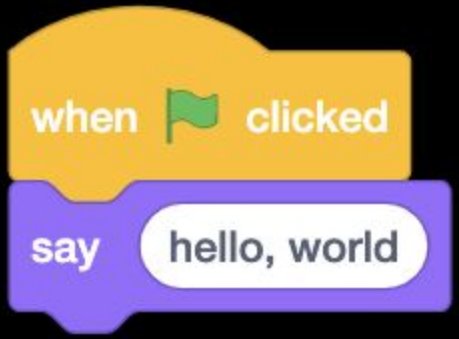
CS50 for MBAs

Python


when clicked
say
hello, world

```
print("hello, world")
```

cs50.harvard.edu/python

Programming Languages

- C
- R
- SQL
- Java
- Node.js [JavaScript]
- GoLang
- PHP
- C#
- HTML
- Spyder
- CSS
- C++
- MATLAB
- Objective-C
- ...

source code

```
01111111 01000101 01001100 01000110 00000010 00000001 00000001 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000001 00000000 00111110 00000000 00000001 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00101000 00000010 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 01000000 00000000 00000000 00000000
00000000 00000000 01000000 00000000 00001010 00000000 00000001 00000000
01010101 01001000 10001001 11100101 01001000 10000011 11101100 00010000
01001000 10111111 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 10110000 00000000 11101000 00000000 00000000 00000000
00000000 00110001 11001001 10001001 01000101 11111100 10001001 11001000
01001000 10000011 11000100 00010000 01011101 11000011 01101000 01100101
01101100 01101100 01101111 00101100 00100000 01110111 01101111 01110010
01101100 01100100 00001010 00000000 00000000 01100011 01101100 01100001
01101110 01100111 00100000 01110110 01100101 01110010 01110011 01101001
...
```

```
...
main:                                   # @main
        .cfi_startproc
# %bb.0:
        pushq   %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset %rbp, -16
        movq    %rsp, %rbp
        .cfi_def_cfa_register %rbp
        subq    $16, %rsp
        movabsq $.L.str, %rdi
        movb    $0, %al
        callq   printf
        xorl    %ecx, %ecx
        movl    %eax, -4(%rbp)          # 4-byte Spill
        movl    %ecx, %eax
        addq    $16, %rsp
        popq    %rbp
        retq
.Lfunc_end0:
        .size   main, .Lfunc_end0-main
        .cfi_endproc
                                        # -- End function
        .type   .L.str,@object          # @.str
        .section        .rodata.str1.1,"aMS",@progbits,1
.L.str:
        .asciz  "hello, world\n"
        .size   .L.str, 14
...
```

```c
#include <stdio.h>

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

