# Welcome to Transcode

Lecture 1: What is this programming thing anyway

# That programming stuff

## How does a computer work?

Parts:

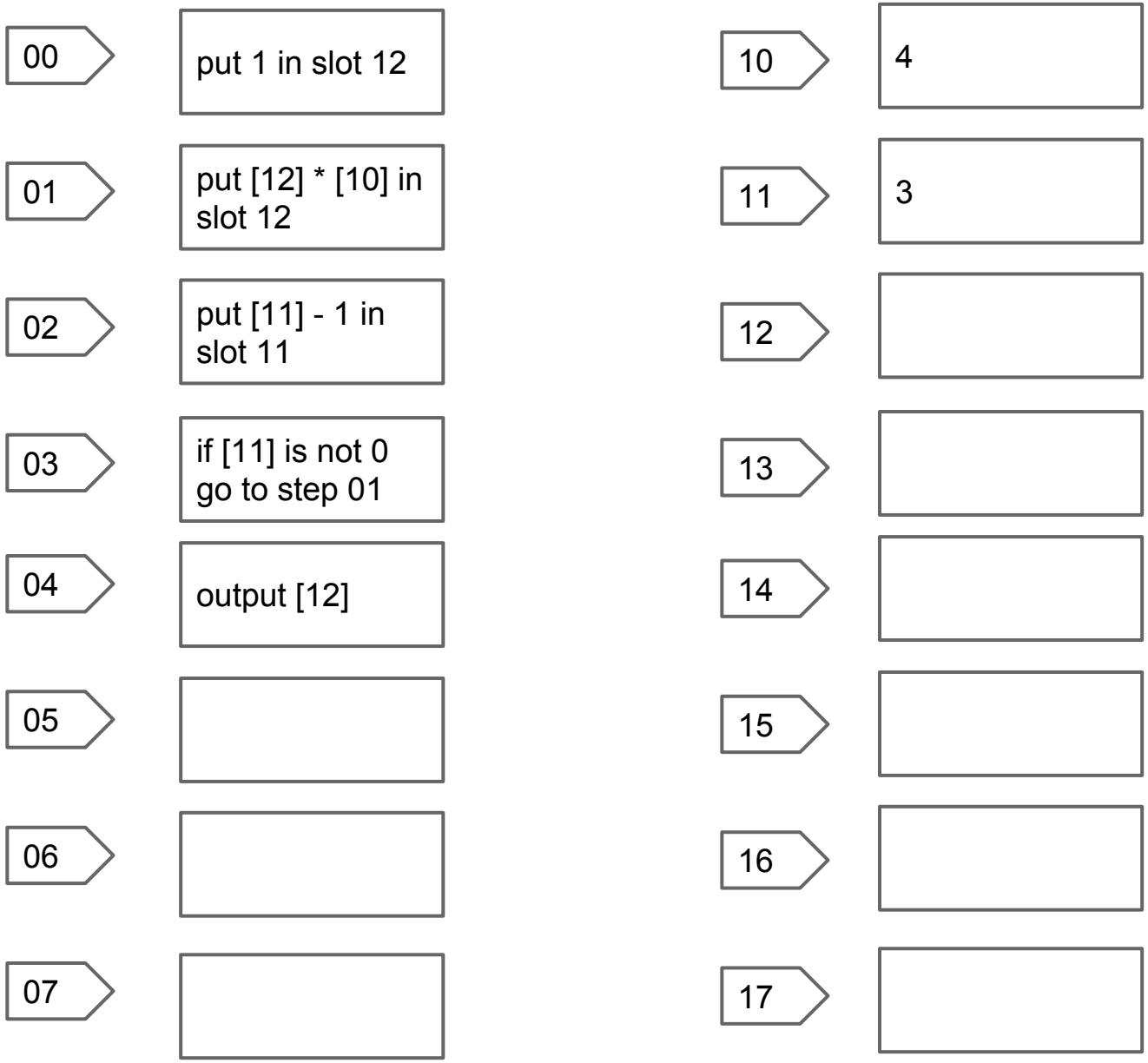
- Memory.  Numbered buckets with numbers in them.
- A set of **instructions** which can be kept in memory.
- A **program counter** which keeps track of what instruction you're on.
- An arithmetic logic unit, which makes calculations

