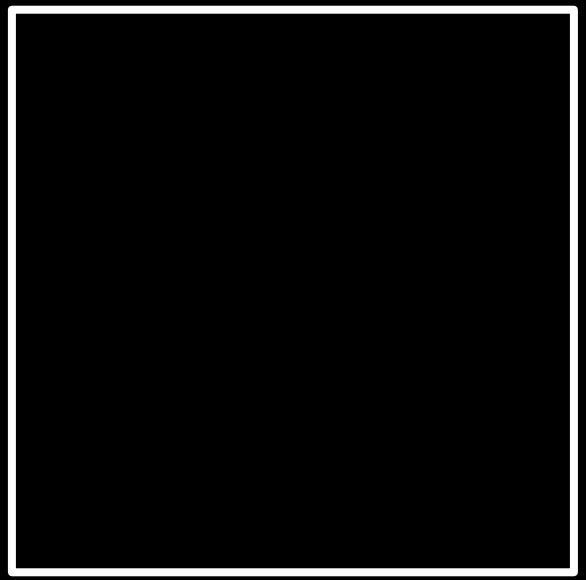
```
01111111 01000101 01001100 01000110 00000010 00000001 00000001 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000010 00000000 00111110 00000000 00000001 00000000 00000000 00000000
10110000 00000101 01000000 00000000 00000000 00000000 00000000 00000000
01000000 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
...
```