```
#include <iostream>

int main()
{
    std::cout << "hello, world" << std::endl;
}
```

```
class Hello
{
    public static void main(String [] args)
    {
        System.out.println("hello, world");
    }
}
```

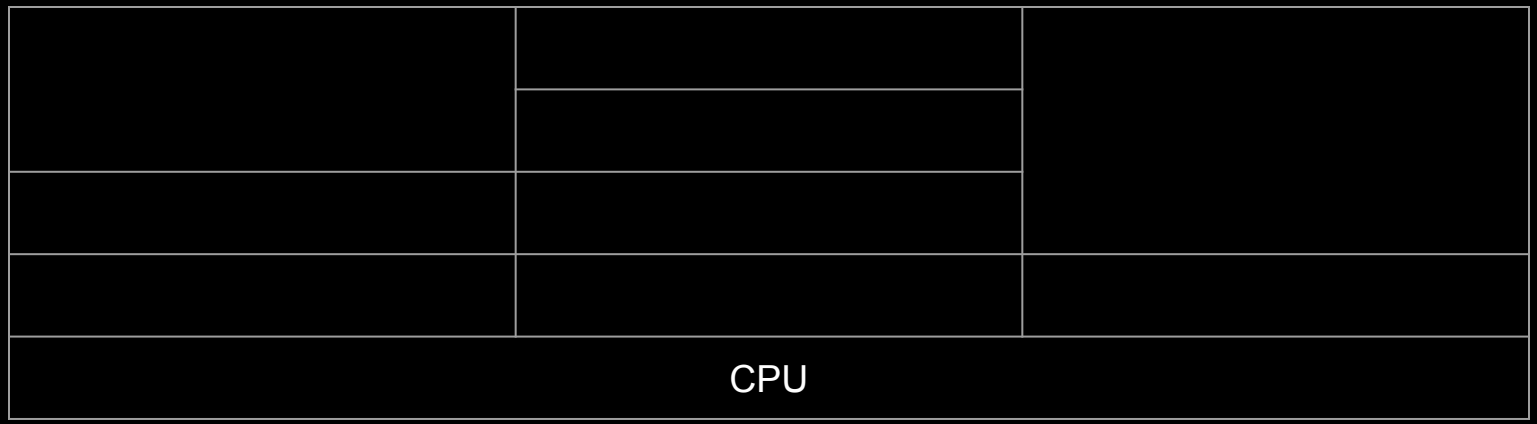
```
print("hello, world")
```

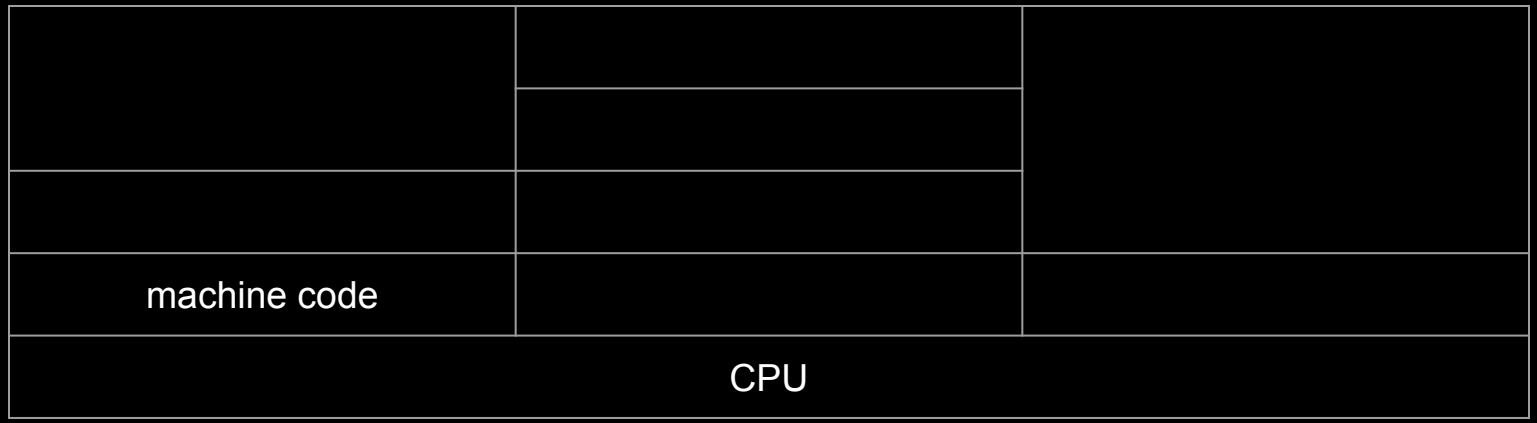
wikipedia.org/wiki/List_of_programming_languages

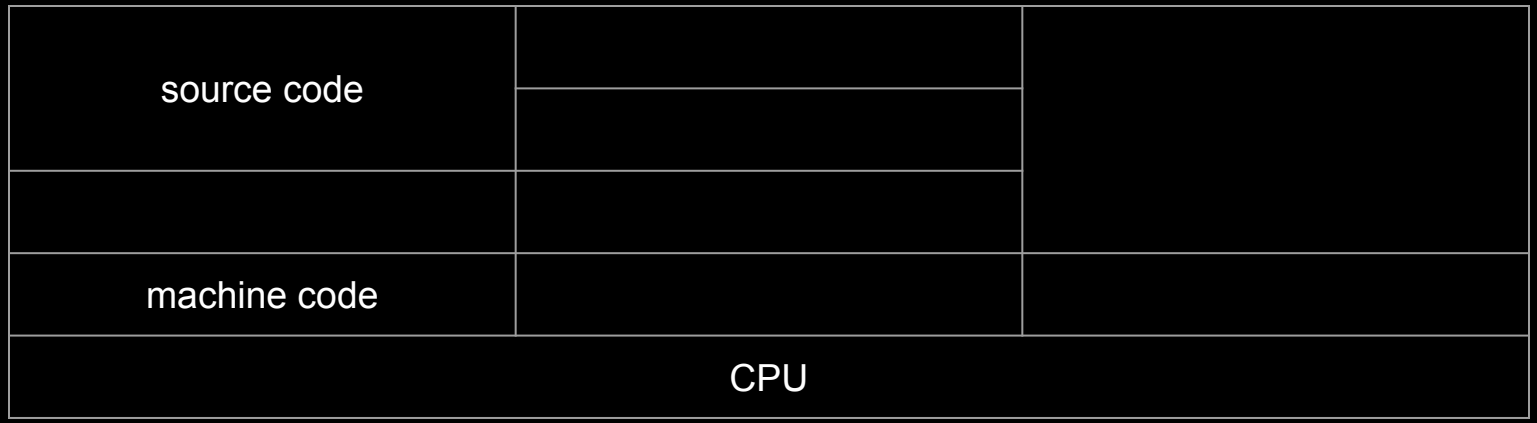
helloworldcollection.de

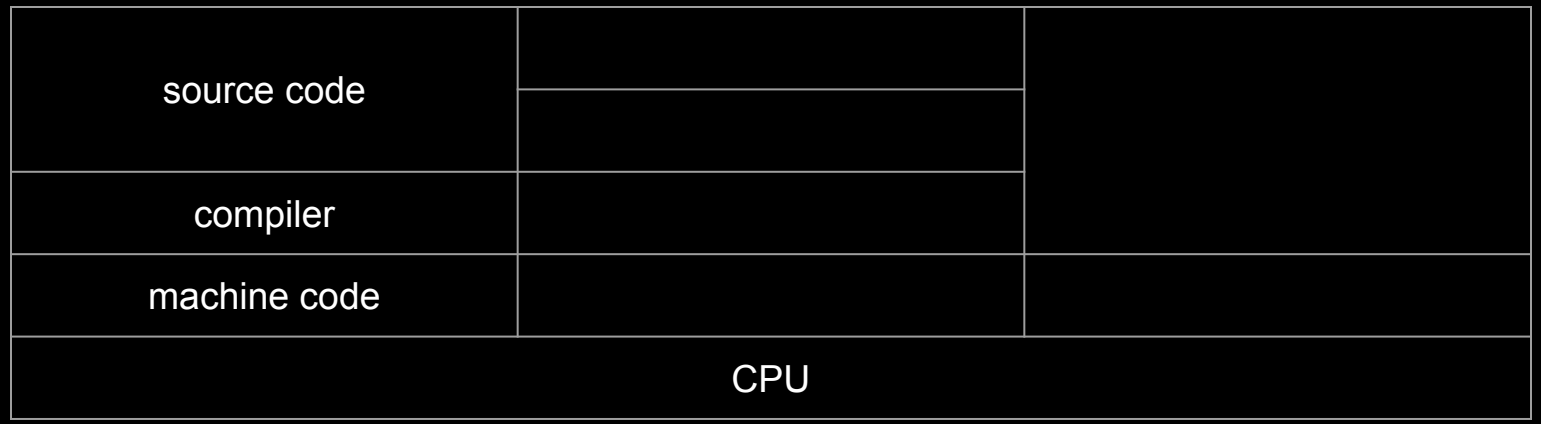
```
01111111 01000101 01001100 01000110 00000010 00000001 00000001 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000001 00000000 00111110 00000000 00000001 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00101000 00000010 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 01000000 00000000 00000000 00000000
00000000 00000000 01000000 00000000 00001010 00000000 00000001 00000000
01010101 01001000 10001001 11100101 01001000 10000011 11101100 00010000
01001000 10111111 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 10110000 00000000 11101000 00000000 00000000 00000000
00000000 00110001 11001001 10001001 01000101 11111100 10001001 11001000