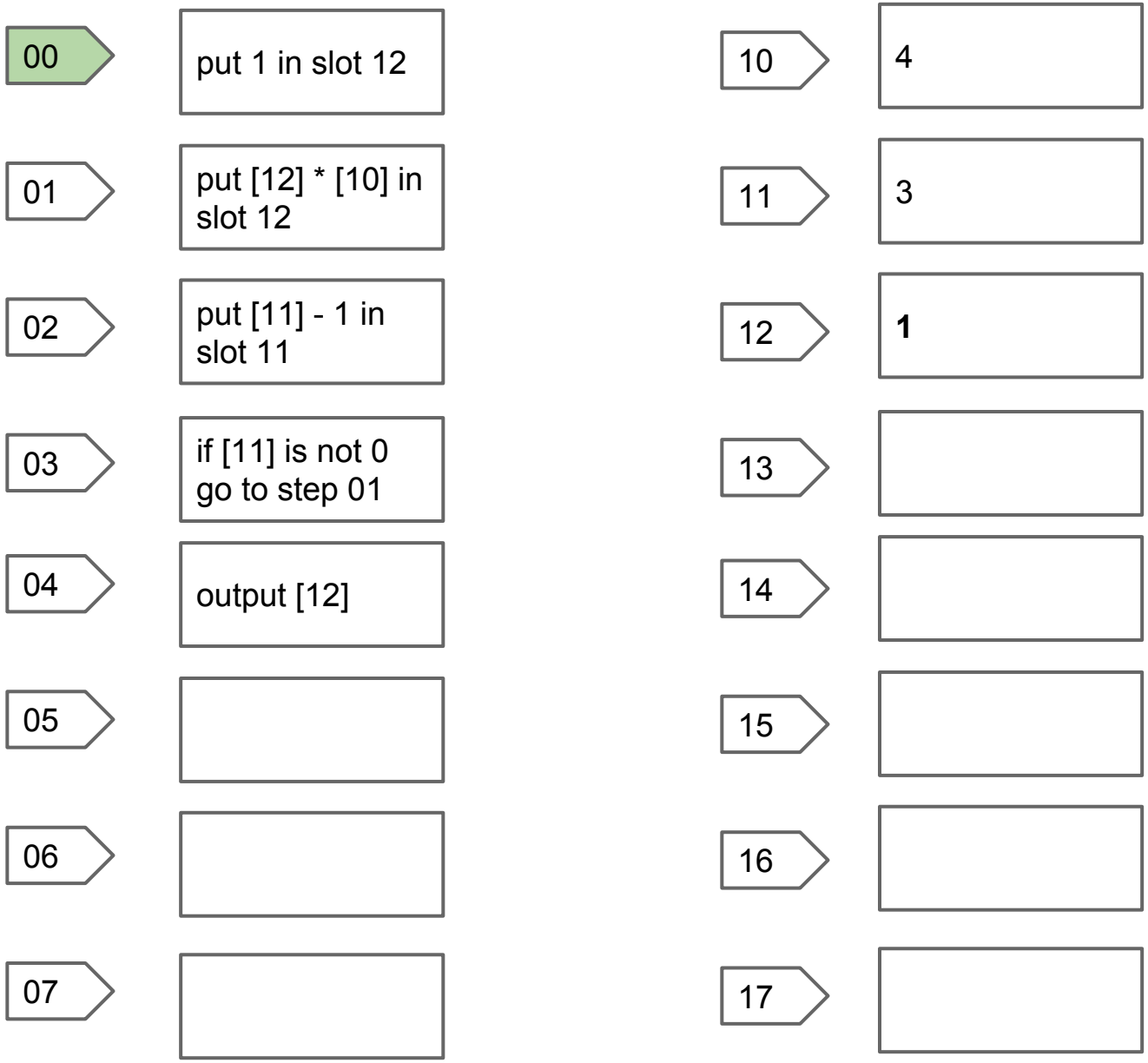
## How does a computer work?