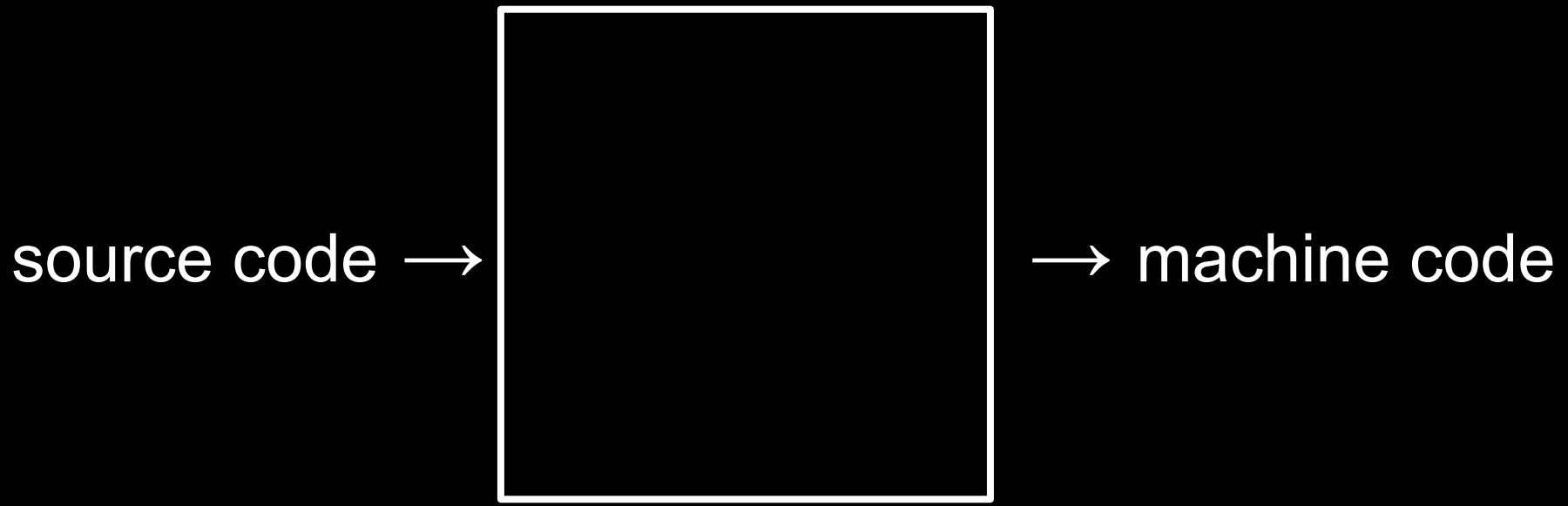
machine code

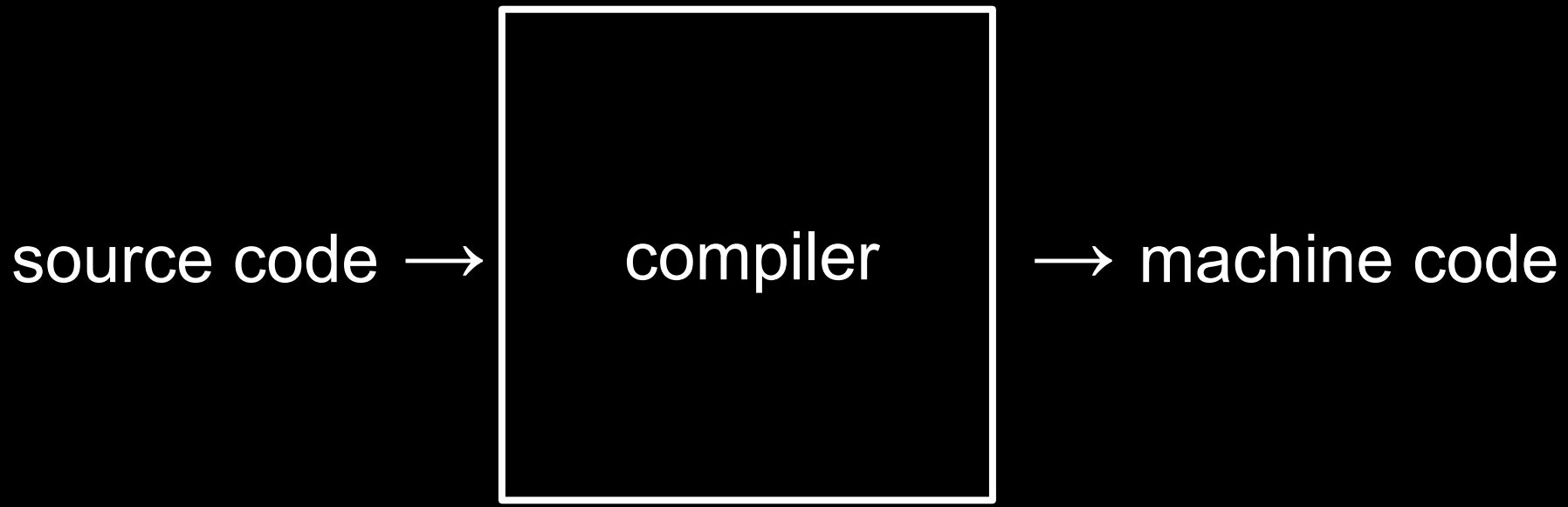

input →
→ output



source code ⟶

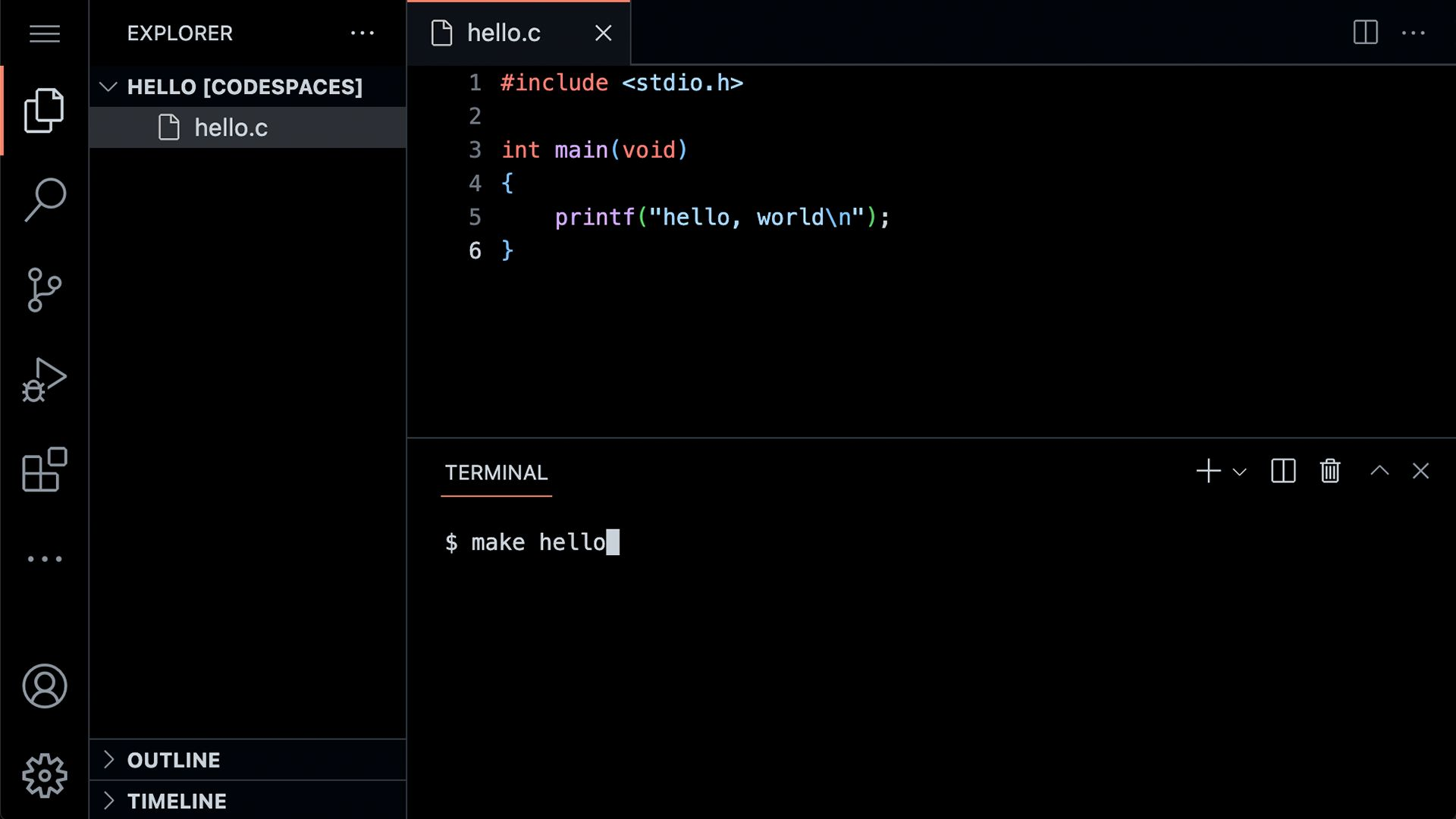


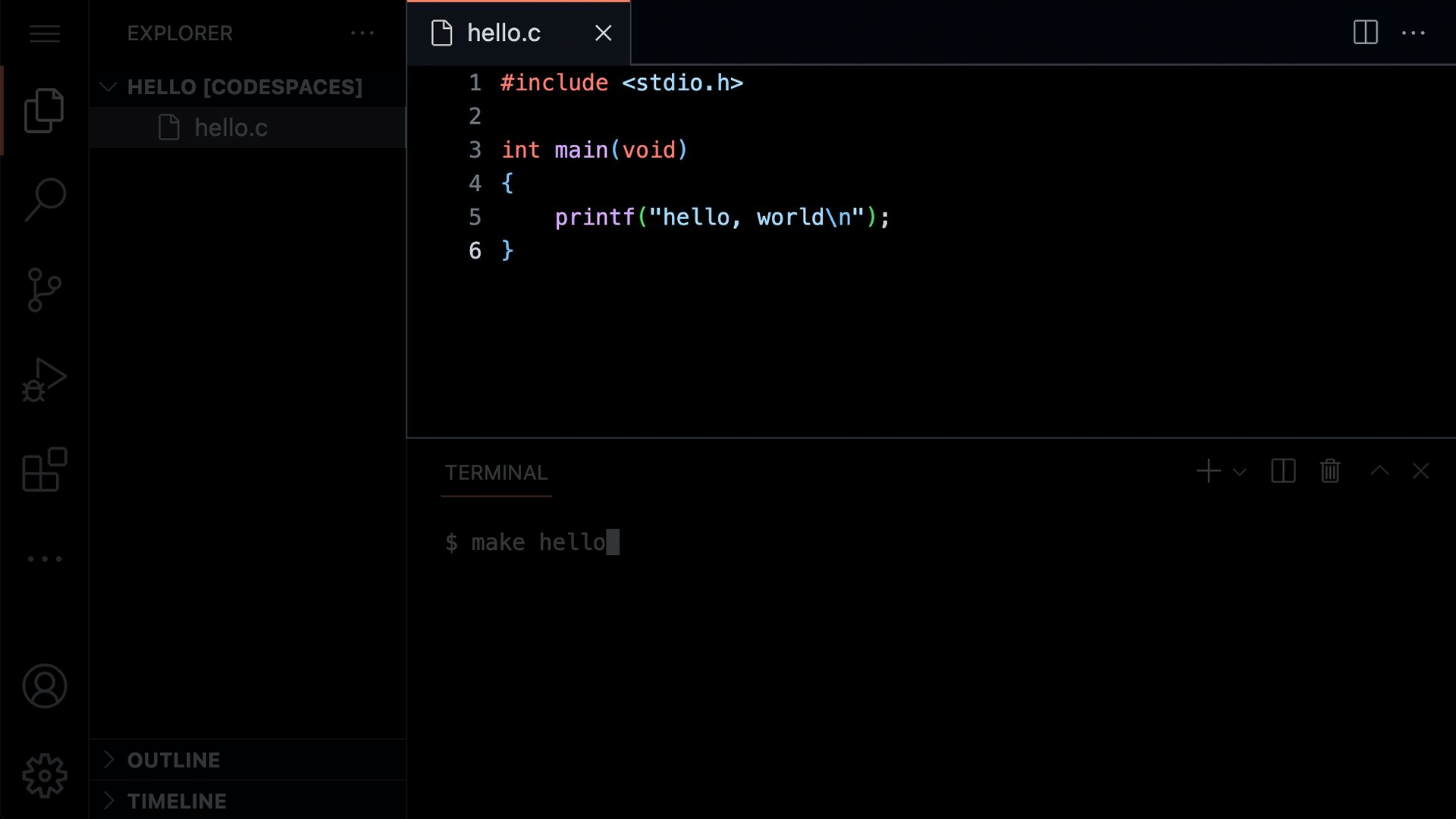

source code ⟶
⟶ machine code

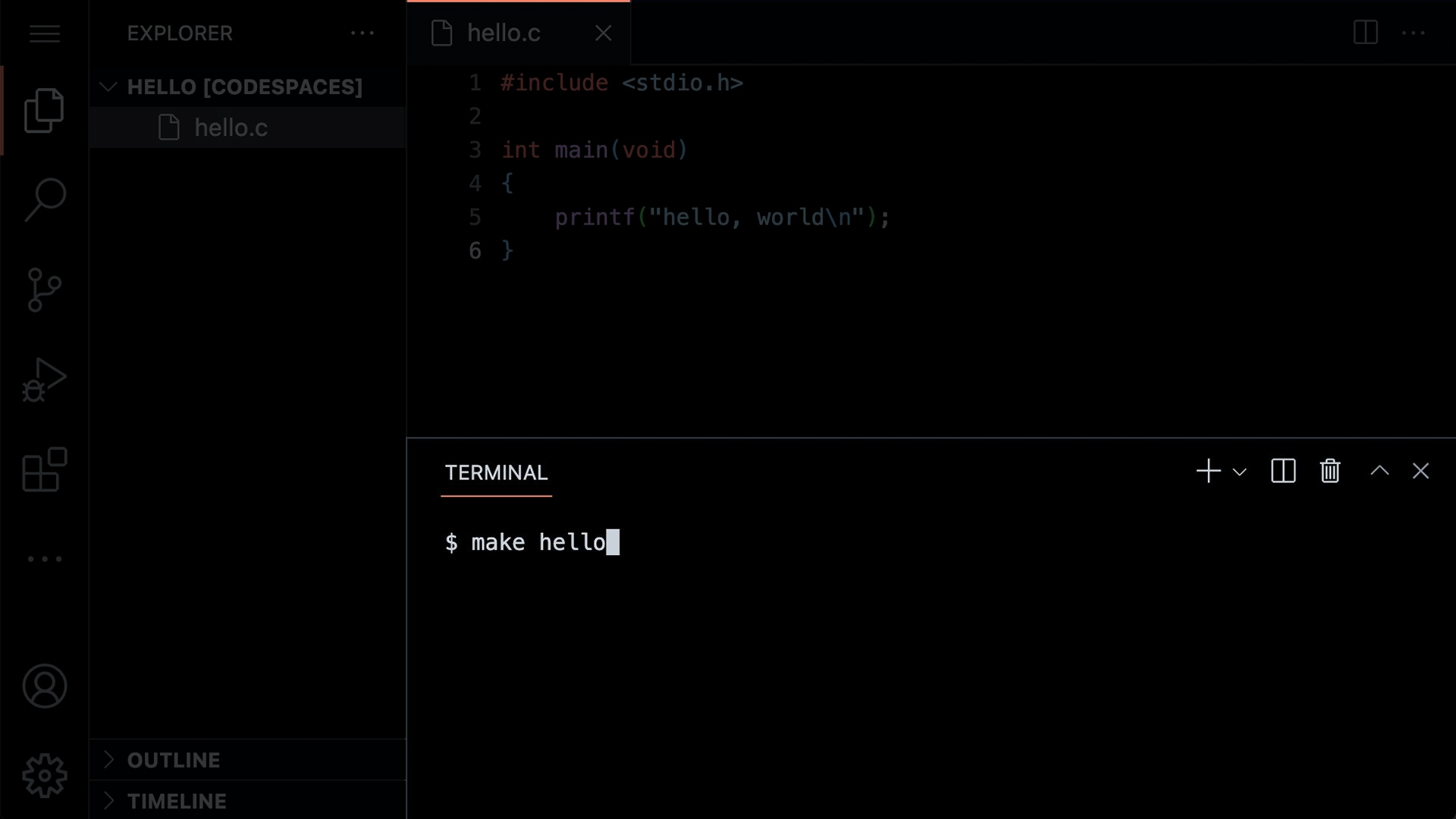

source code
compiler
machine code

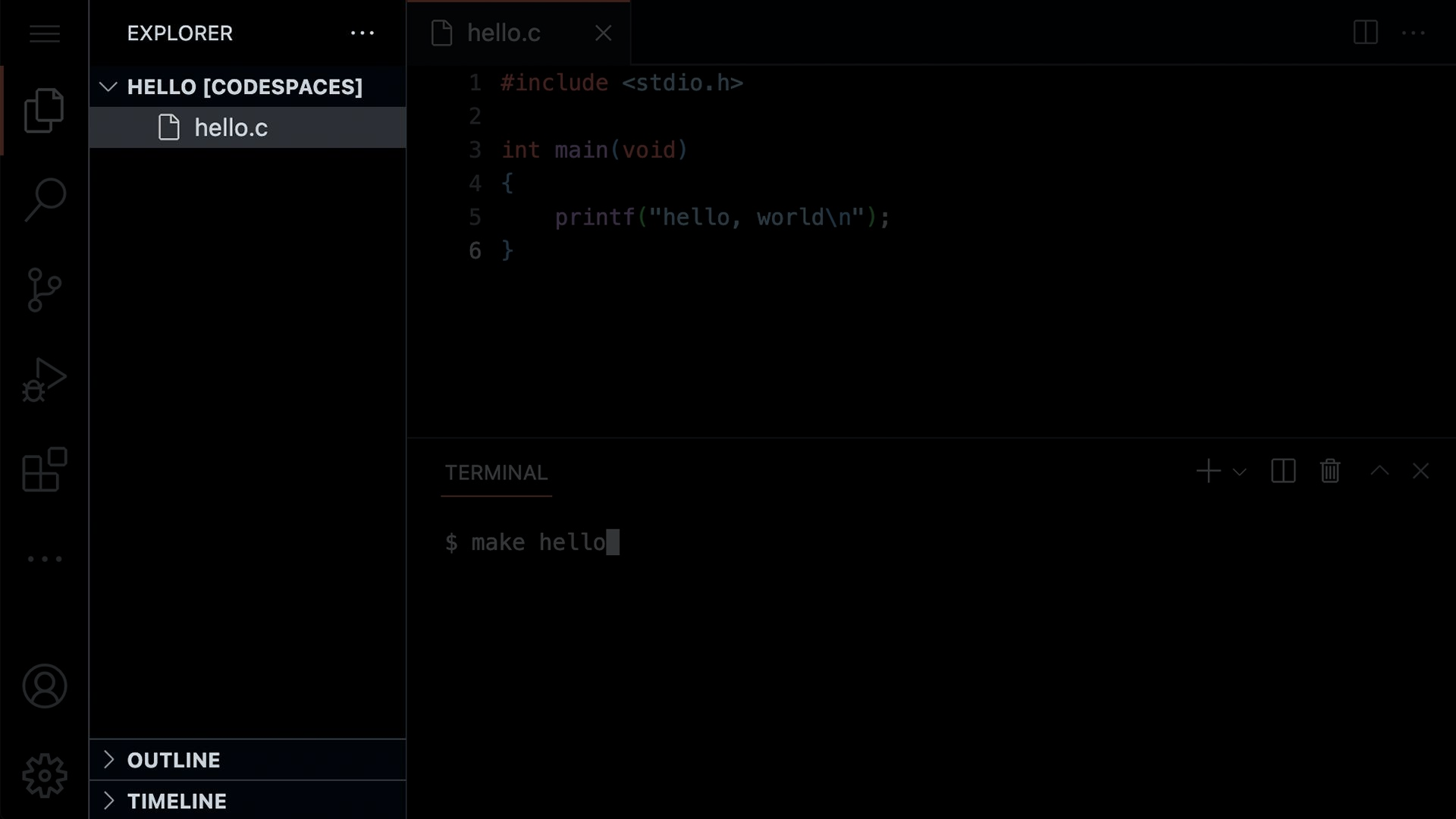
Visual Studio Code for CS50

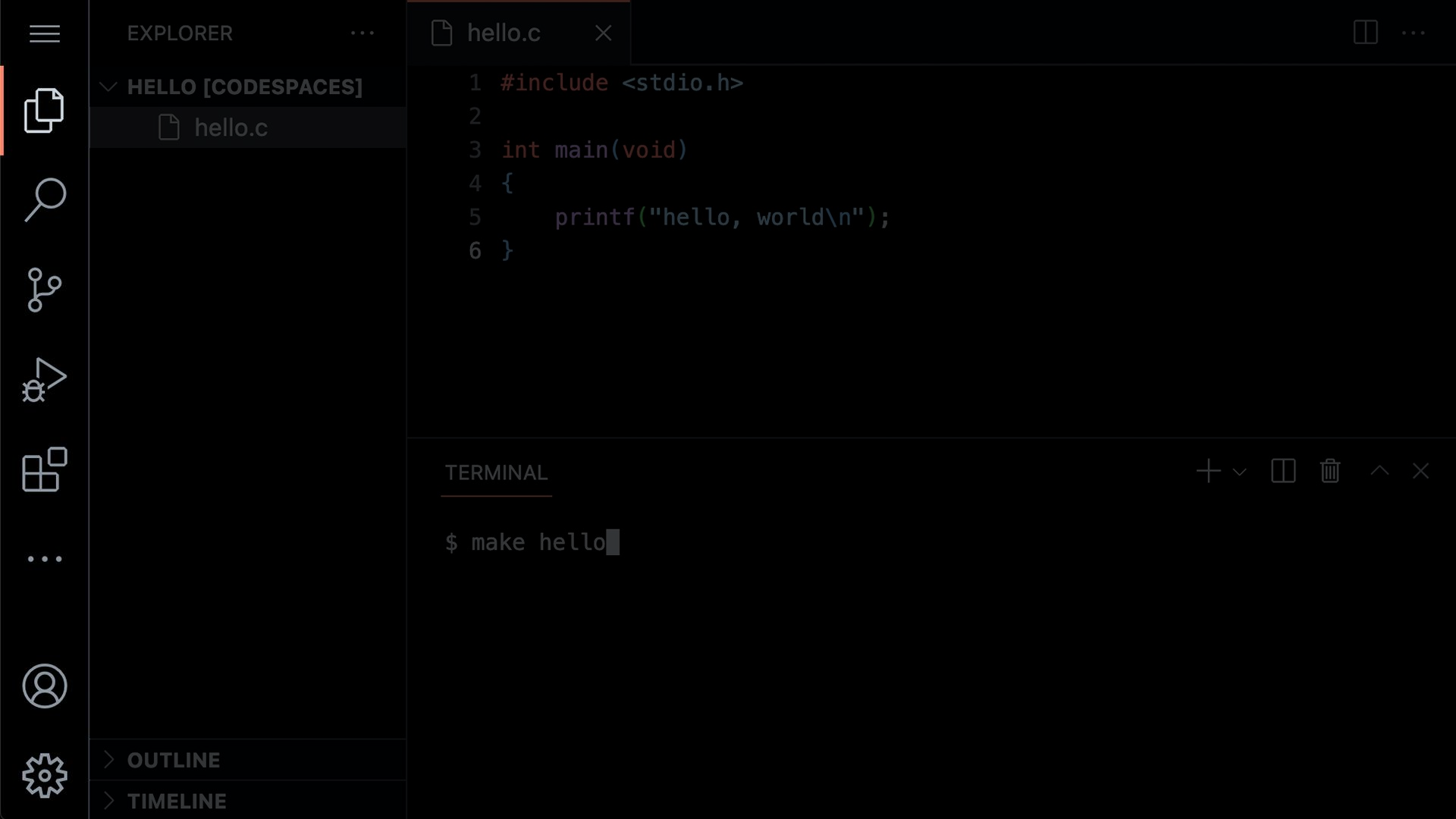
cs50.dev

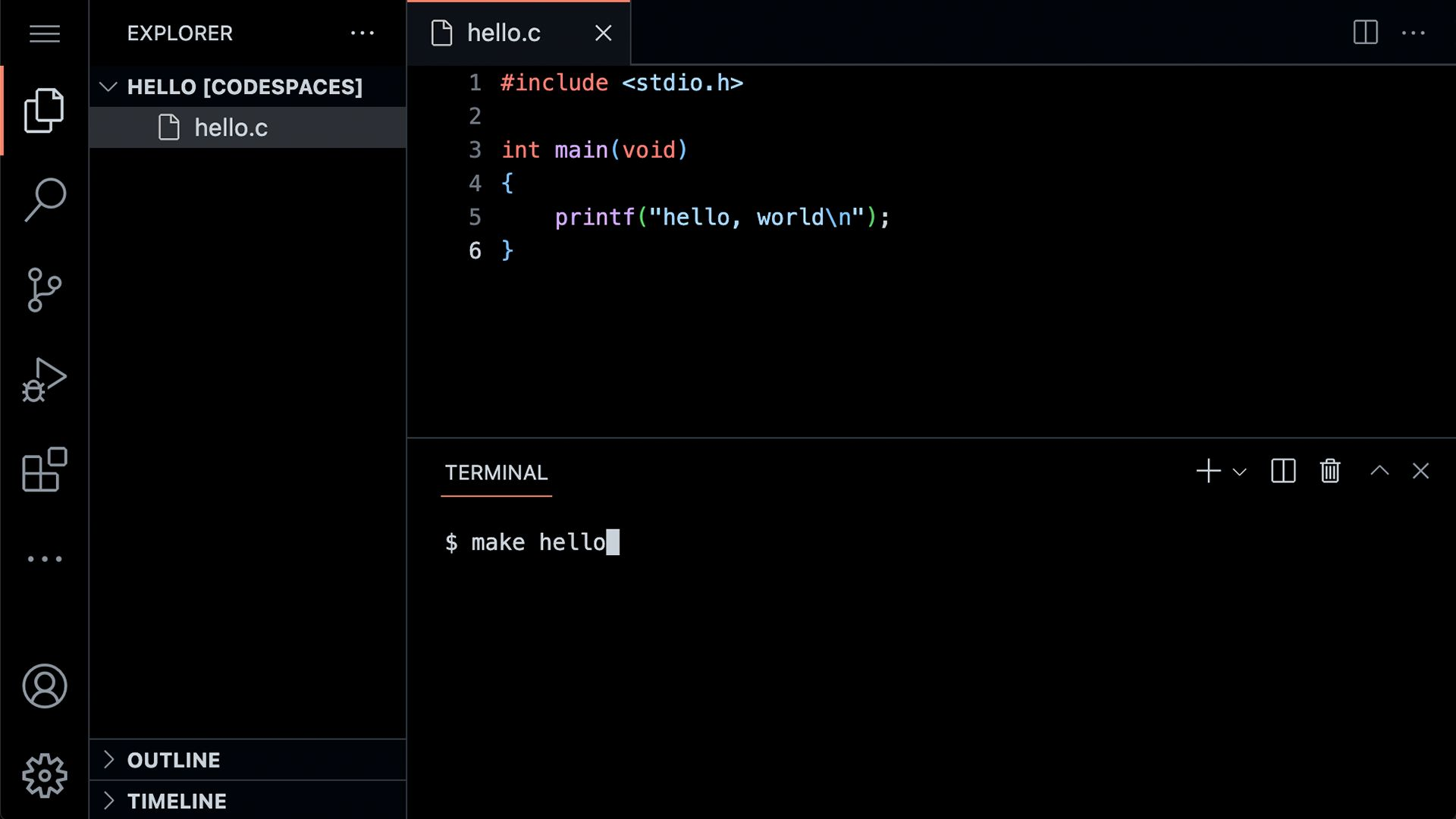

EXPLORER
HELLO [CODESPACES]
hello.c
hello.c
1 #include <stdio.h>
2
3 int main(void)
4 {
5 printf("hello, world\n");
6 }
TERMINAL
$ make hello
OUTLINE
TIMELINE


EXPLORER
HELLO [CODESPACES]
hello.c
hello.c
1 #include <stdio.h>
2
3 int main(void)
4 {
5 printf("hello, world\n");
6 }
TERMINAL
$ make hello
OUTLINE
TIMELINE