01001000 10000011 11000100 00010000 01011101 11000011 01101000 01100101
01101100 01101100 01101111 00101100 00100000 01110111 01101111 01110010
01101100 01100100 00001010 00000000 00000000 01100011 01101100 01100001
01101110 01100111 00100000 01110110 01100101 01110010 01110011 01101001
...
```

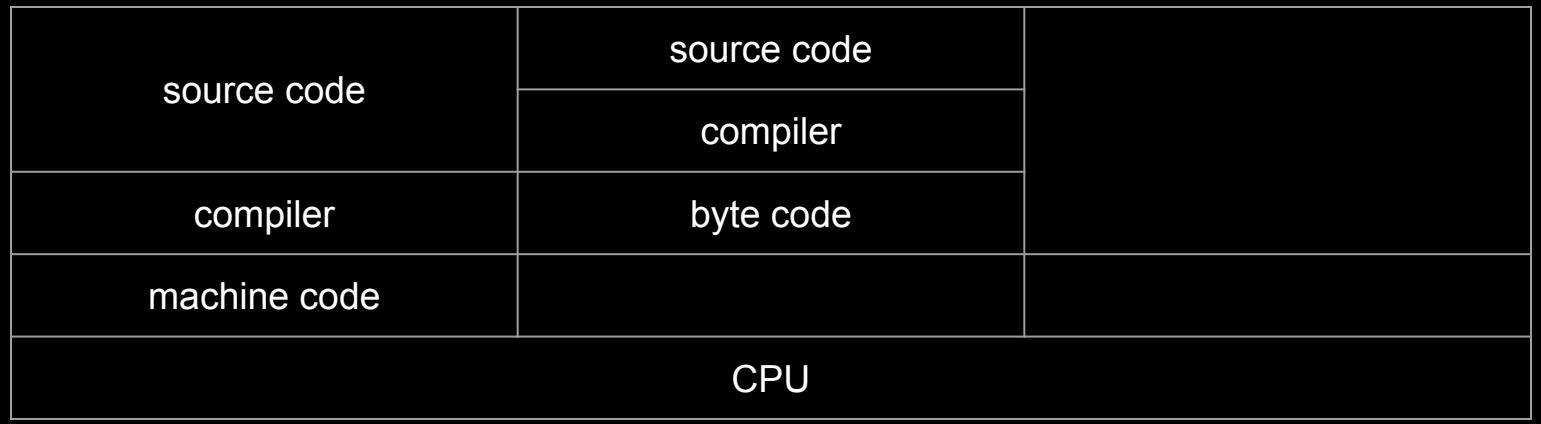
machine code

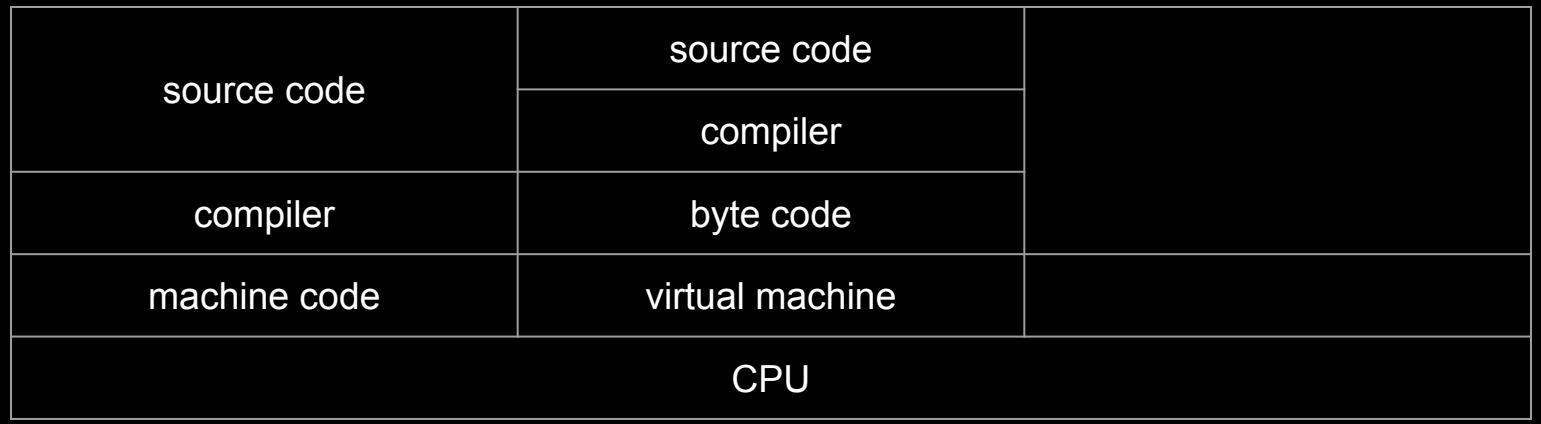

CPU


machine code
CPU


source code
machine code
CPU


source code
compiler
machine code
CPU


source code
source code
compiler
compiler
byte code
machine code
CPU


source code
source code
compiler
compiler
byte code
machine code
virtual machine
CPU

<table>
<tr><td rowspan="2">source code</td><td>source code</td><td rowspan="2">source code</td></tr>
<tr><td>compiler</td></tr>
<tr><td>compiler</td><td>byte code</td></tr>
<tr><td>machine code</td><td>virtual machine</td><td>interpreter</td></tr>
<tr><td colspan="3">CPU</td></tr>
</table>

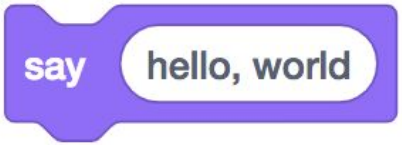
```
print("hello, world")
```

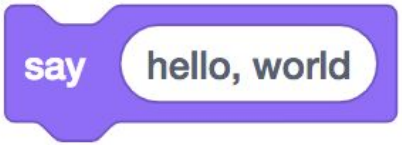
VS Code

code.cs50.io

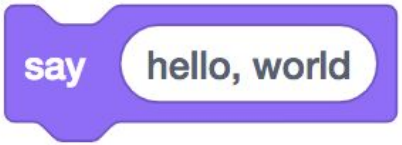
```
python hello.py
```

Python

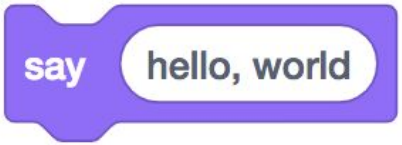

say
hello, world