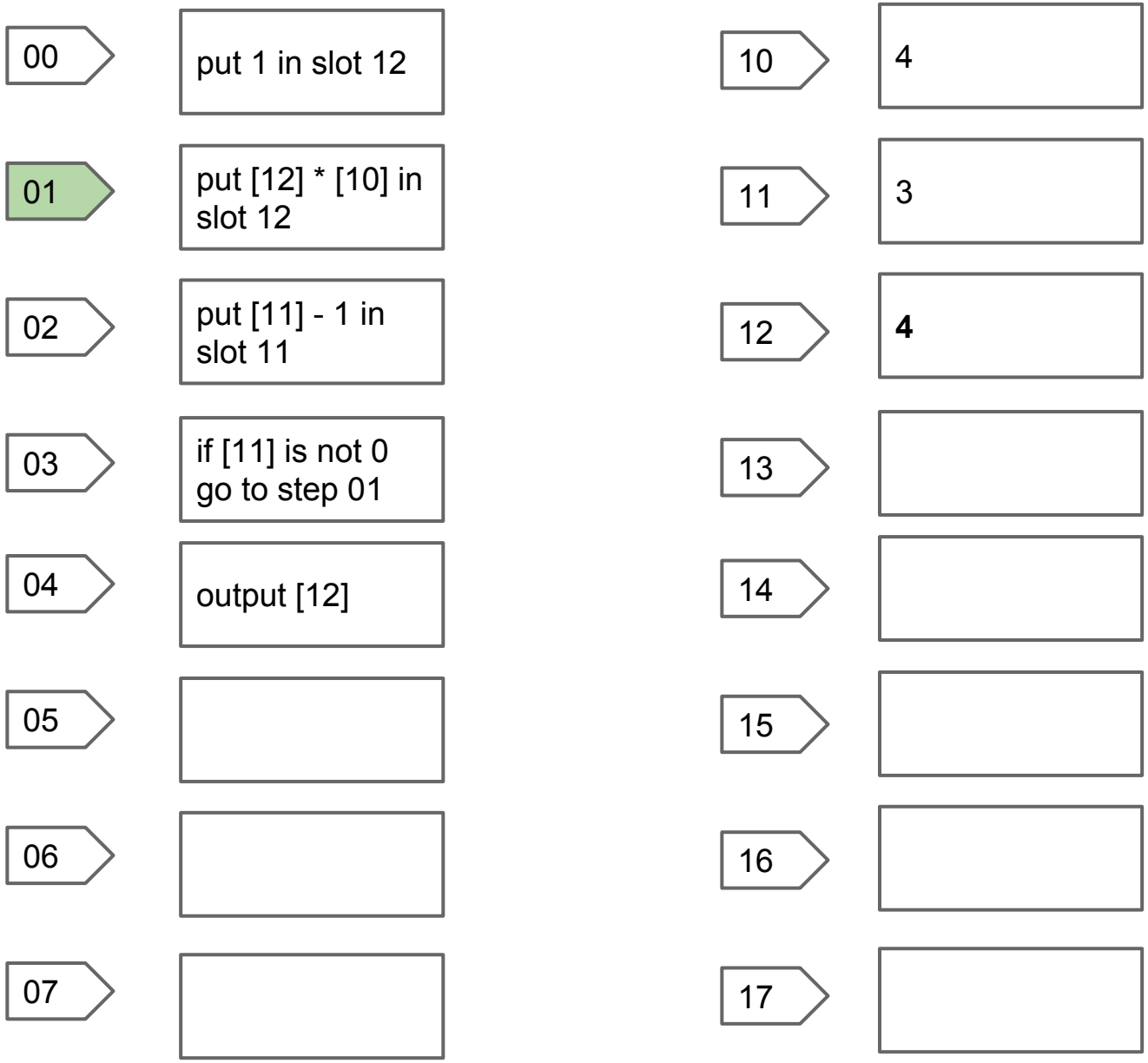
What does it do:

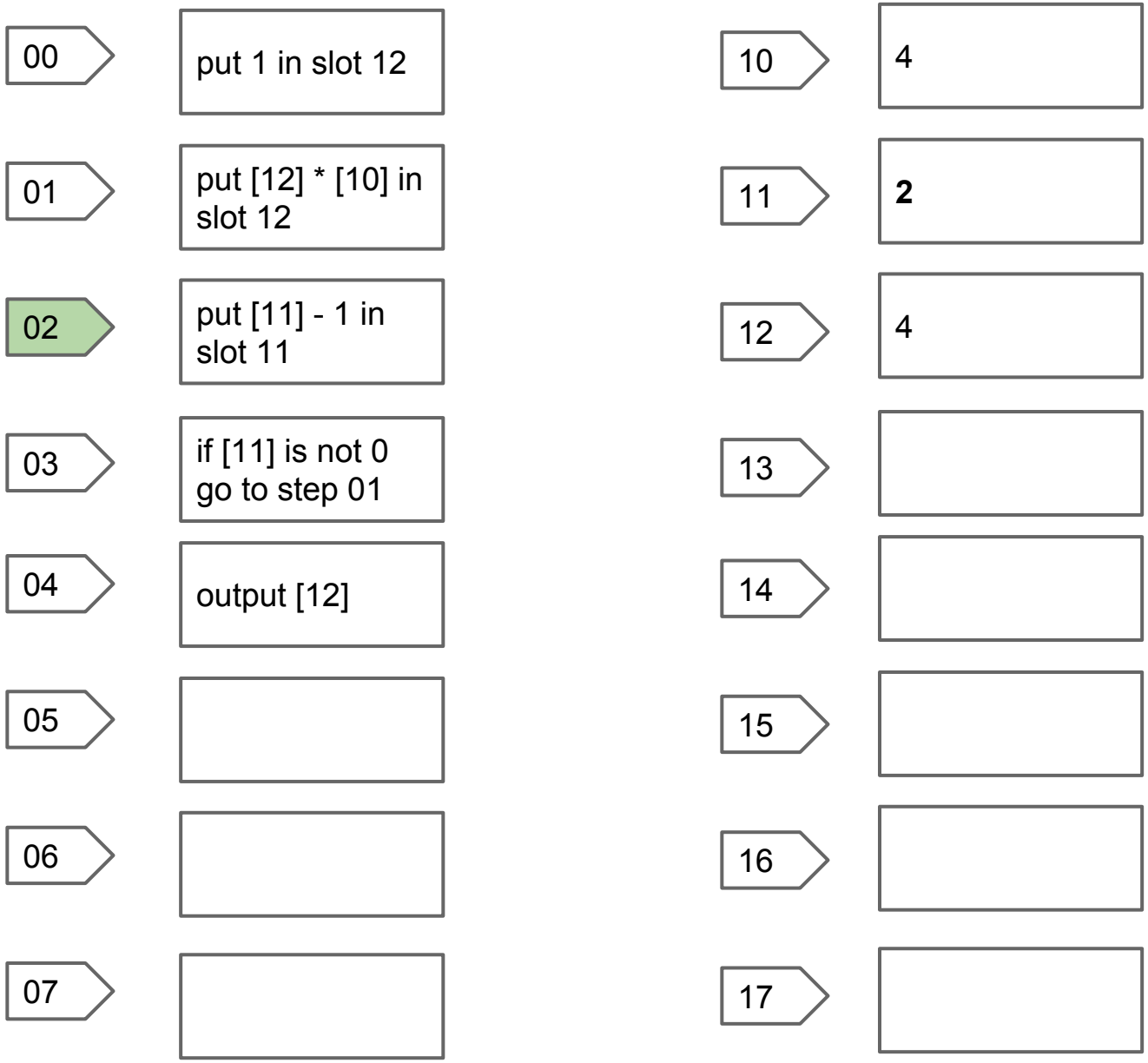
- Looks at where the program counter points
- Interprets that as an **instruction** with **arguments**
- Loads the **arguments** from memory into the ALU and **executes** the instruction
- Stores the **result** back to memory
- Adjusts the program counter
  - Usually just the next instruction
  - Sometimes, jumps to a different one