EXPLORER
HELLO [CODESPACES]
hello.c
hello.c
1 #include <stdio.h>
2
3 int main(void)
4 {
5 printf("hello, world\n");
6 }
TERMINAL
$ make hello
OUTLINE
TIMELINE

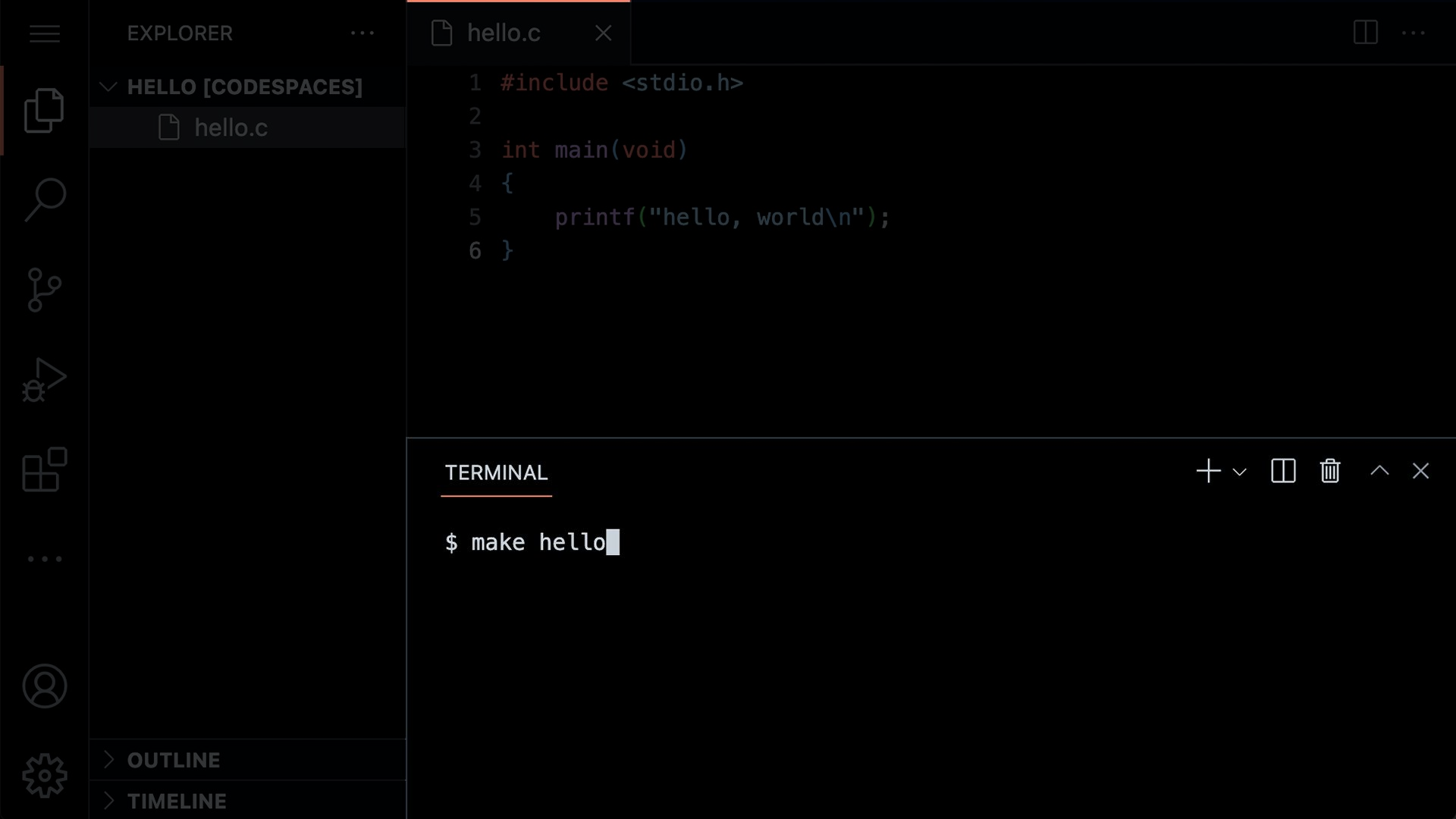

EXPLORER
HELLO [CODESPACES]
hello.c
hello.c
#include <stdio.h>
int main(void)
{
printf("hello, world\n");
}
TERMINAL
$ make hello
OUTLINE
TIMELINE


EXPLORER
HELLO [CODESPACES]
hello.c
hello.c
1 #include <stdio.h>
2
3 int main(void)
4 {
5 printf("hello, world\n");
6 }
TERMINAL
$ make hello
OUTLINE
TIMELINE


EXPLORER
HELLO [CODESPACES]
hello.c
hello.c
1 #include <stdio.h>
2
3 int main(void)
4 {
5 printf("hello, world\n");
6 }
TERMINAL
$ make hello
OUTLINE
TIMELINE

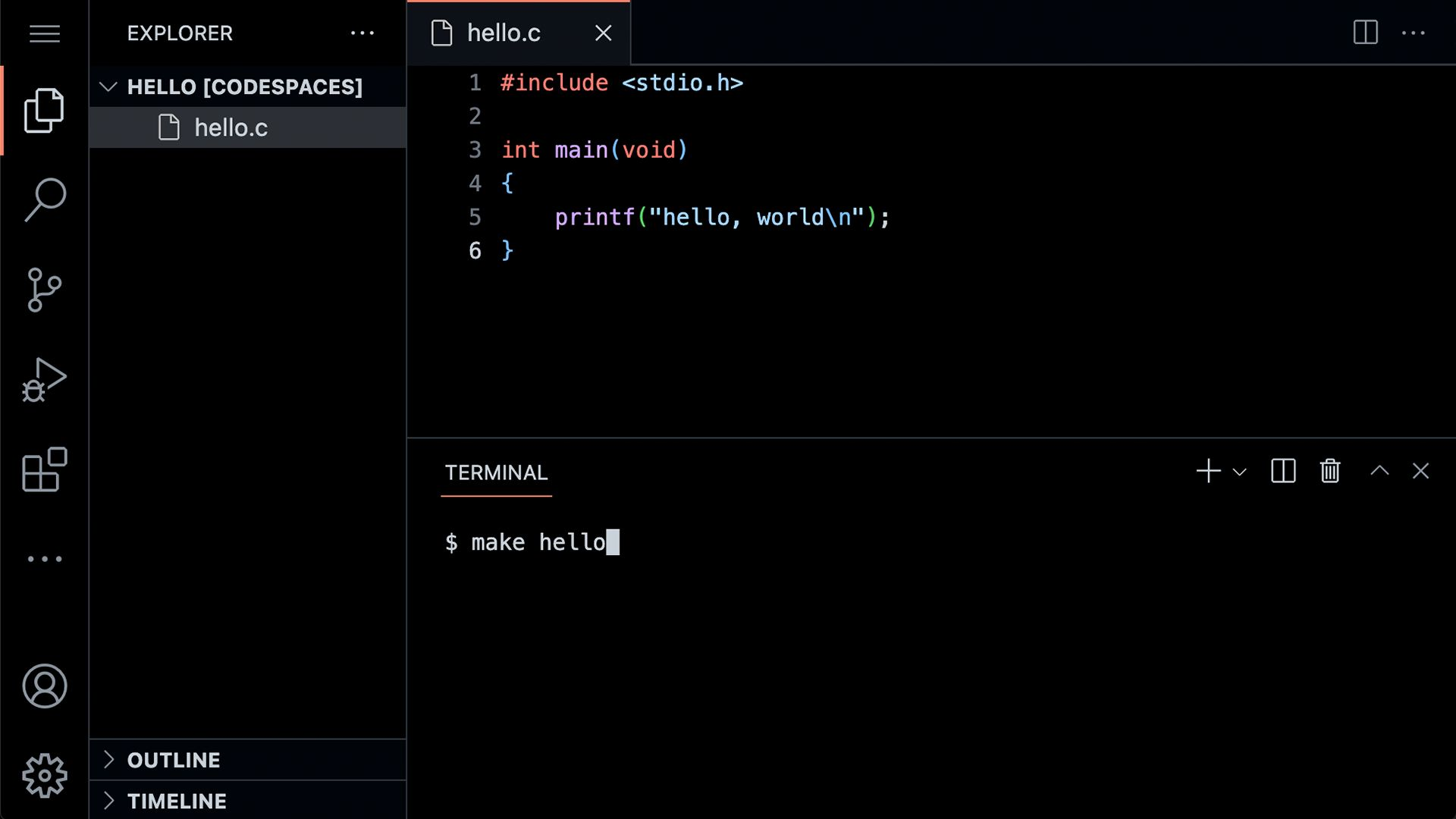
graphical user interface

GUI


EXPLORER
HELLO [CODESPACES]
hello.c
hello.c
1 #include <stdio.h>
2
3 int main(void)
4 {
5 printf("hello, world\n");
6 }
TERMINAL
$ make hello
OUTLINE
TIMELINE

command-line interface

CLI


EXPLORER
HELLO [CODESPACES]
hello.c
hello.c
1 #include <stdio.h>
2
3 int main(void)
4 {
5 printf("hello, world\n");
6 }
TERMINAL
$ make hello
OUTLINE
TIMELINE

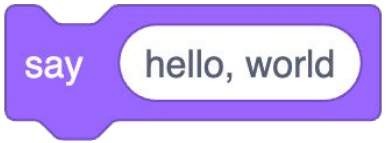
```c
#include <stdio.h>

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

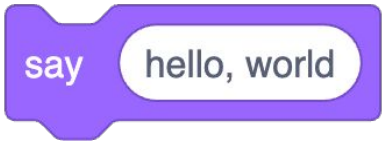
```
code hello.c

make hello

./hello
```


when clicked
say
hello, world

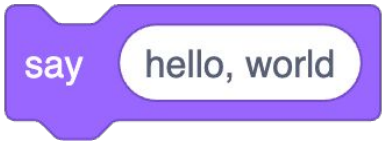
```c
#include <stdio.h>

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

```c
#include <stdio.h>

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

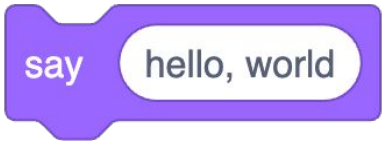

say
hello, world



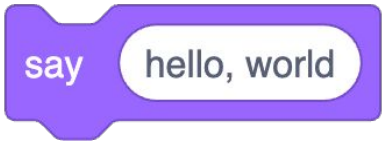

```
print (                    )
```



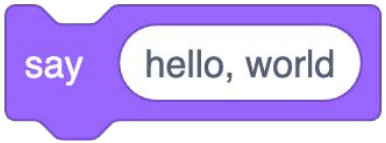

```
printf(                    )
```



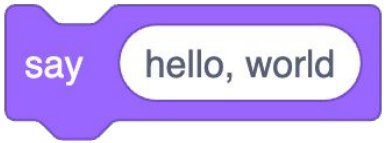

```
printf( hello, world   )
```




```
printf("hello, world  ")
```

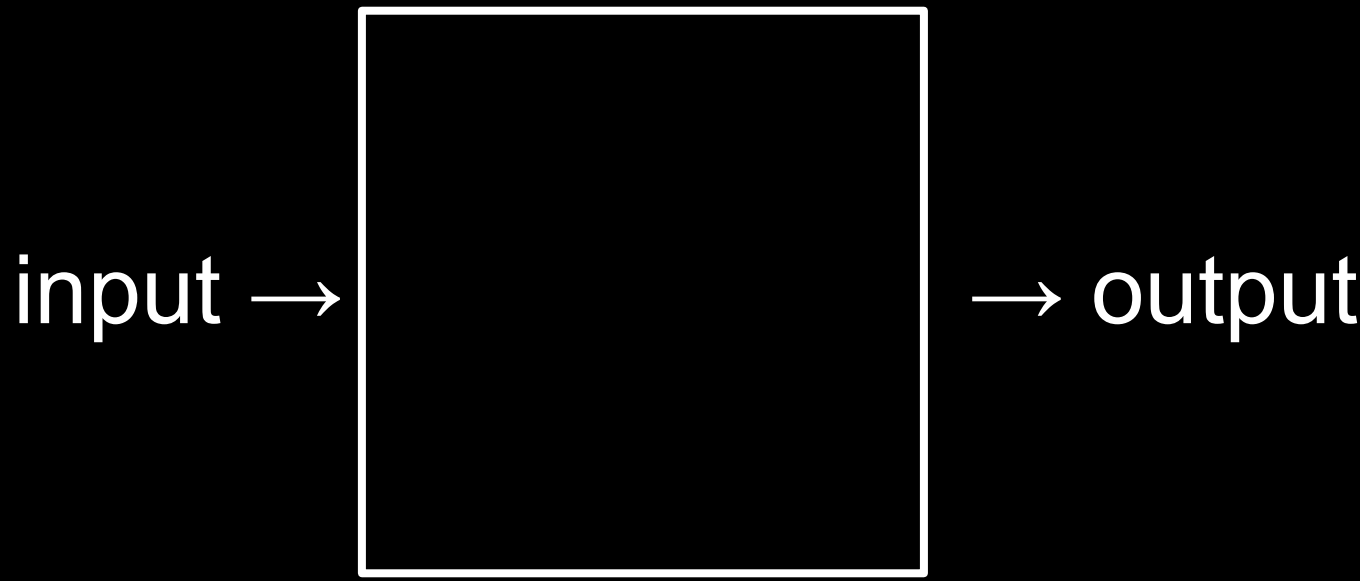


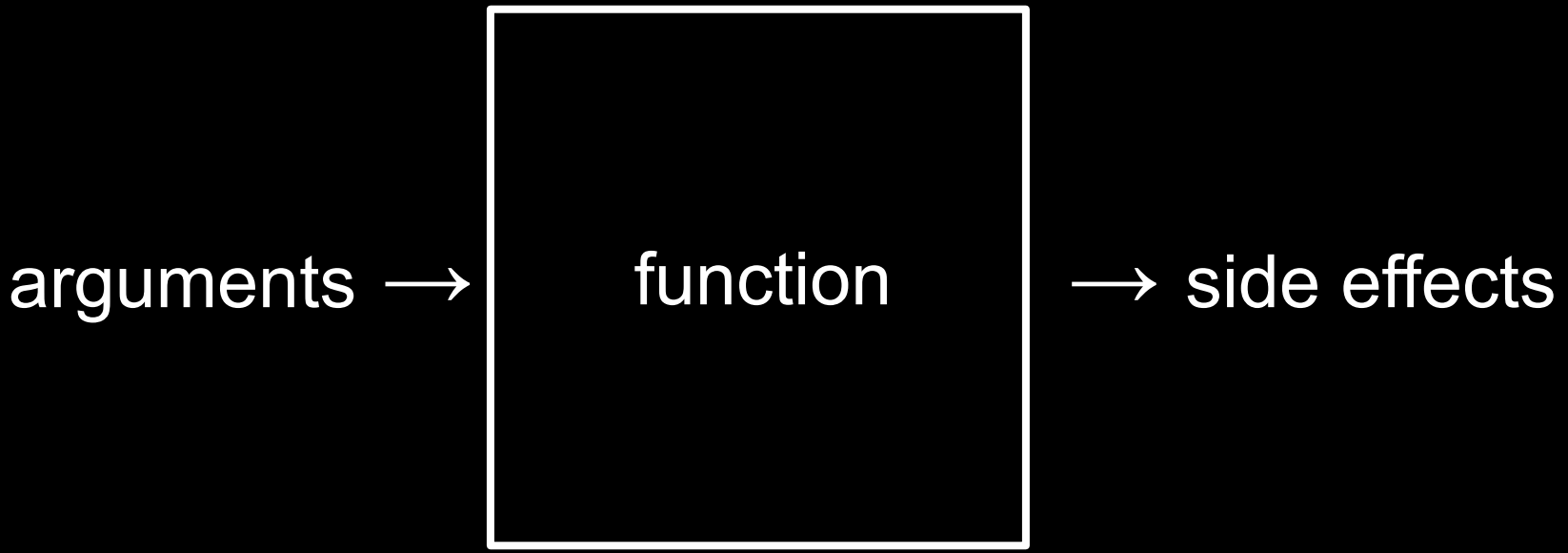