```
print(                )
```



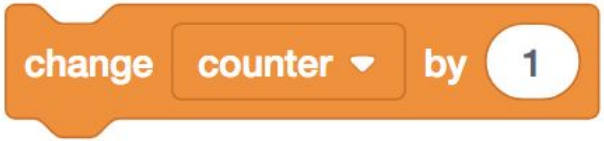

```
print( hello, world )
```

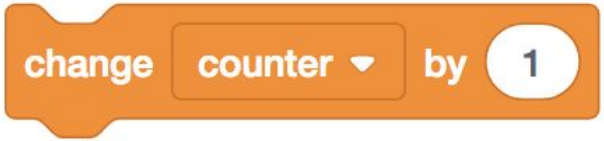



```
print("hello, world")
```

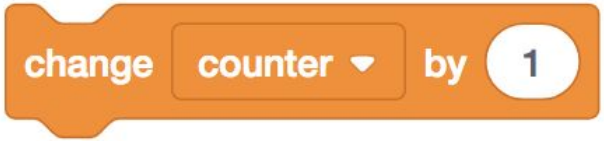

set
counter
to
0


```
counter = 0
```


change
counter
by
1



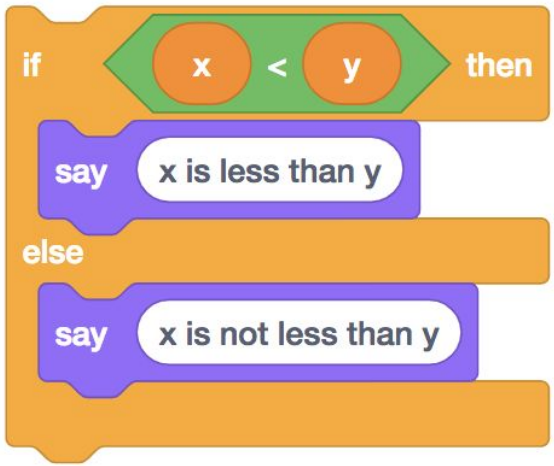

```
counter = counter + 1
```

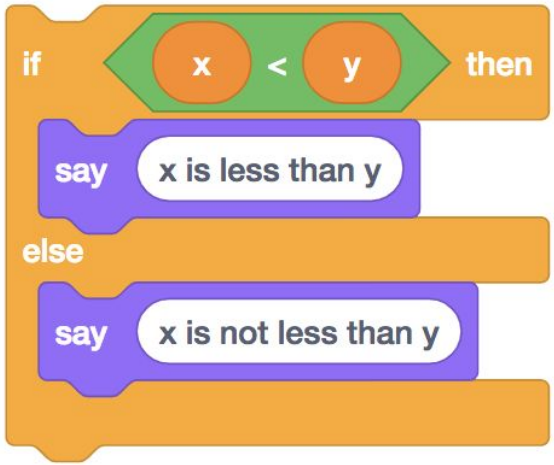



```
counter += 1
```

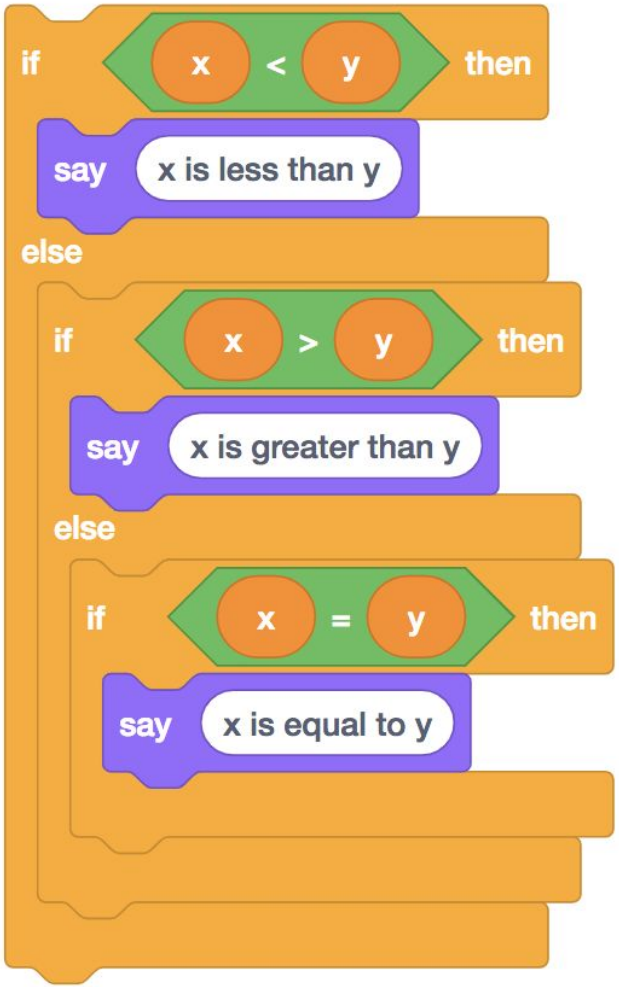

if
x
<
y
then
say
x is less than y

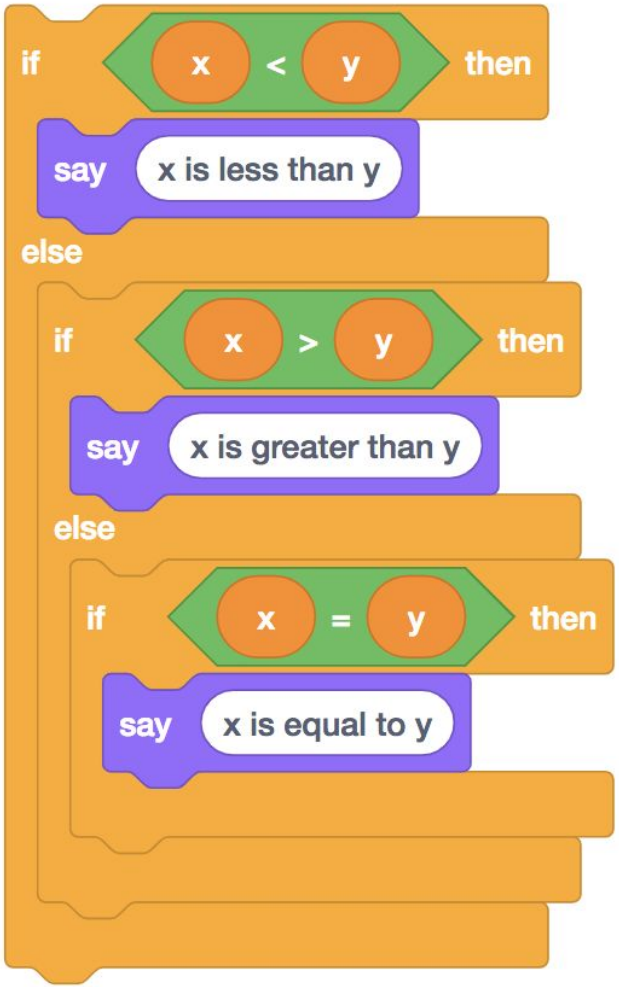