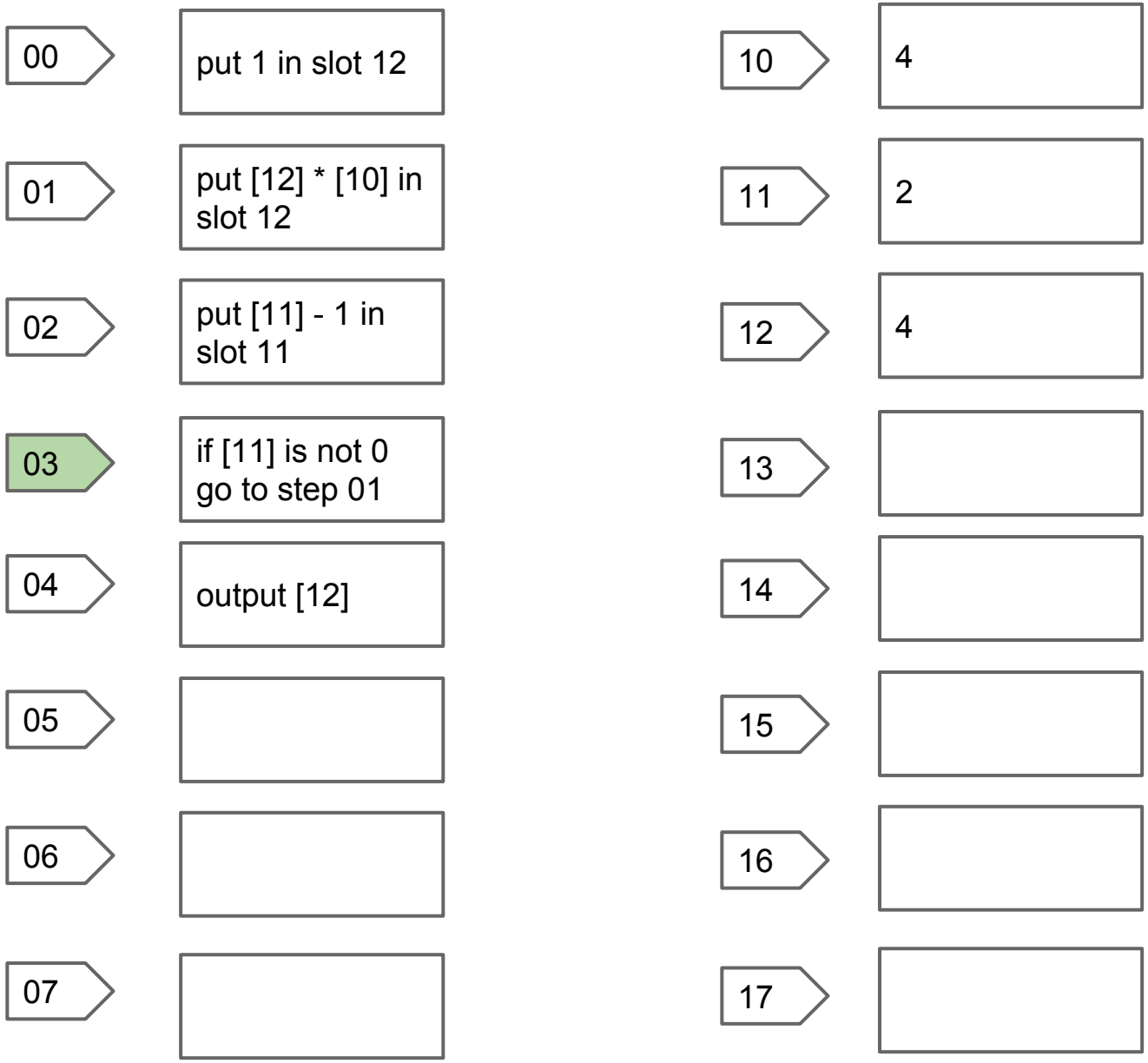
# Let's do an example

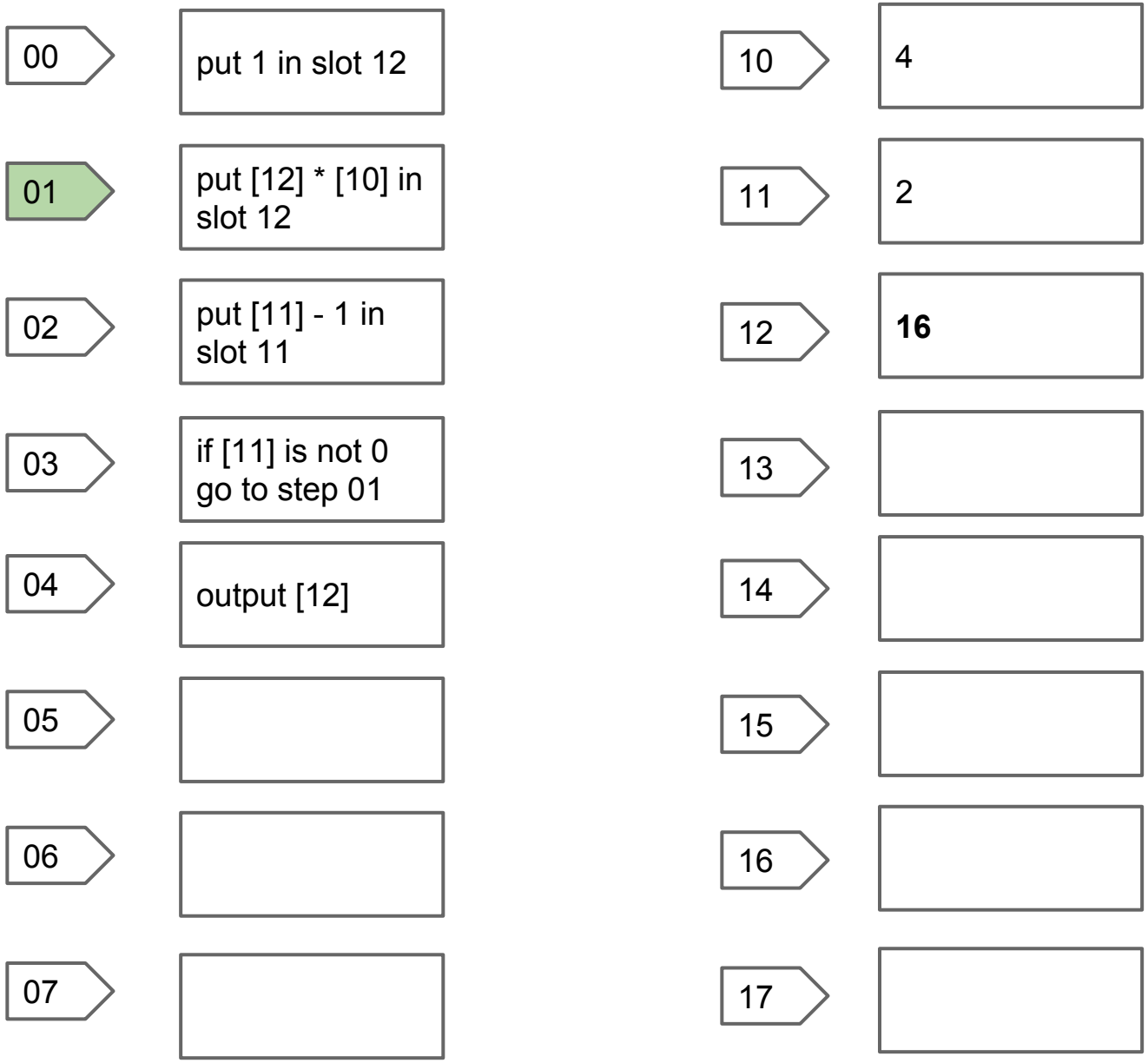

00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