```
printf("hello, world\n")
```

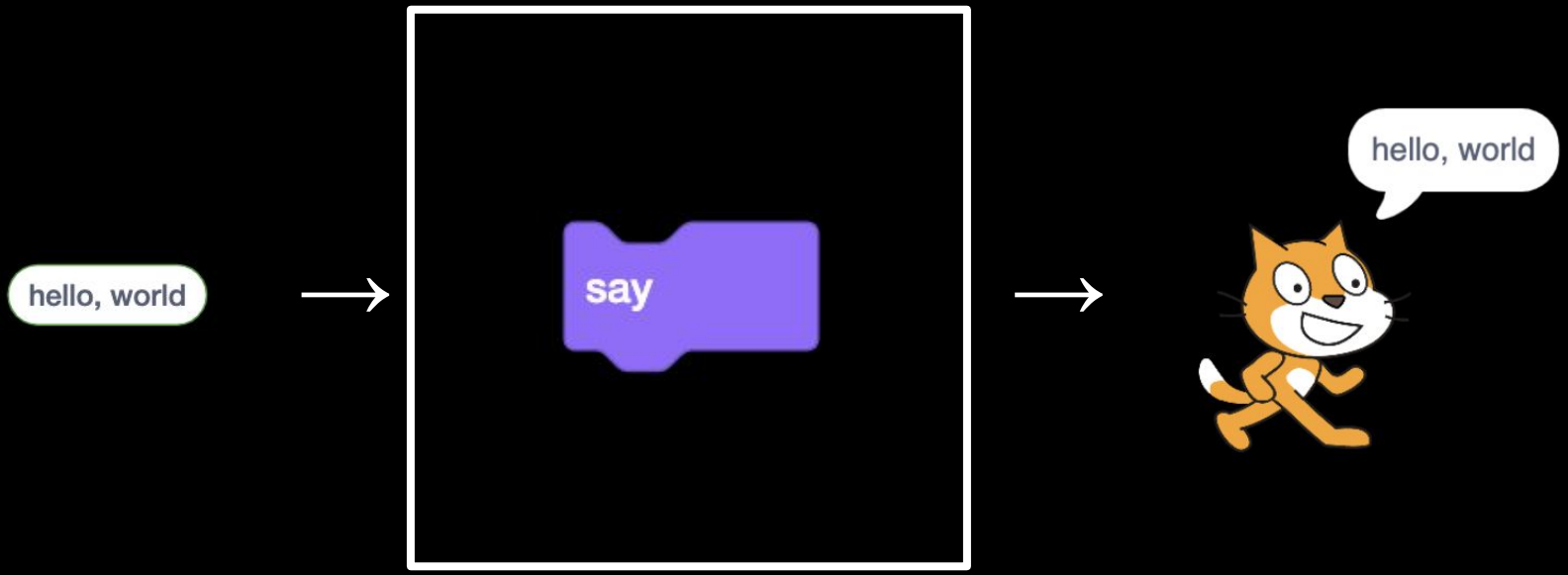



```
printf("hello, world\n");
```

$$f(x)$$


input →
→ output


arguments
function
side effects


hello, world
say
hello, world

escape sequences

\n

\r

\"

\'

\\

…

header files

libraries

stdio.h

Manual Pages

manual.cs50.io

manual.cs50.io/#stdio.h

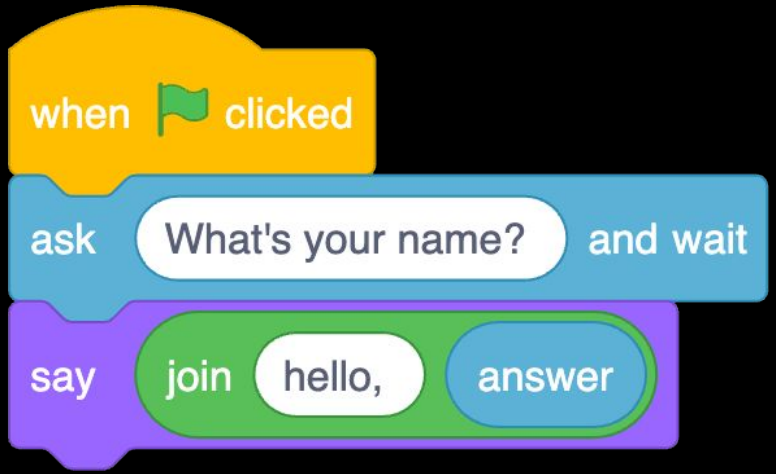
manual.cs50.io/3/printf

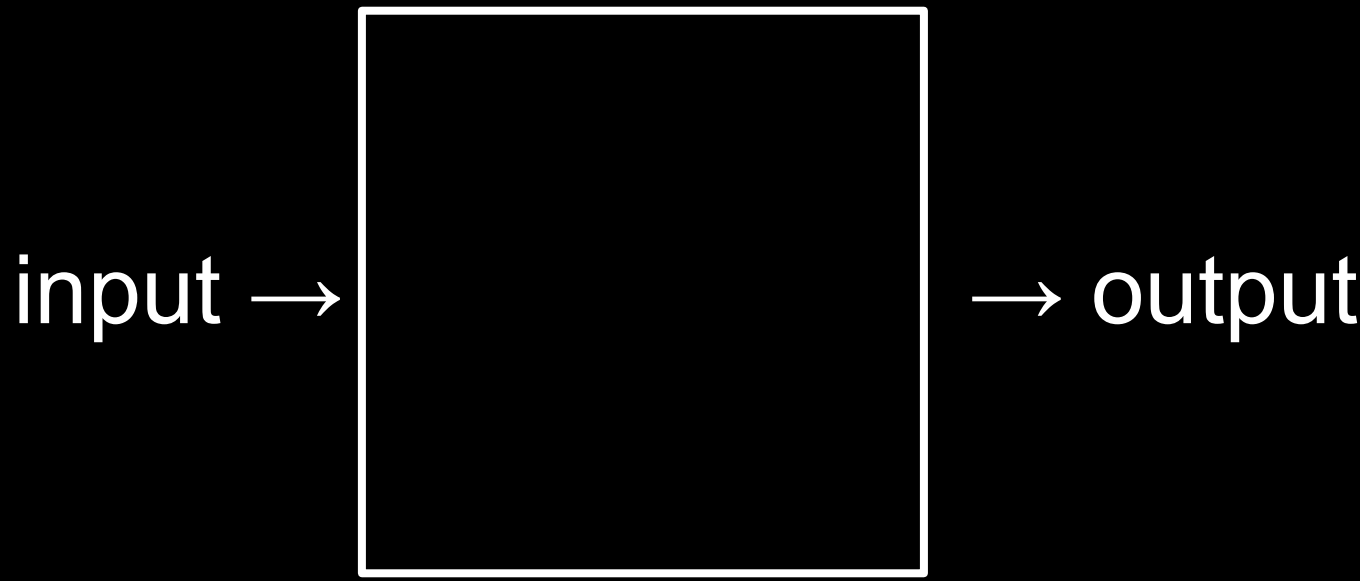
cs50.h

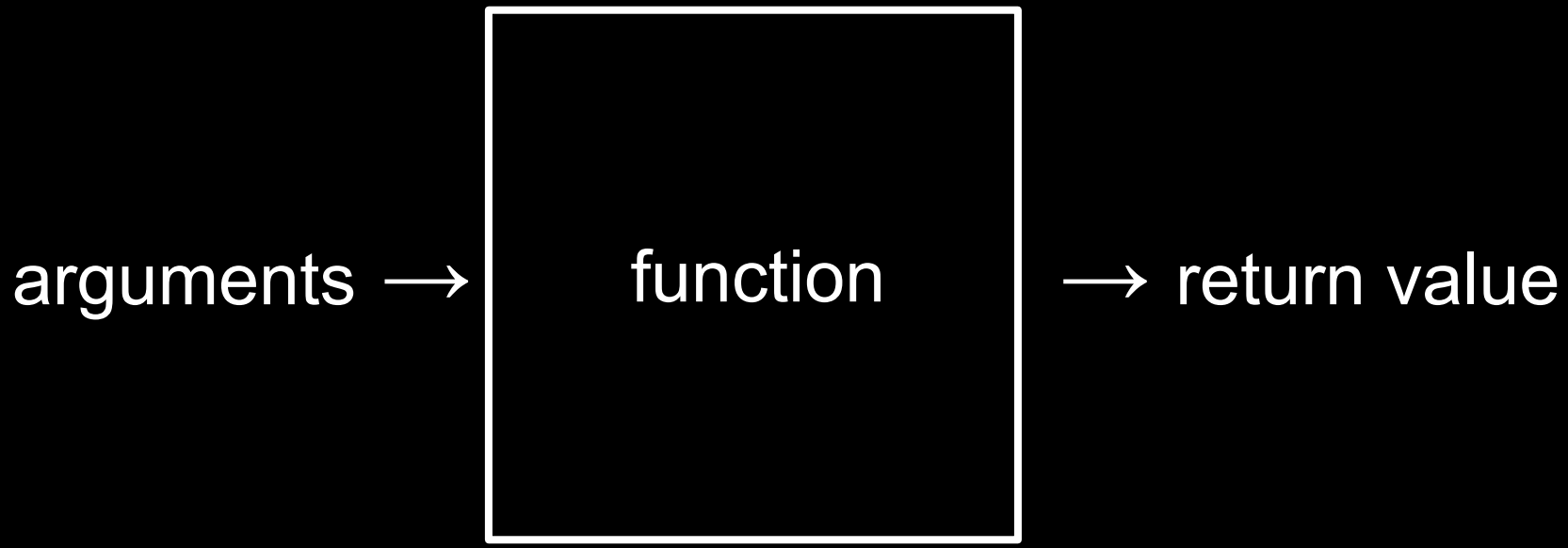
manual.cs50.io/#cs50.h

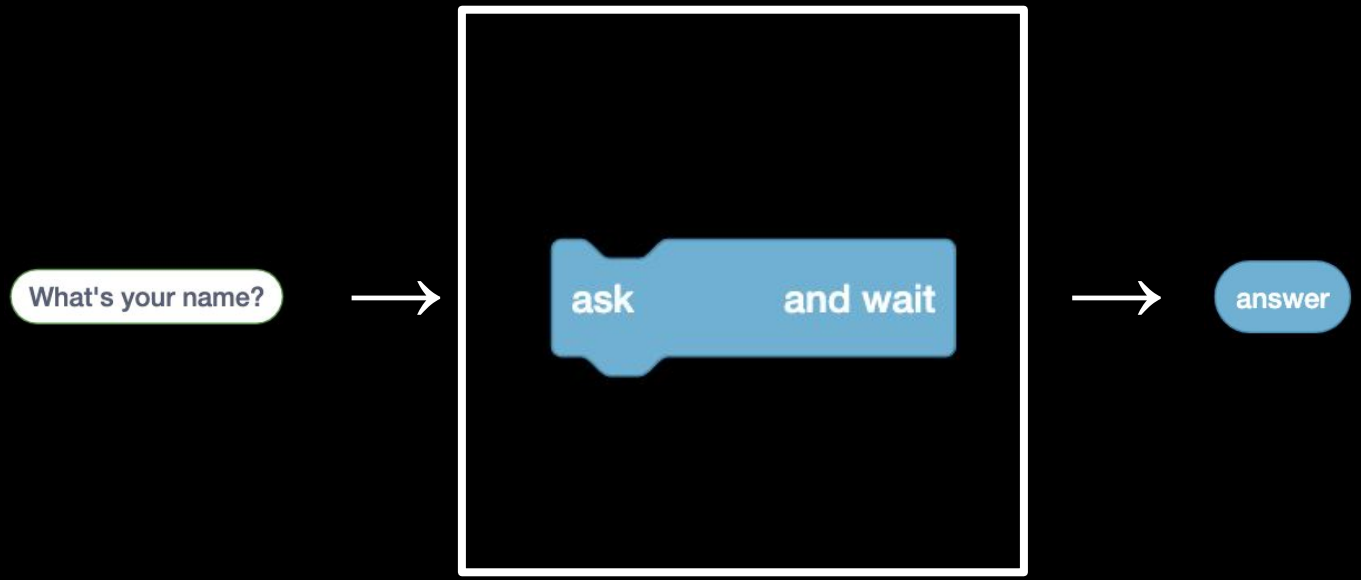
```
get_char

get_double

get_float

get_int

get_long

get_string

...
```

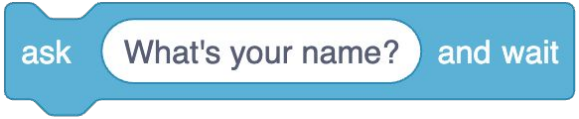
```
get_char

get_double

get_float

get_int

get_long

get_string

...
```


when clicked
ask What's your name? and wait
say join hello, answer

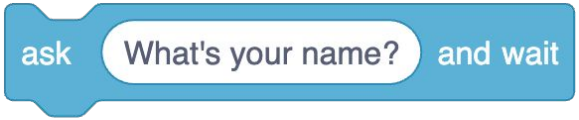

input →
→ output


arguments
function
return value

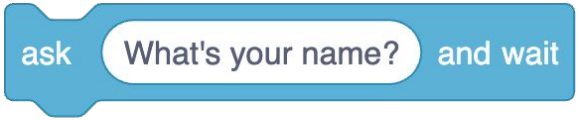

What's your name?
ask
and wait
answer


ask
What's your name?
and wait

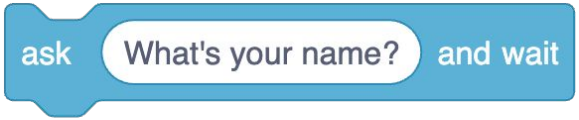

answer



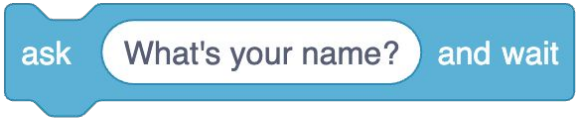