```
if x < y:
    print("x is less than y")
```


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

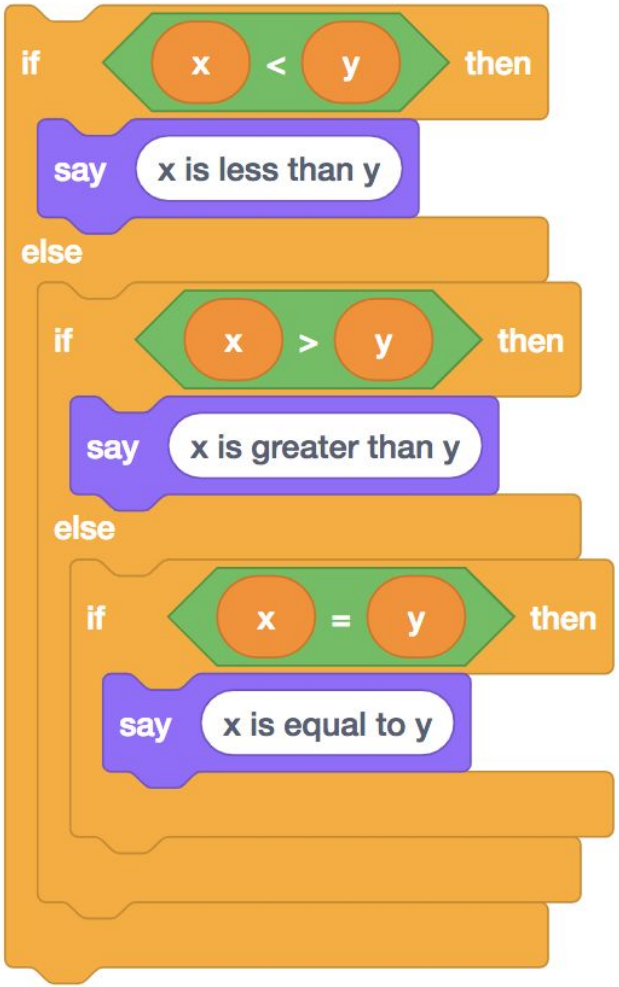


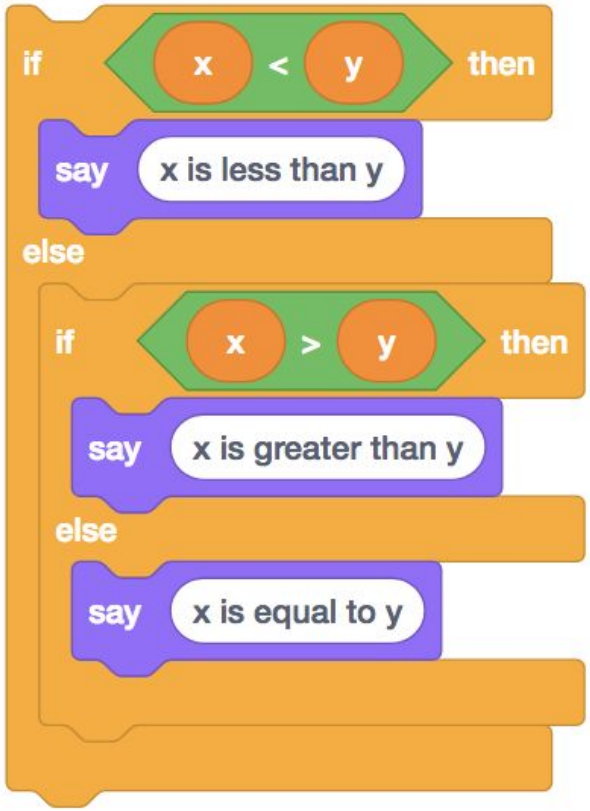

```
if x < y:
    print("x is less than y")
else:
    print("x is not less than y")
```

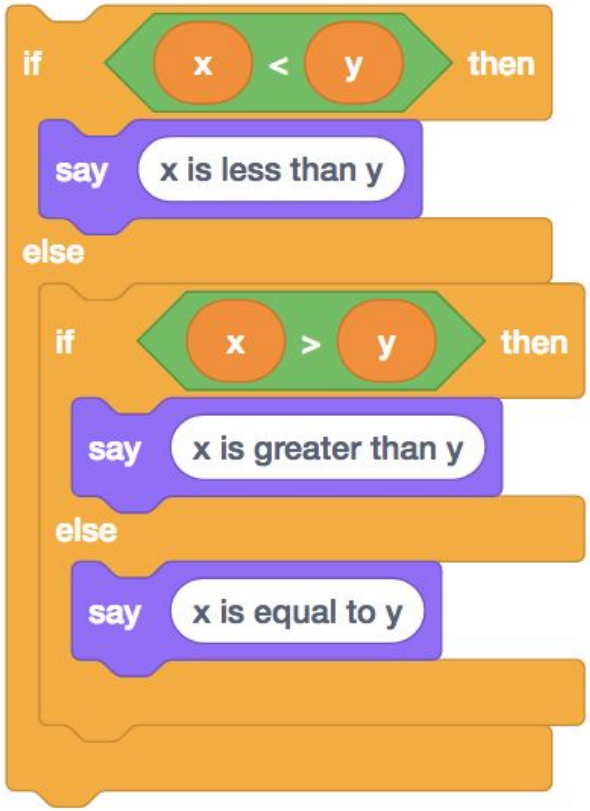

if x < y then
say x is less than y
else
if x > y then
say x is greater than y
else
if x = y then
say x is equal to y



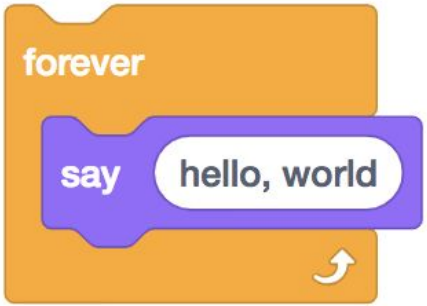

```
if x < y:
    print("x is less than y")
elif x > y:
    print("x is greater than y")
elif x == y:
    print("x is equal to y")
```

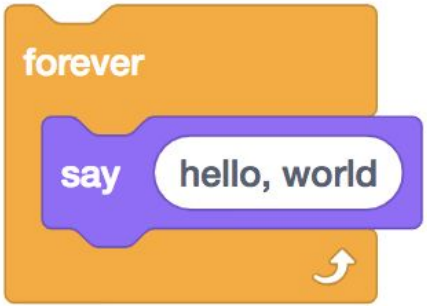

if x < y then
say x is less than y
else
if x > y then
say x is greater than y
else
if x = y then
say x is equal to y


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

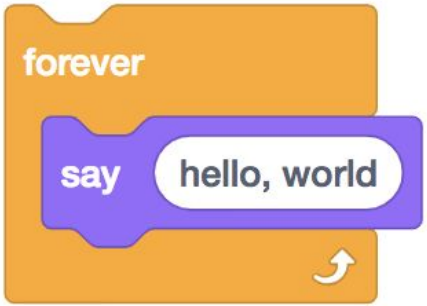