3
12
13
14
15
16
17

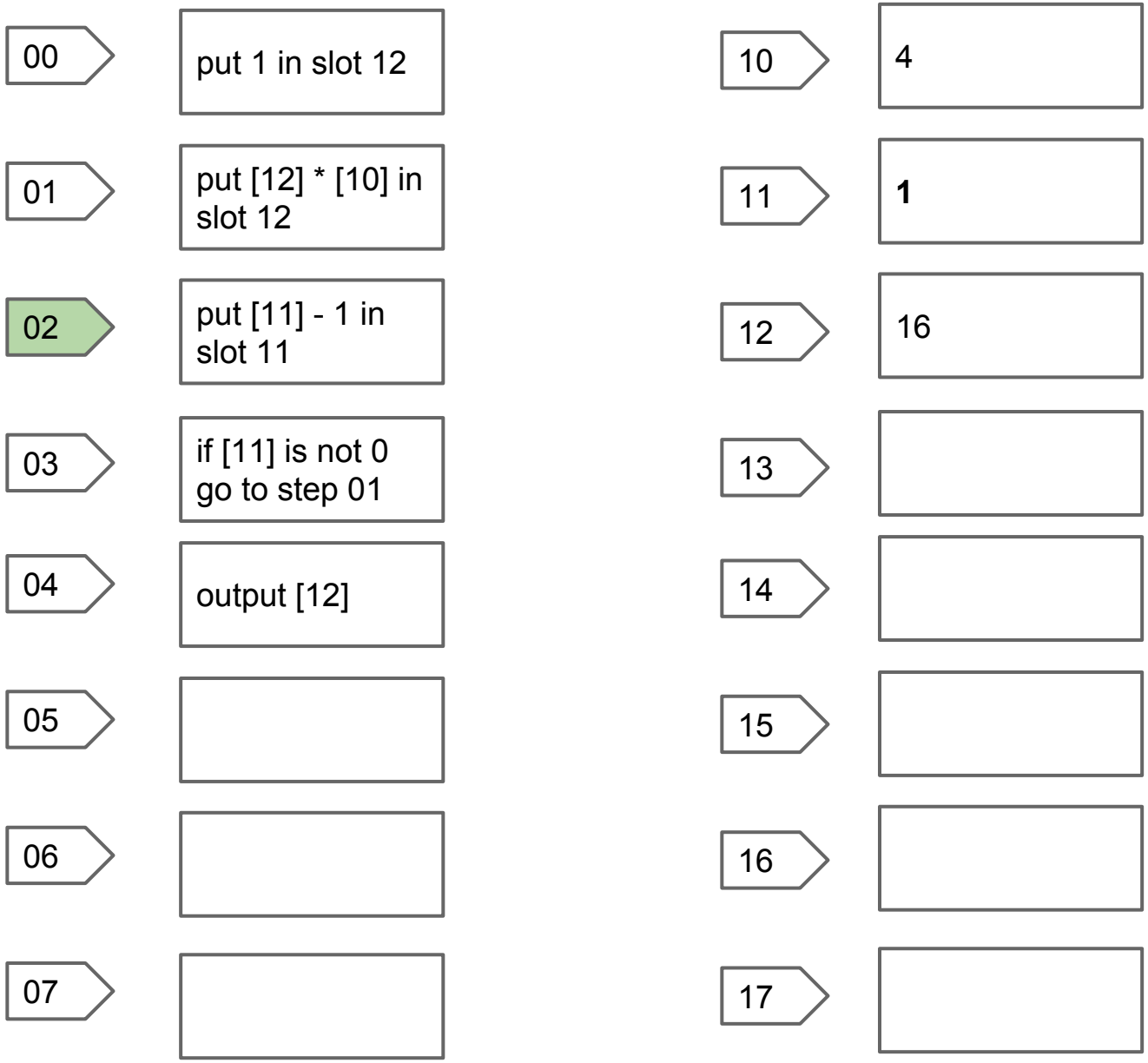

00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