```
get_string(                    )
```



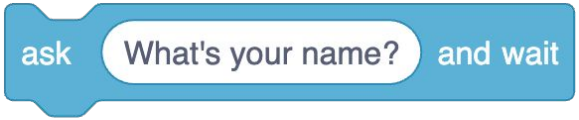

```
get_string( What's your name?  )
```



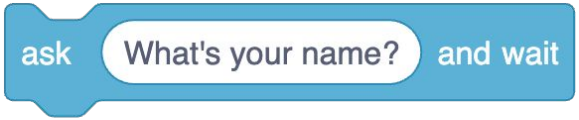

```
get_string("What's your name? ")
```




```
answer = get_string("What's your name? ")
```




```
string answer = get_string("What's your name? ")
```




```
string answer = get_string("What's your name? ");
```


say
join
hello,
answer


```
printf(                    );
```


```
printf( hello, %s           );
```


```
printf("hello, %s  "        );
```


```
printf("hello, %s\n"        );
```


```
printf("hello, %s\n", answer);
```

```
cd

cp

ls

mkdir

mv

rm

rmdir

...
```

```
cd
cp
ls
mkdir
mv
rm
rmdir
...
```

types

```
bool

char

double

float

int

long

string

...
```

```
bool

char

double

float

int

long

string

...
```

```
bool

char

double

float

int

long

string

...
```

```
get_char

get_double

get_float

get_int

get_long

get_string

...
```

```
get_char

get_double

get_float

get_int

get_long

get_string

...
```

format codes

%c

%f

%i

%li

%s

%c

%f

%i

%li

%s

%c

%f

%i

%li

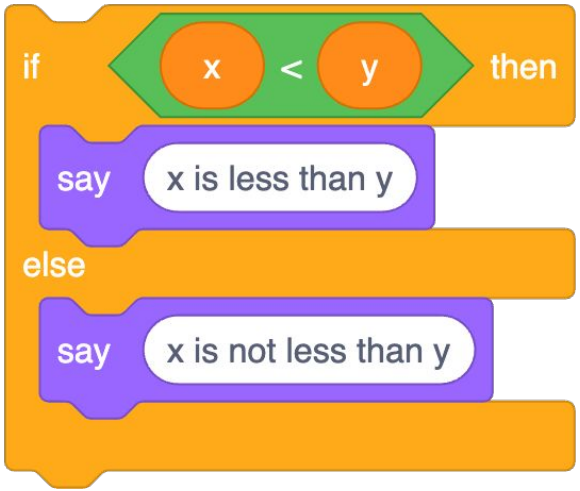
%s

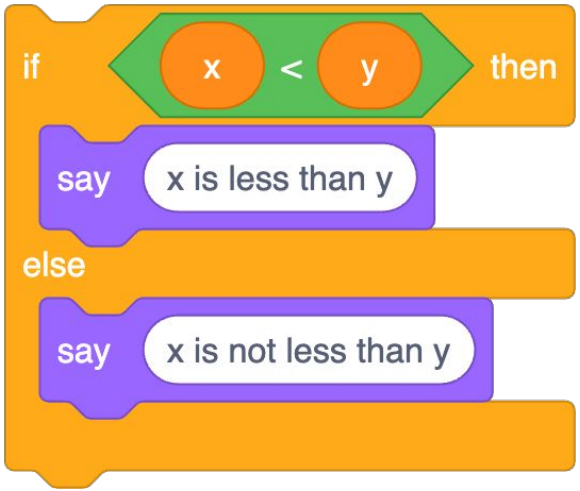
conditionals


if
x
<
y
then
say
x is less than y

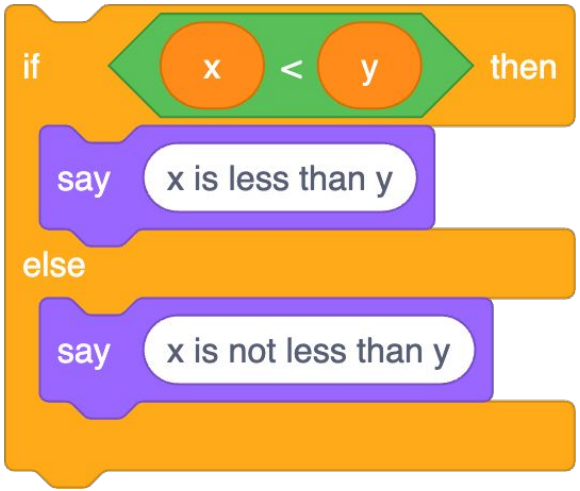

```
if (x < y)
{

}
```


```
if (x < y)
{
    printf("x is less than y\n");
}
```

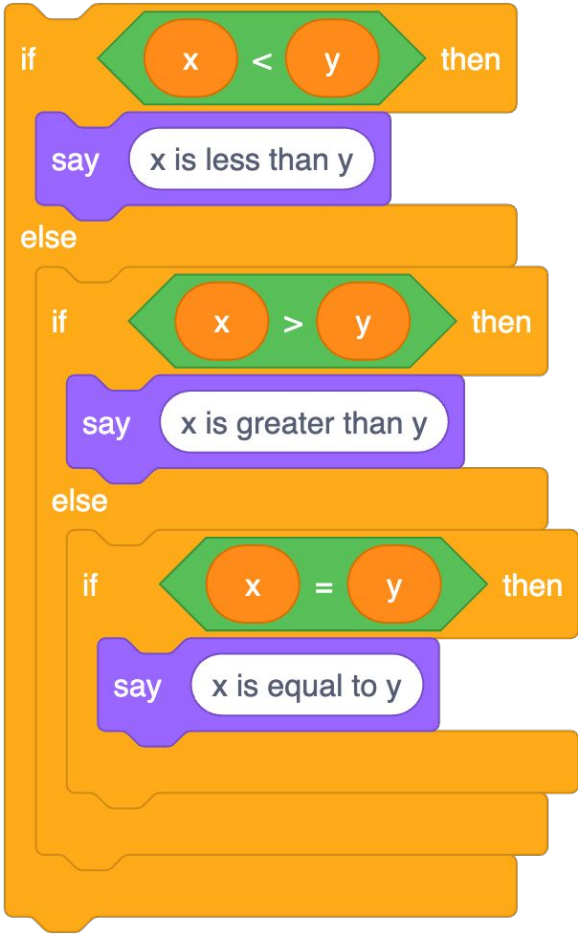

if
x
<
y
then
say
x is less than y
else
say
x is not less than y

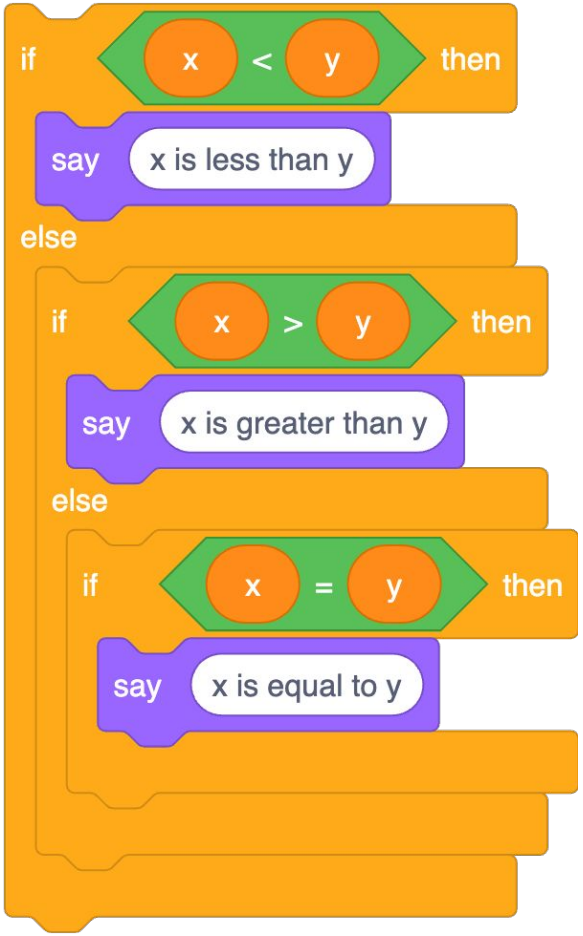



```
if (x < y)
{

}
else
{

}
```

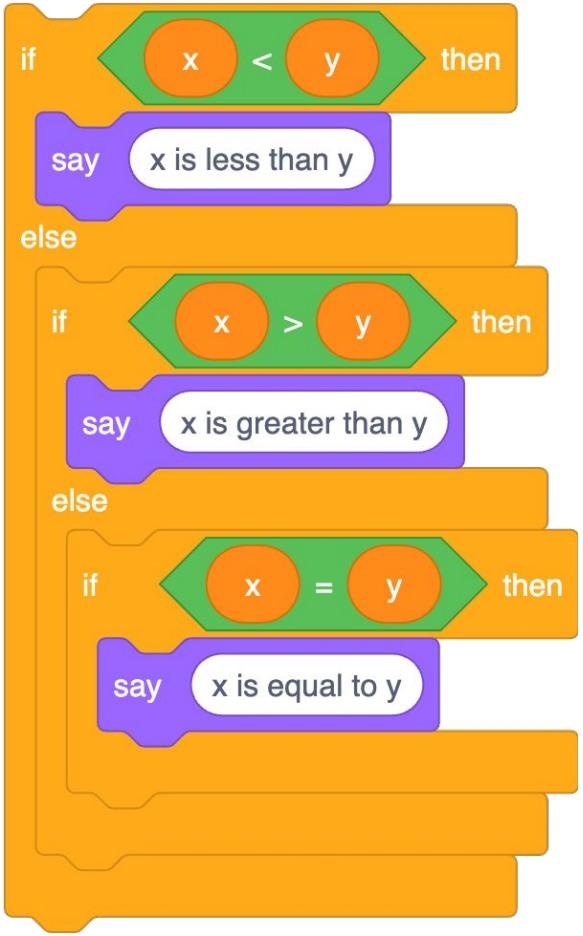



```
if (x < y)
{
    printf("x is less than y\n");
}
else
{
    printf("x is not less than y\n");
}
```

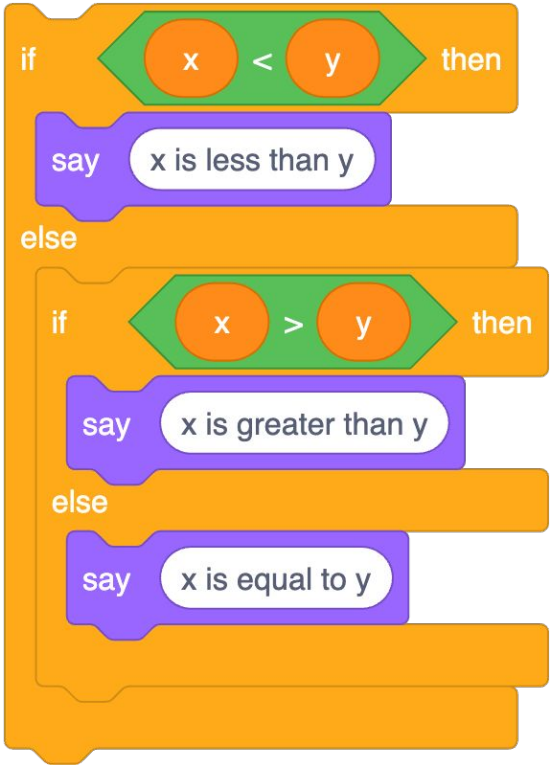

if x < y then
say x is less than y
else
if x > y then
say x is greater than y
else
if x = y then
say x is equal to y




```
if (x < y)
{

}
else if (x > y)
{

}
else if (x == y)
{

}
```




```
if (x < y)
{
    printf("x is less than y\n");
}
else if (x > y)
{
    printf("x is greater than y\n");
}
else if (x == y)
{
    printf("x is equal to y\n");
}
```




```
if (x < y)
{
    printf("x is less than y\n");
}
else if (x > y)
{
    printf("x is greater than y\n");
}
else
{
    printf("x is equal to y\n");
}
```

operators

=

<

<=

>

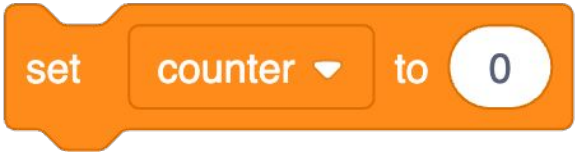
>=

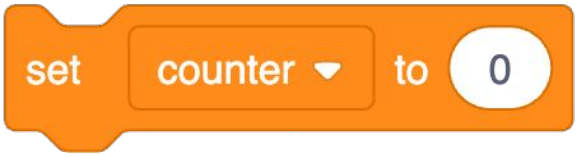
==

!=

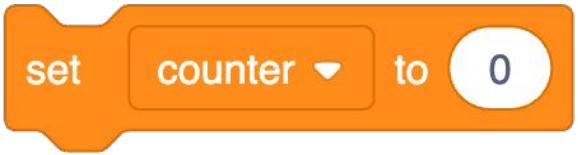
...

variables

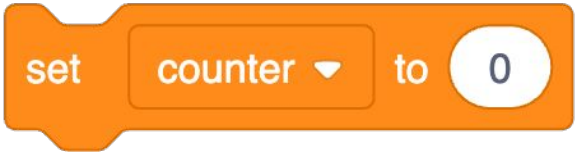

set
counter
to
0



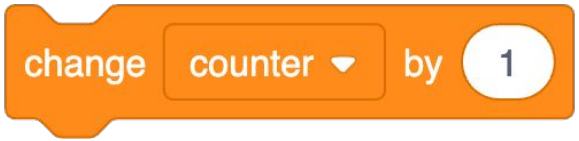

```
counter = 0
```

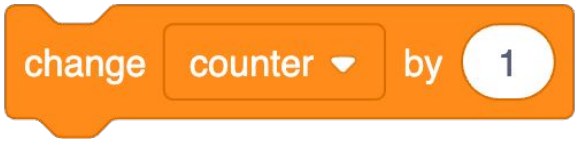



```
int counter = 0
```

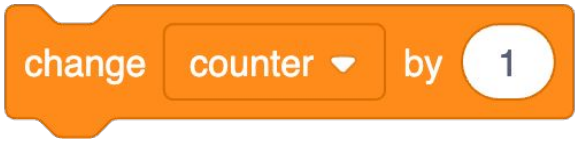



```
int counter = 0;
```

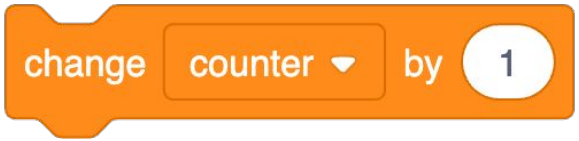

change
counter
by
1



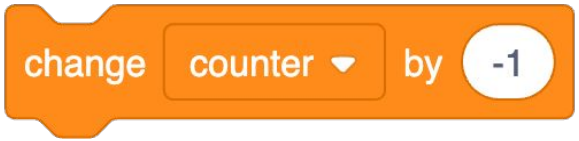