```
if x < y:
    print("x is less than y")
elif x > y:
    print("x is greater than y")
else:
    print("x is equal to y")
```

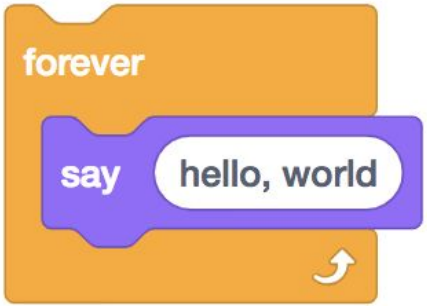

forever
say
hello, world




```
while
```




```
while
    print("hello, world")
```




```
while True:
    print("hello, world")
```


repeat
3
say
hello, world

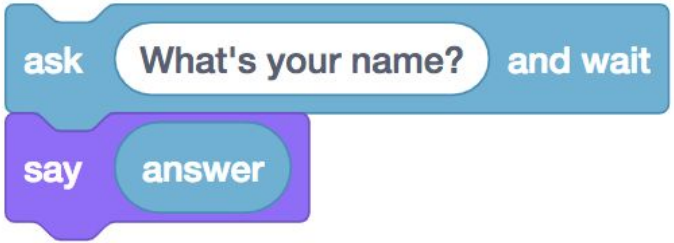

```
i = 0
while i < 3:
    print("hello, world")
    i += 1
```

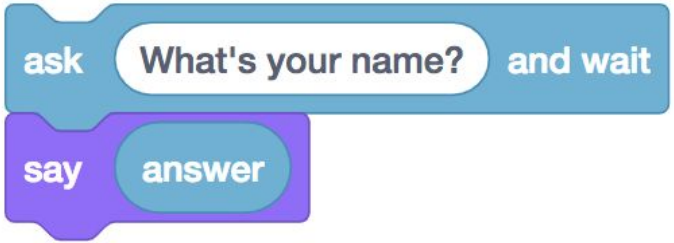

```
i = 1
while i <= 3:
    print("hello, world")
    i += 1
```


repeat
3
say
hello, world

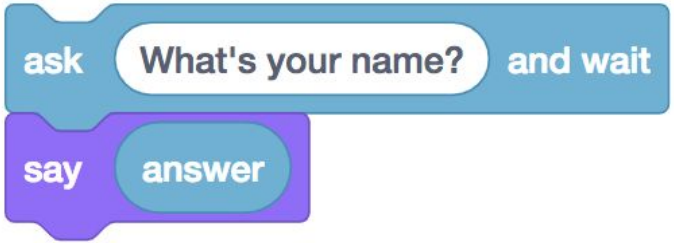

```
for i in [0, 1, 2]:
    print("hello, world")
```


```
for i in range(3):
    print("hello, world")
```


ask
What's your name?
and wait
say
answer




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




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


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


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


```
answer = input("What's your name? ")
print("hello, " + answer)
```


```
answer = input("What's your name? ")
print("hello,", answer)
```

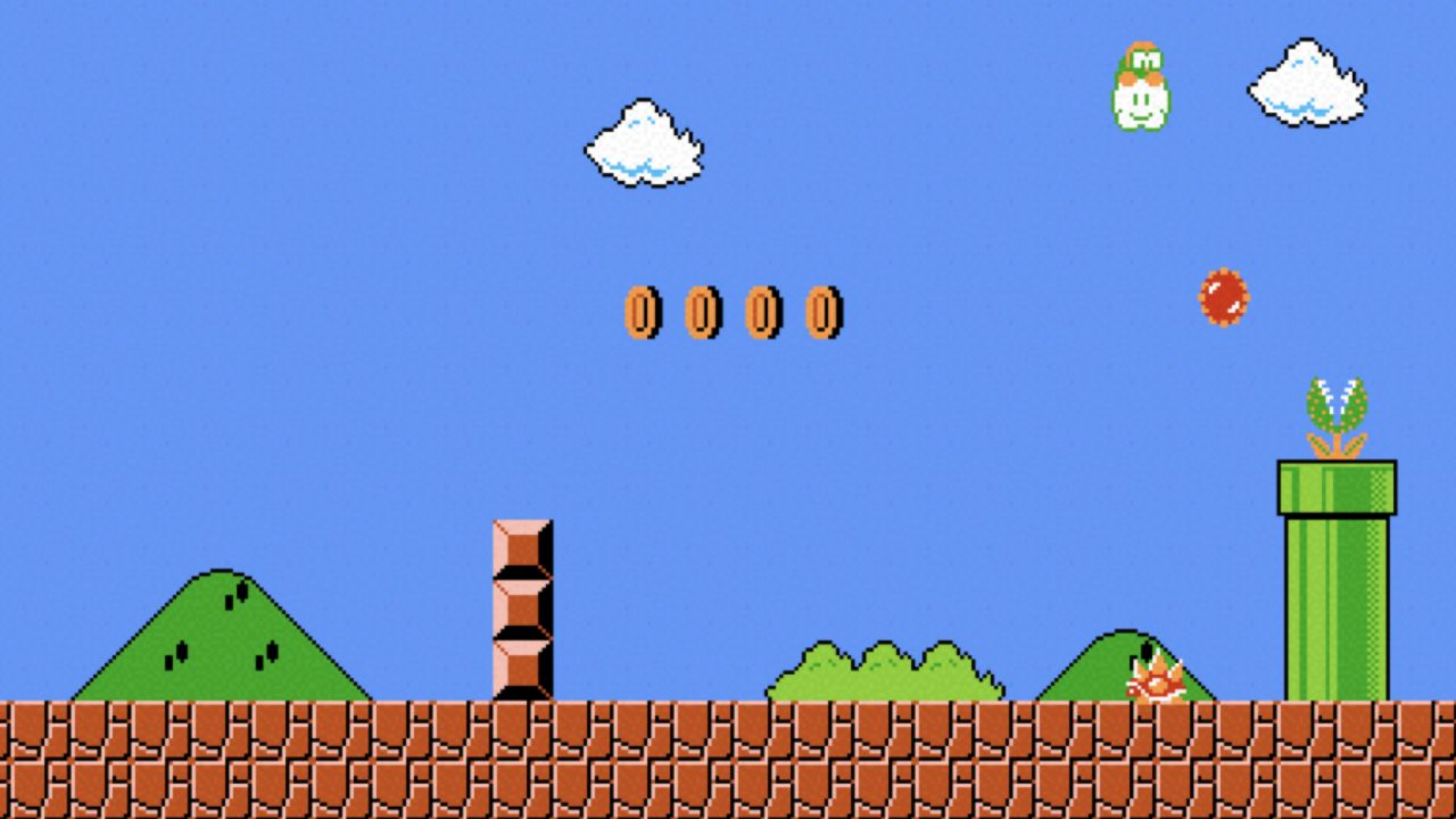