3
12
1
13
14
15
16
17

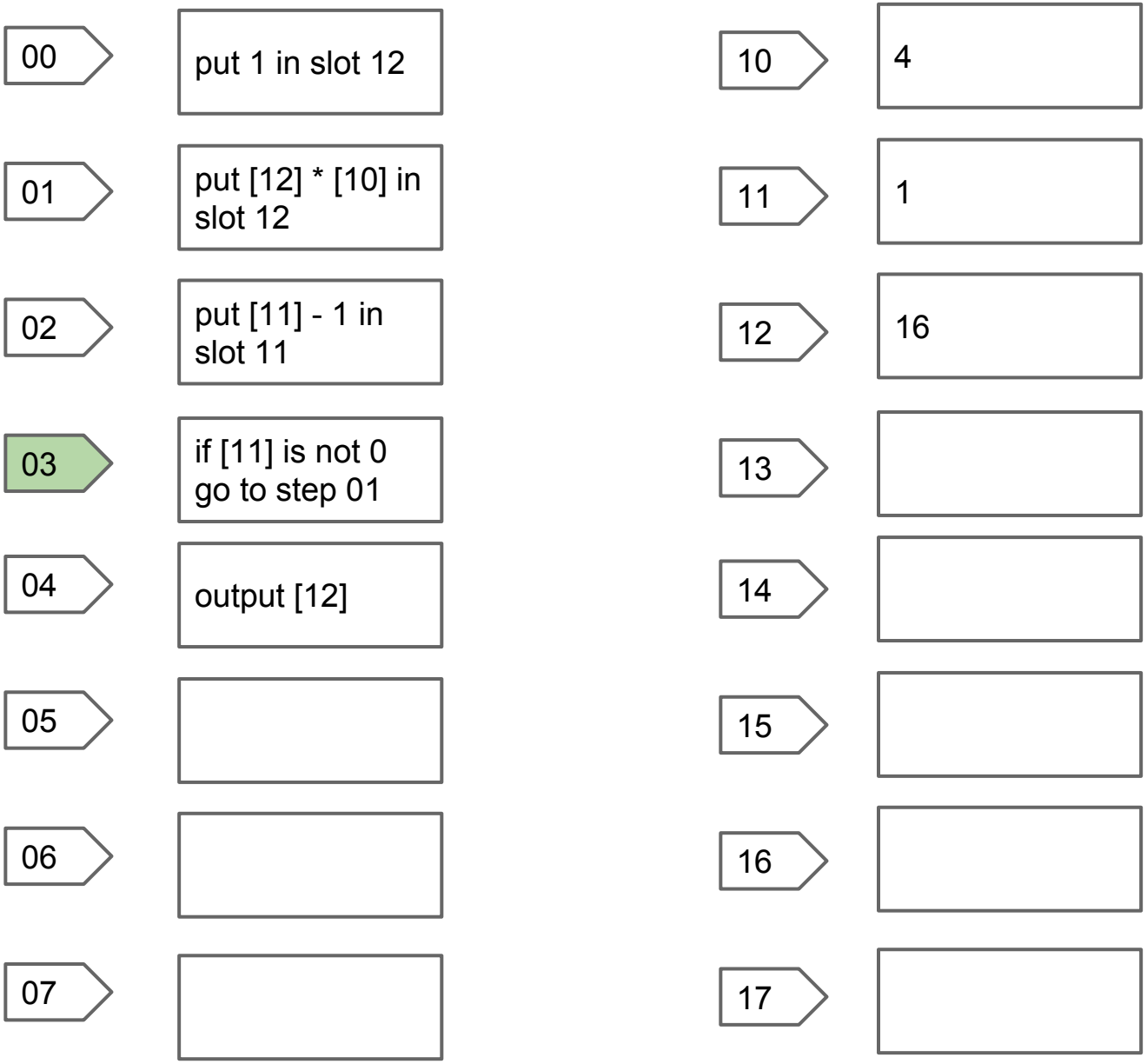

00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
3
12
4