```
counter = counter + 1;
```



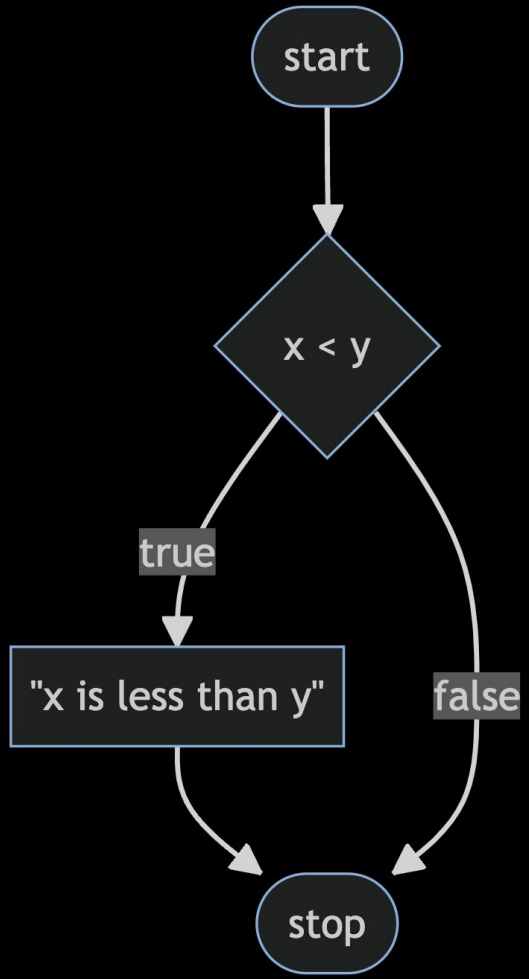

```
counter += 1;
```

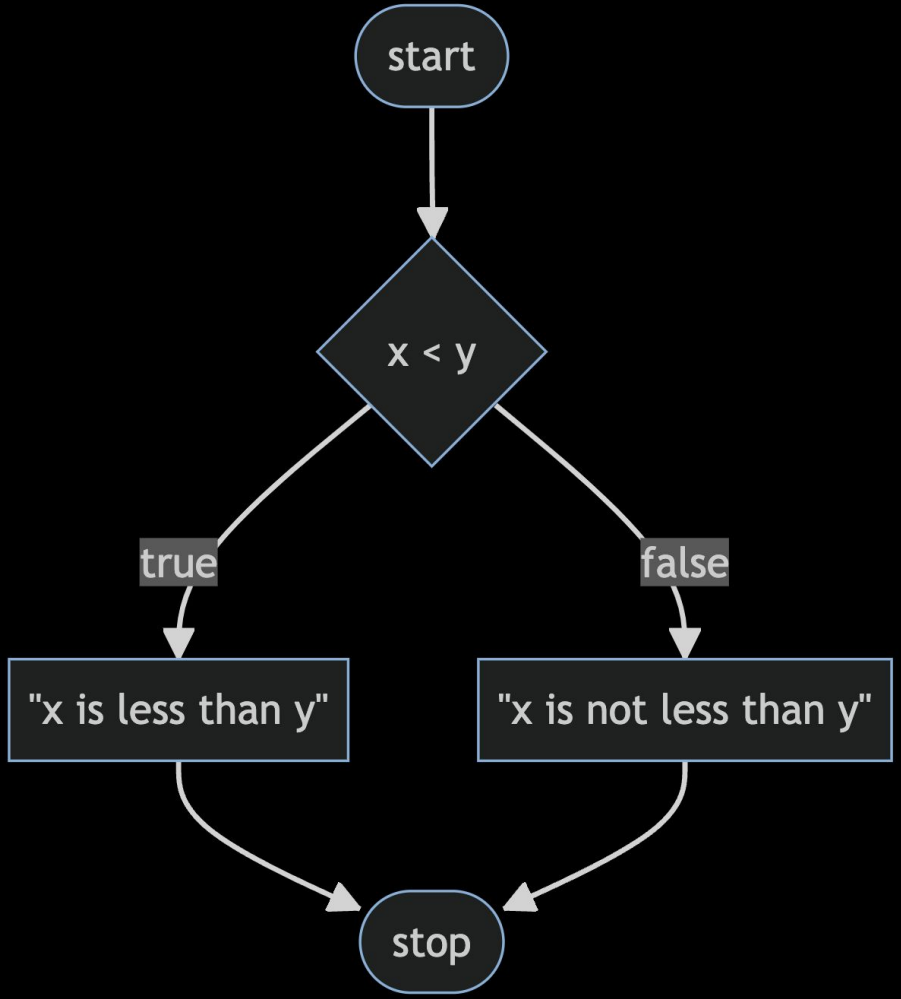


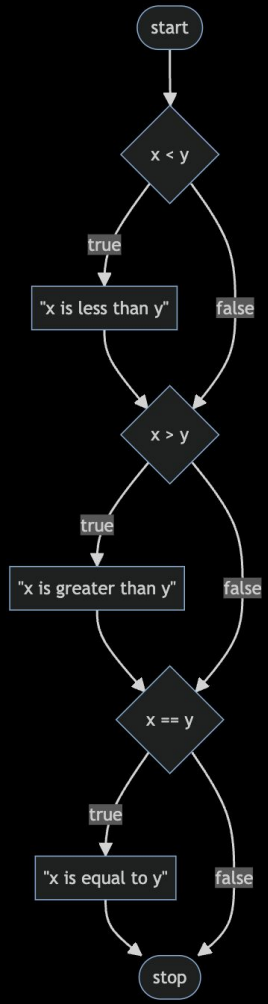

```
counter++;
```

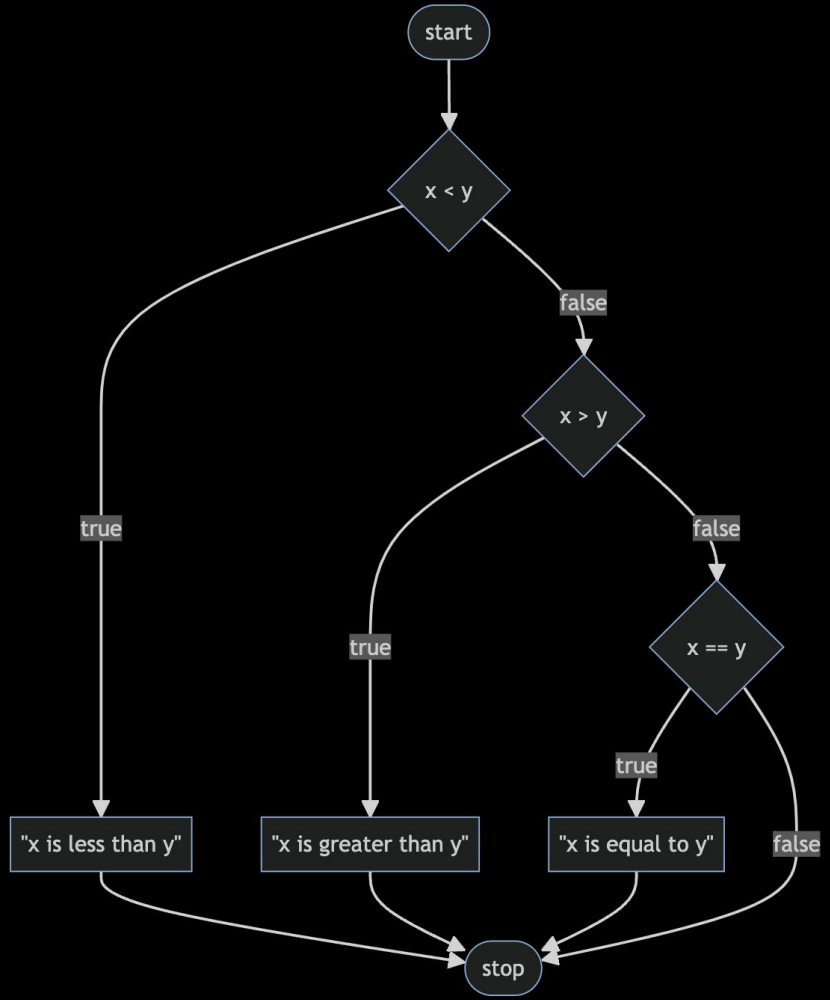


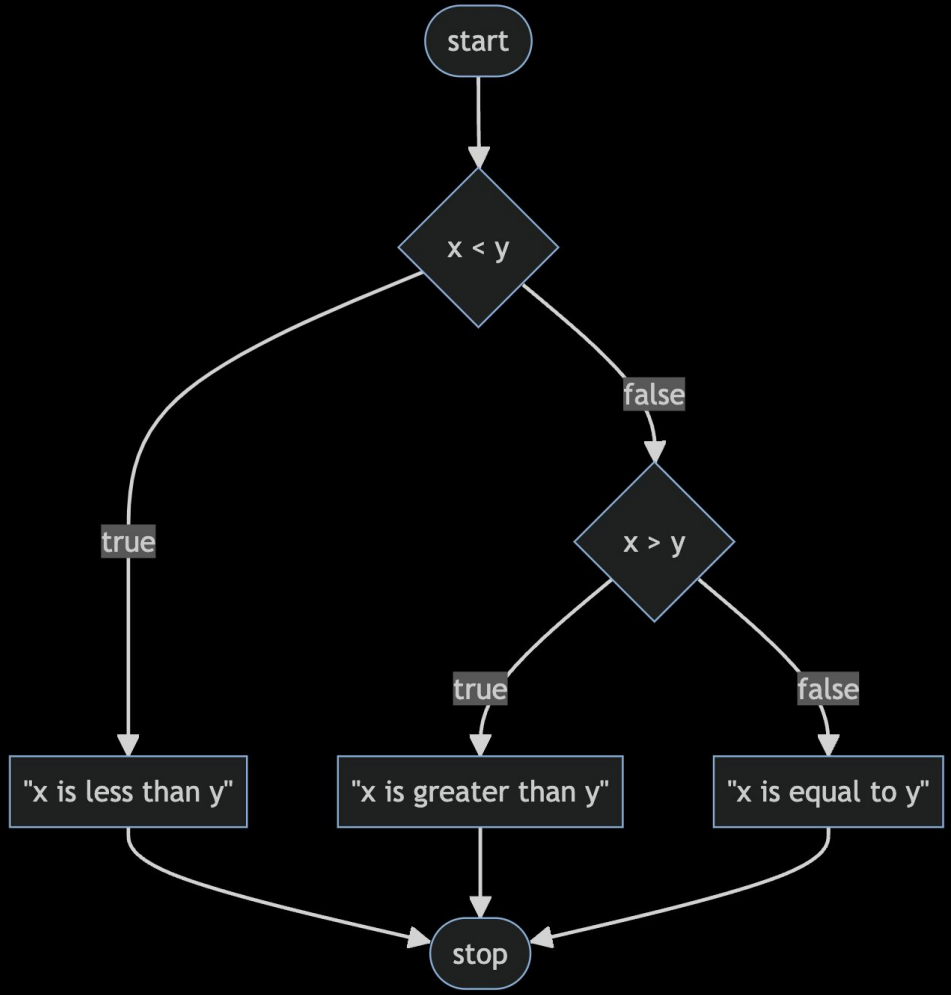

```
counter--;
```


start
x < y
true
"x is less than y"
false
stop


start
x < y
true
false
"x is less than y"
"x is not less than y"
stop


start
x < y
true
"x is less than y"
false
x > y
true
"x is greater than y"
false
x == y
true
"x is equal to y"
false
stop


start
x < y
true
false
x > y
true
false
x == y
true
false
"x is less than y"
"x is greater than y"
"x is equal to y"
stop


start
x < y
true
false
x > y
true
false
"x is less than y"
"x is greater than y"
"x is equal to y"
stop

```
bool

char

double

float

int

long

string

...
```

```
bool

char

double

float

int

long

string

...
```

```
get_char

get_double

get_float

get_int

get_long

get_string

...
```

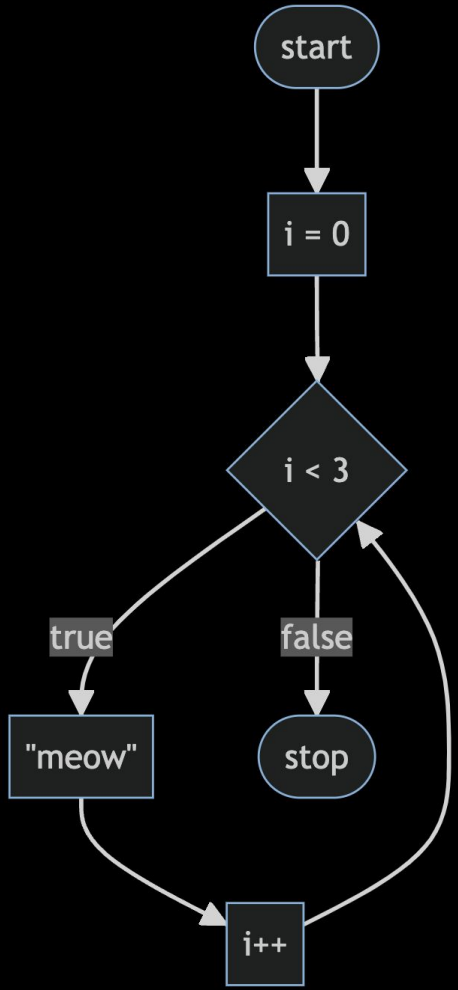
```
get_char

get_double

get_float

get_int

get_long

get_string

...
```

loops


repeat
3
say
meow

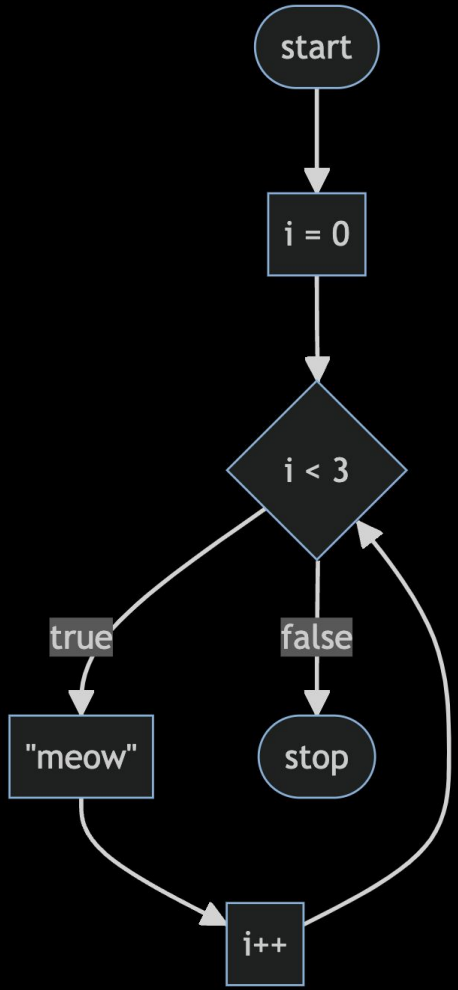

```
int i = 3;
while (i > 0)
{

    i--;
}
```


```
int i = 3;
while (i > 0)
{
    printf("meow\n");
    i--;
}
```


start
i = 0
i < 3
true
false
"meow"
stop
i++