```
answer = input("What's your name? ")
print(f"hello, {answer}")
```

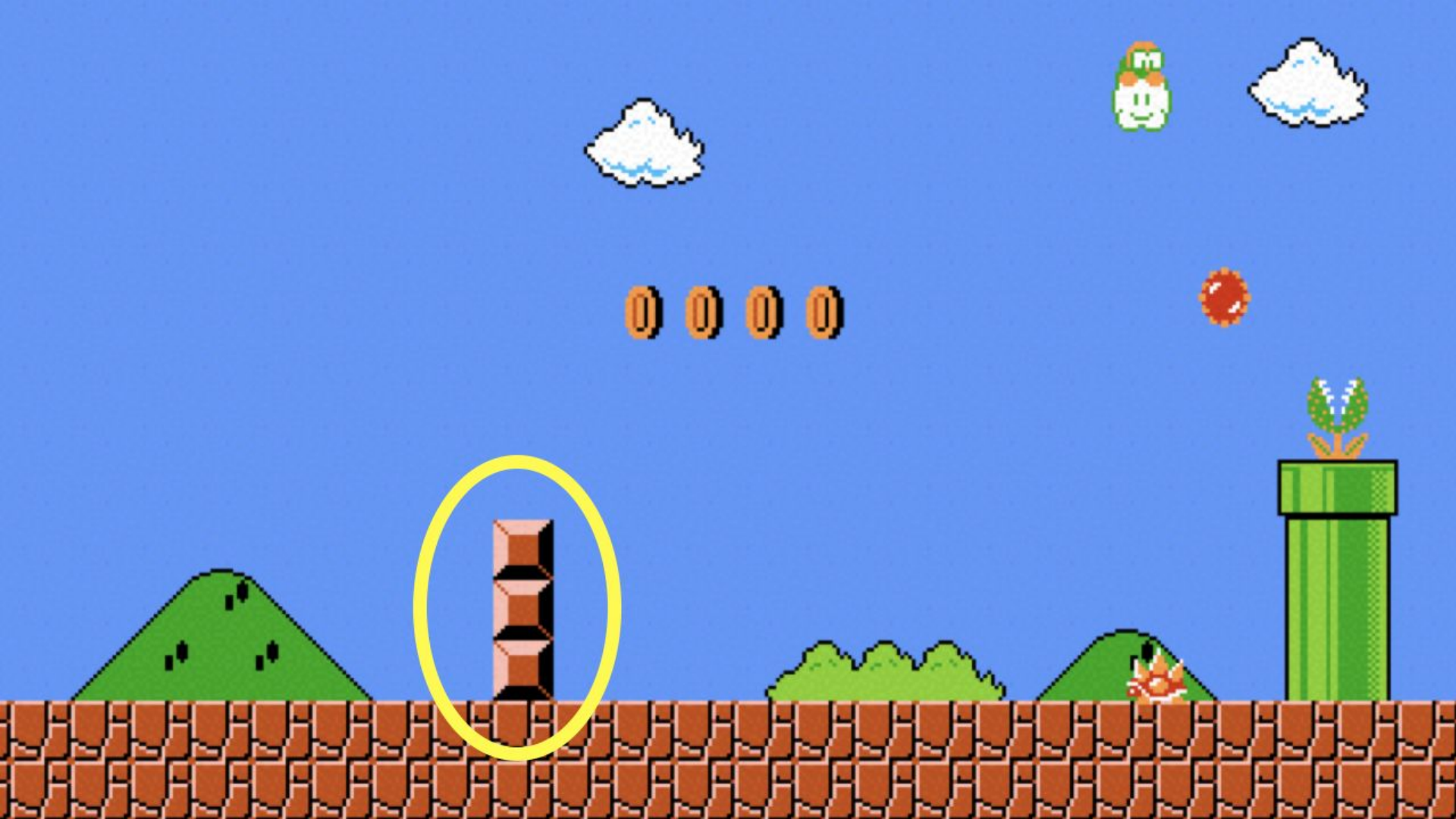
`bool`	Boolean value
`float`	floating-point value
`int`	integer
`str`	string
`...`	

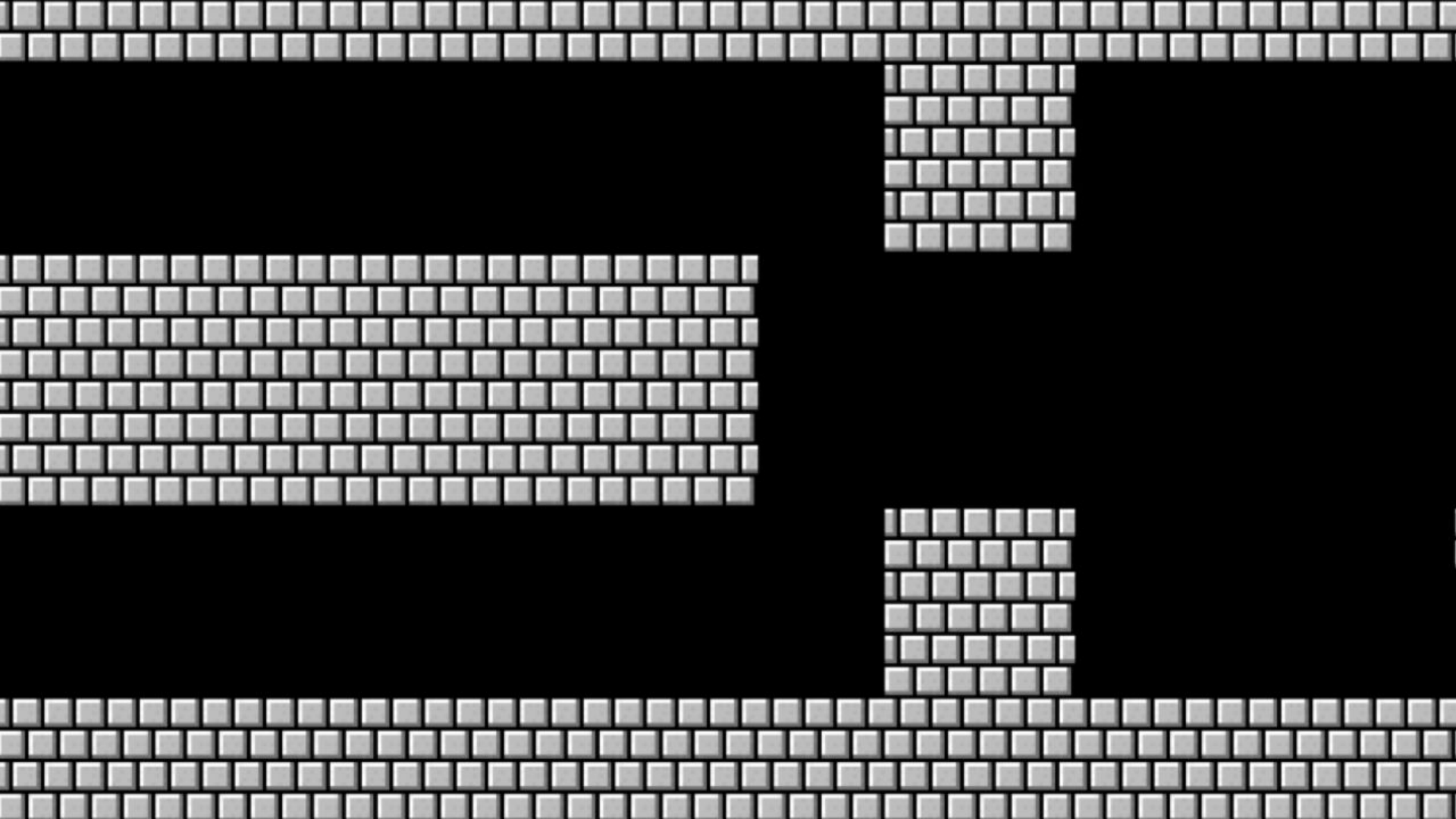
`range`	sequence of numbers
`list`	sequence of mutable values
`tuple`	sequence of immutable values
`dict`	collection of key-value pairs
`set`	collection of unique values
`...`	

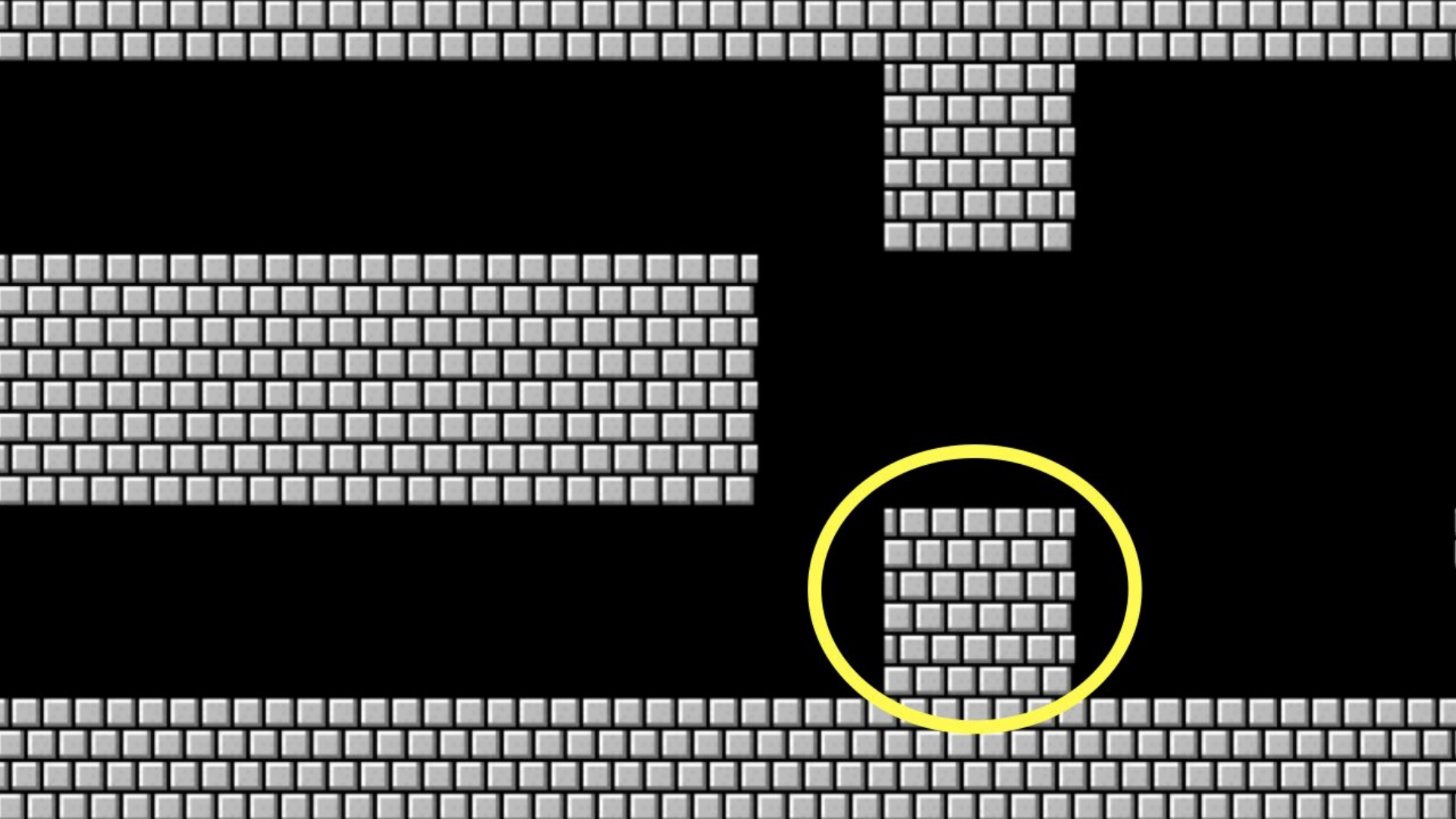
docs.python.org

docs.python.org/3/library/stdtypes.html#text-sequence-type-str









floating-point imprecision

integer overflow

1 2 3

1 2 4

1 2 5

1 2 6

1 2 7

1 2 8

1 2 9

1 2 10

1 2 9

1

1 2 0

1 3 0

9 9 9

1

9 9 0

1

9 0 0

1

0 0 0

1 0 0 0

0 0 0

Assignment 2

Office Hours

Lab 0

CS50 for MBAs

Python