```
int i = 0;
while (i < 3)
{
    printf("meow\n");
    i++;
}
```


```
for (int i = 0; i < 3; i++)
{
    printf("meow\n");
}
```


start
i = 0
i < 3
true
false
"meow"
stop
i++

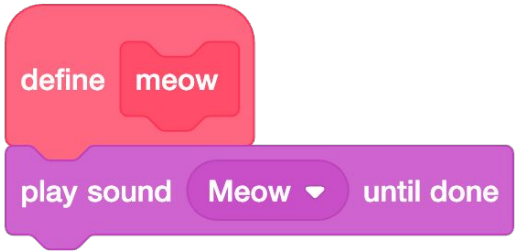

forever
say
meow

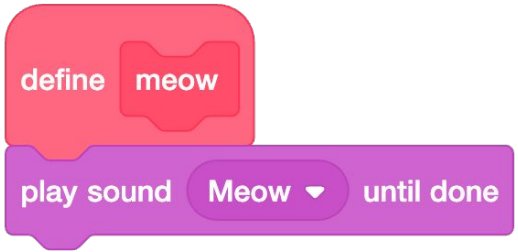

```
while (    )
{

}
```


```
while (true)
{

}
```

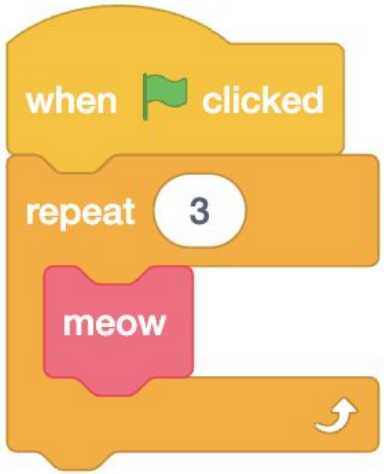

```
while (true)
{
    printf("meow\n");
}
```

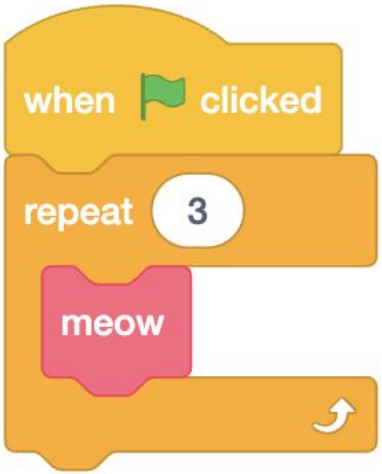
control-c


define meow
play sound Meow until done

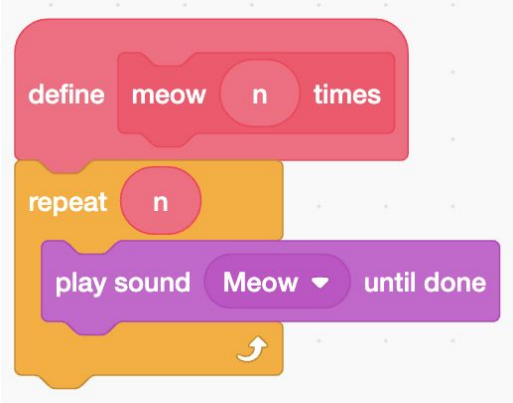


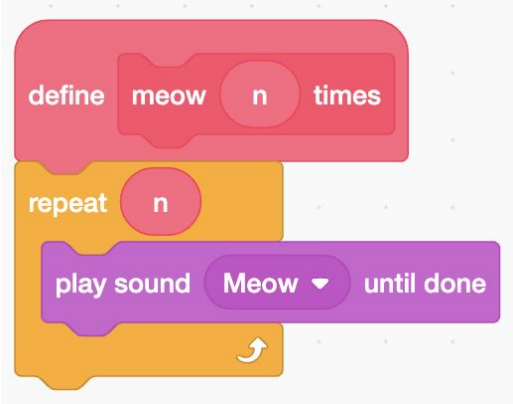

```
void meow(void)
{
    printf("meow\n");
}
```


when clicked
repeat 3
meow

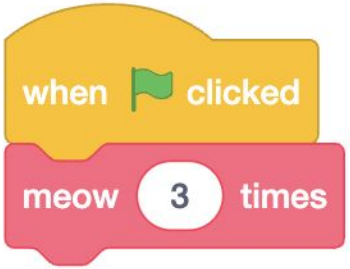


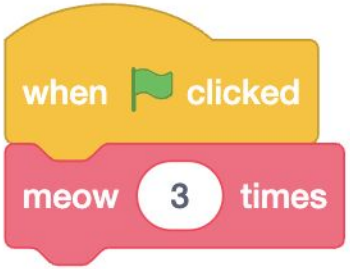

```c
int main(void)
{
    for (int i = 0; i < 3; i++)
    {
        meow();
    }
}
```


define meow n times
repeat n
play sound Meow until done




```
void meow(int n)
{
    for (int i = 0; i < n; i++)
    {
        printf("meow\n");
    }
}
```

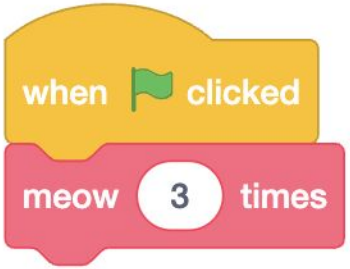

when clicked
meow 3 times




```
int main(void)
{
    meow(3);
}
```

scope

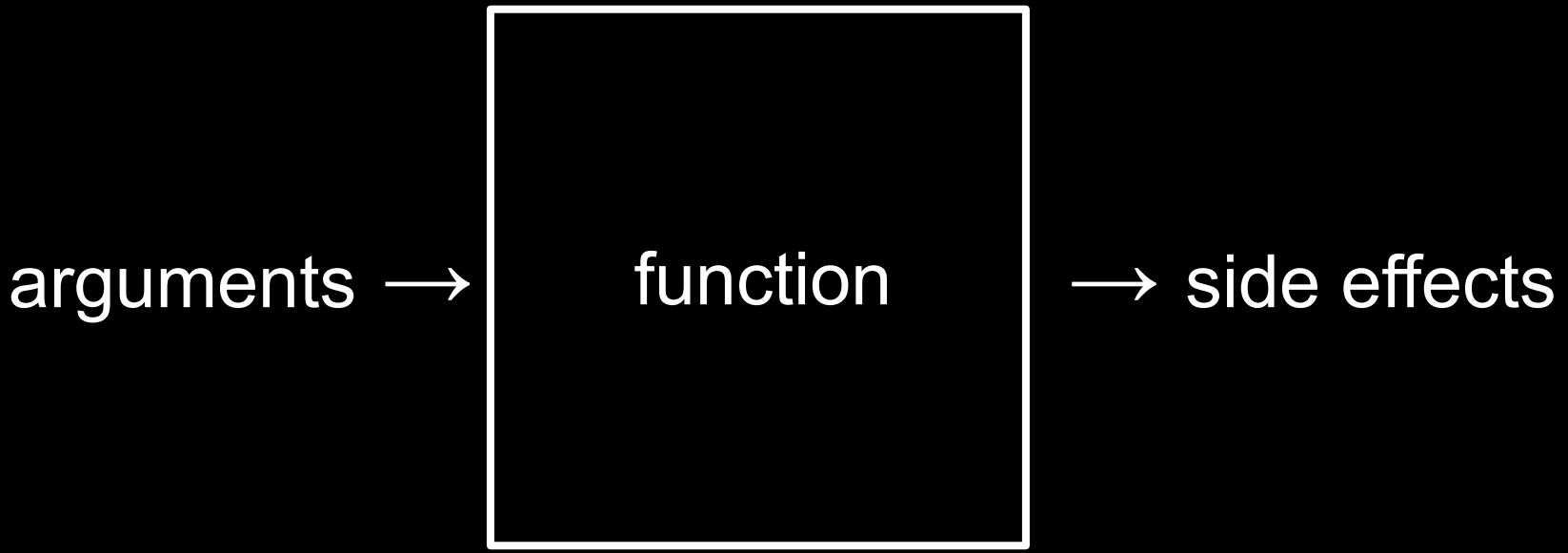
If you'd like a stress ball, what do today's binary bulbs spell?

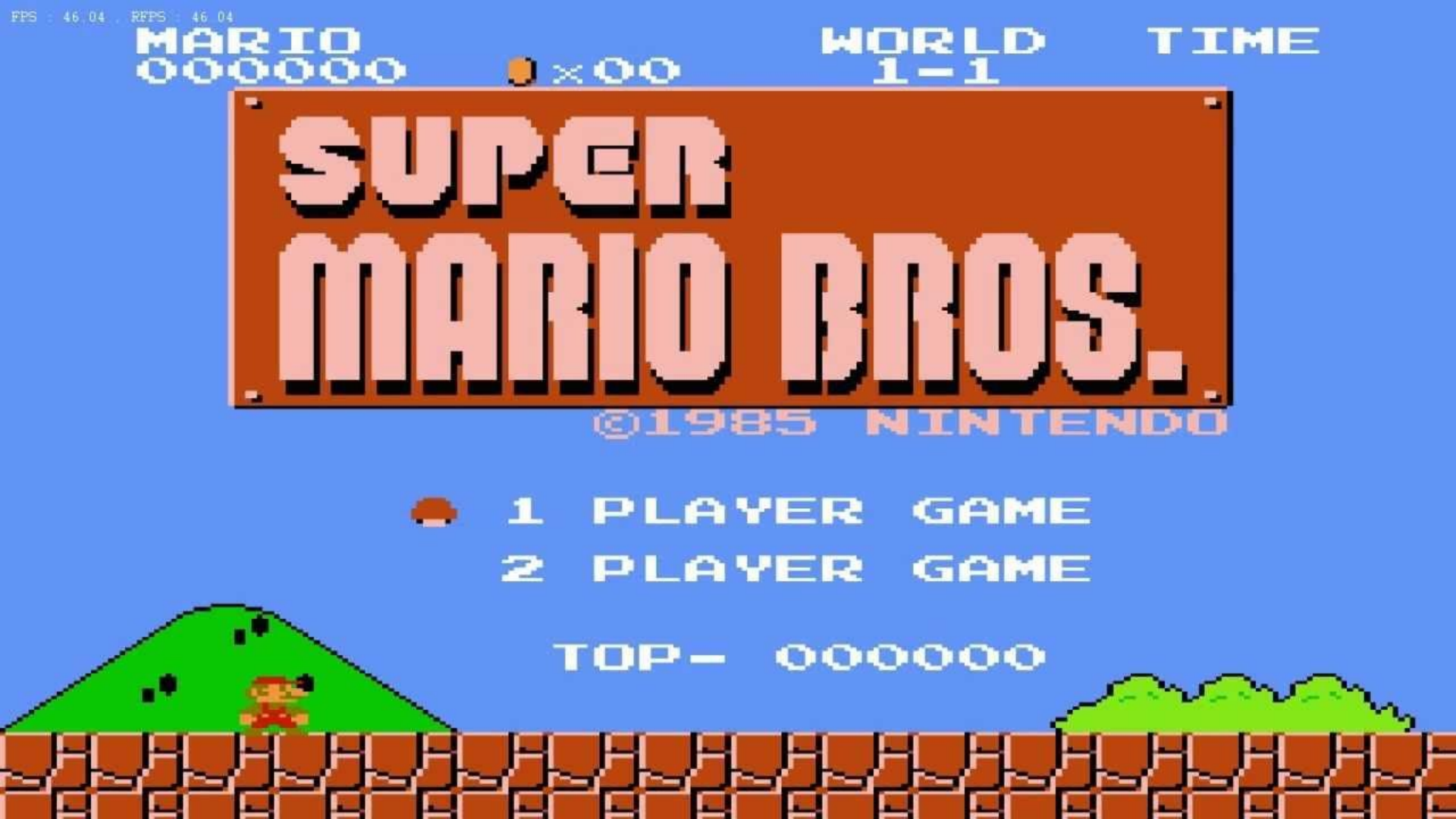


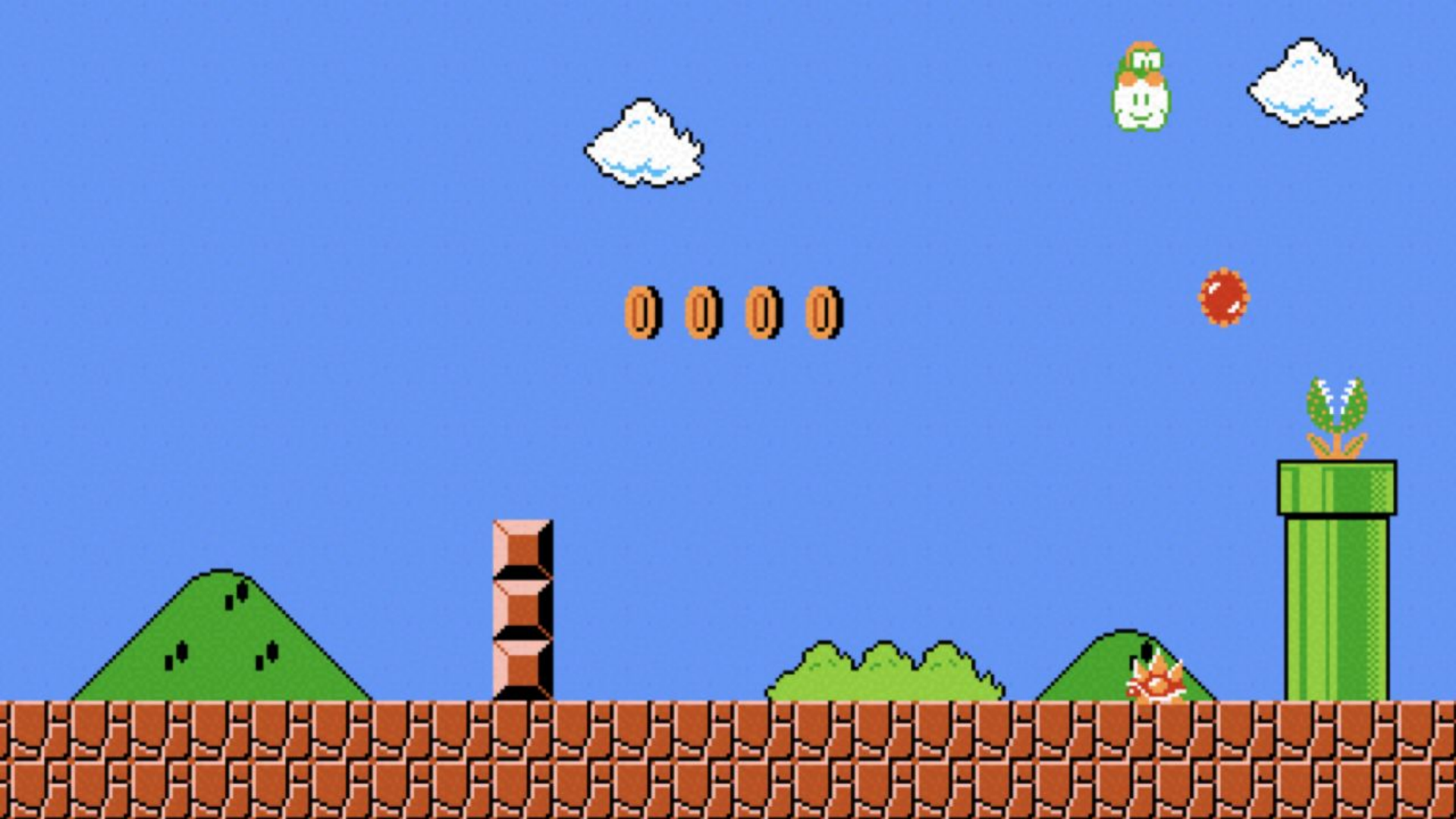

```
int main(void)
{
    meow(3);
}
```

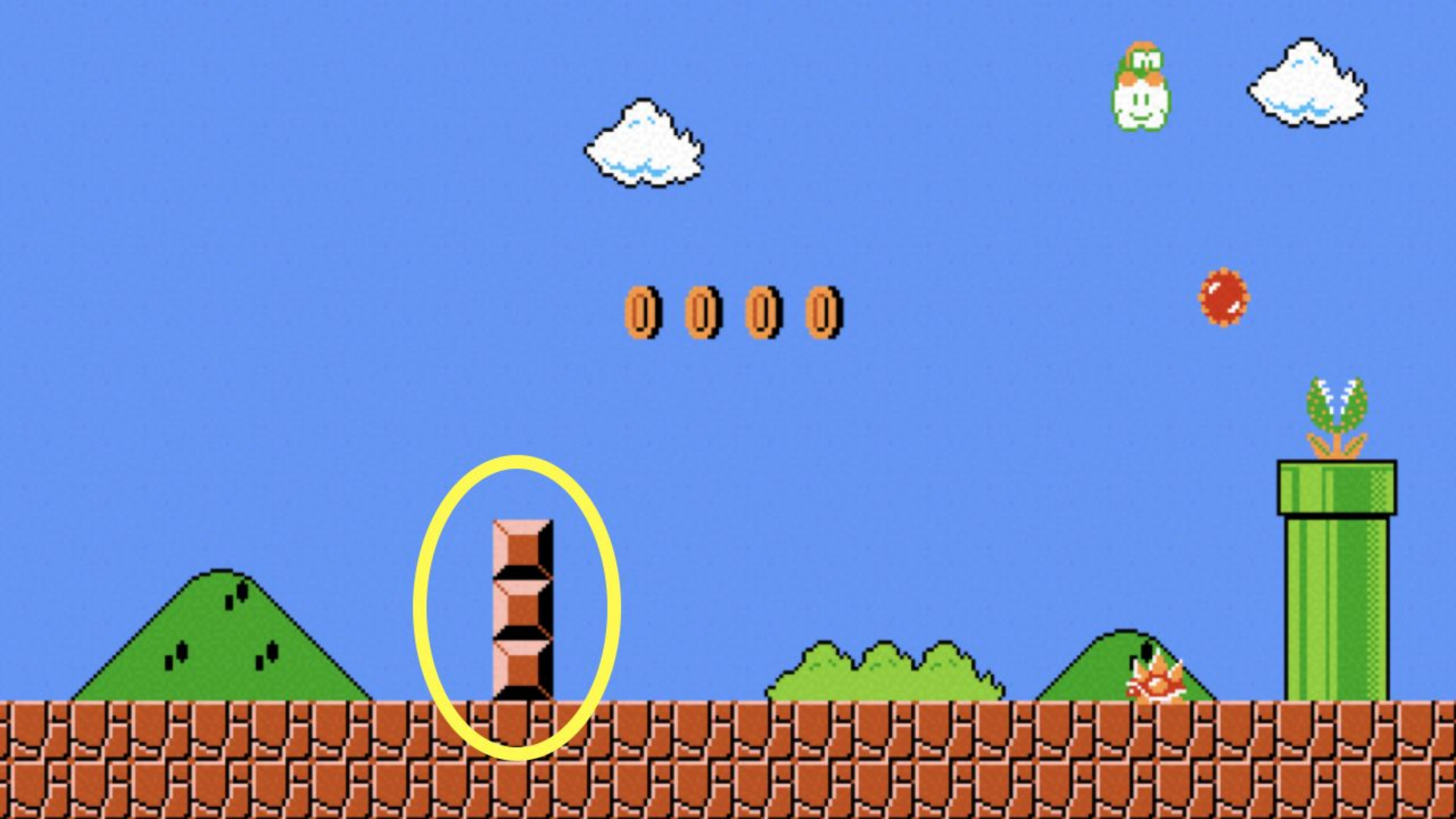
correctness, design, style

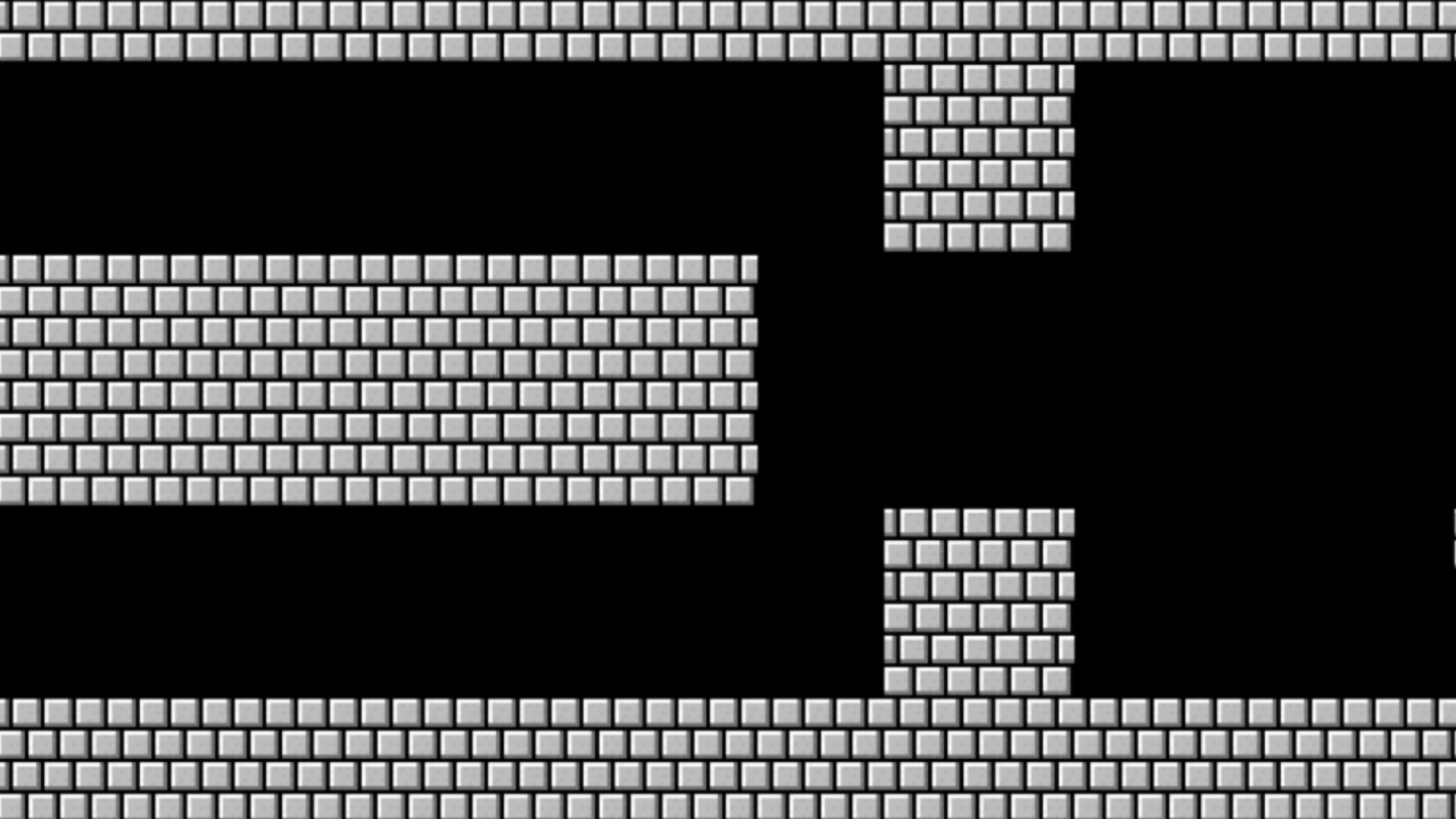
check50, design50, style50

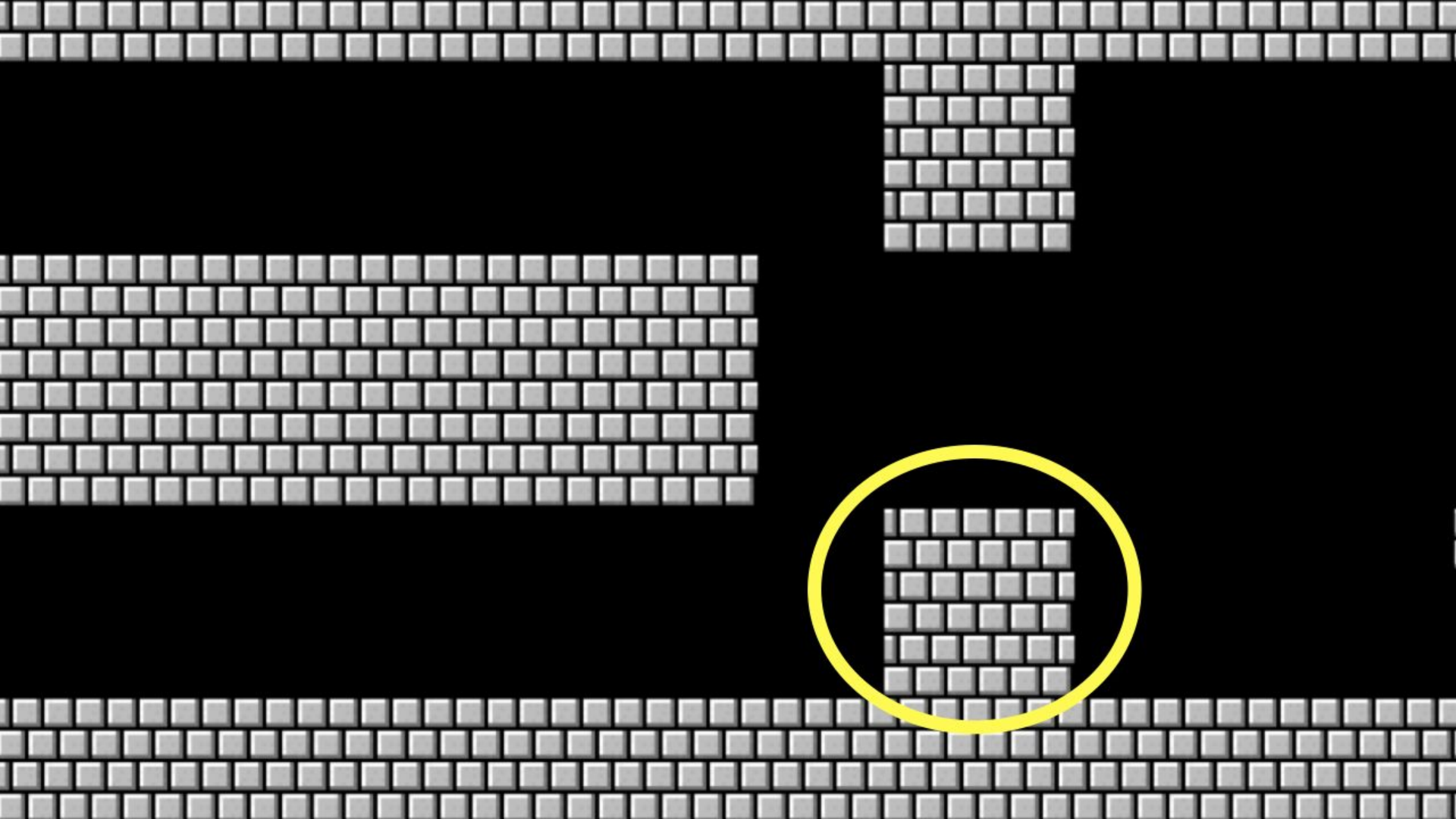

arguments
function
side effects





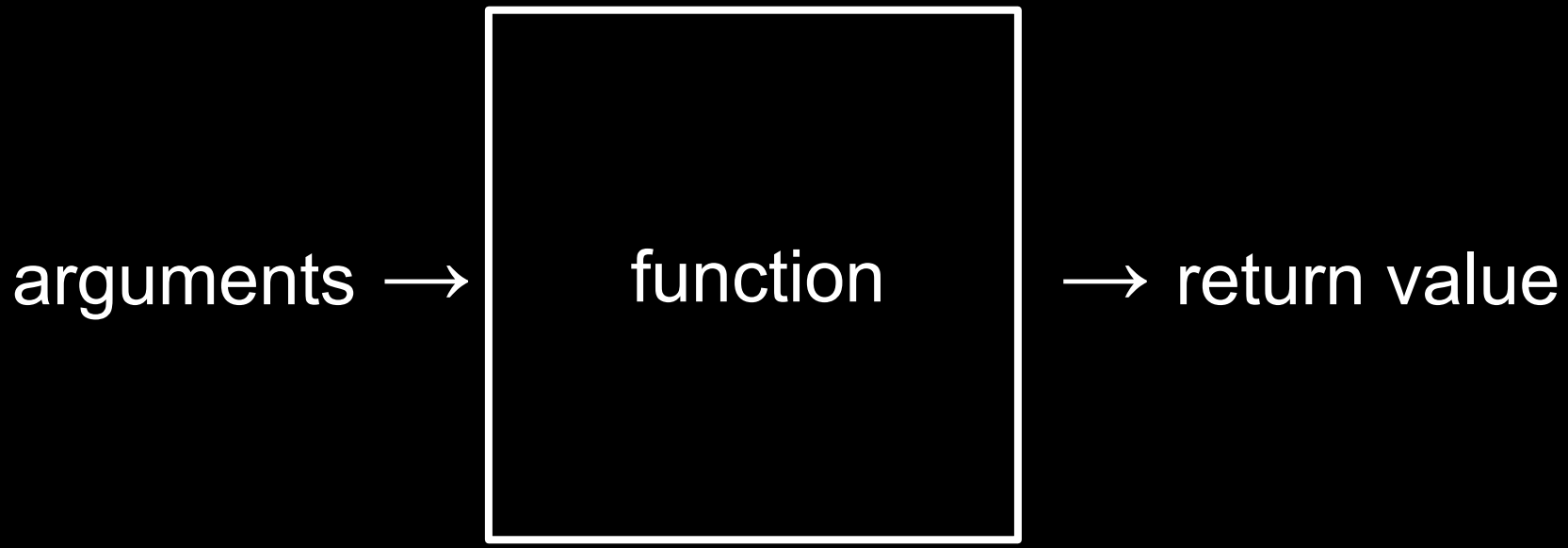






constants

comments


arguments
function
return value

+

-

*

/

%

…

integer overflow


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

0000

0001

0010

0011

0100

0101

0110

0111

1000

Using MAME to warp to level 256, the split screen is shown.

truncation

floating-point imprecision

1999

1999

1900

19 January 2038

This is CS50