13
14
15
16
17

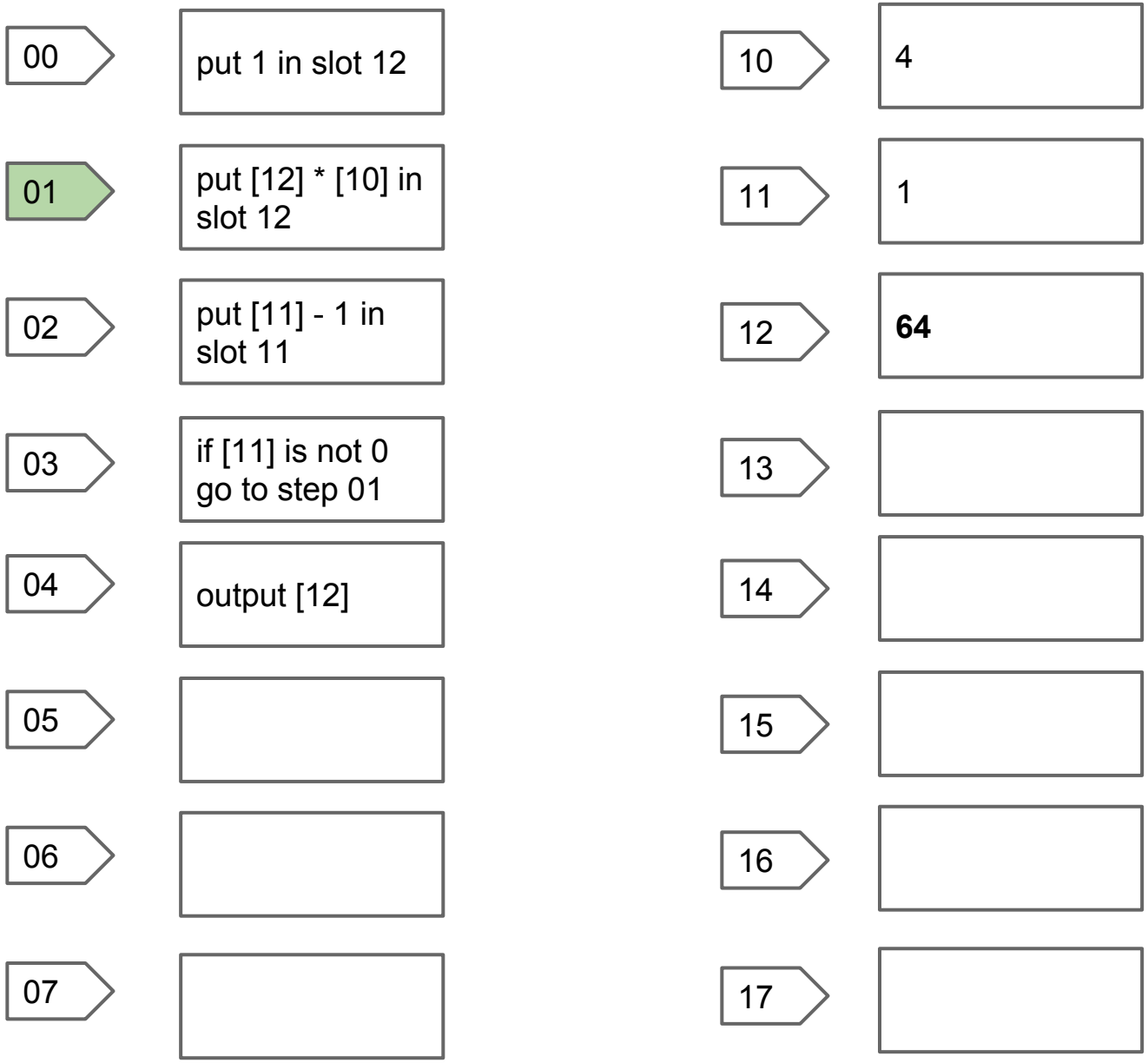

00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
2
12
4
13
14
15
16
17

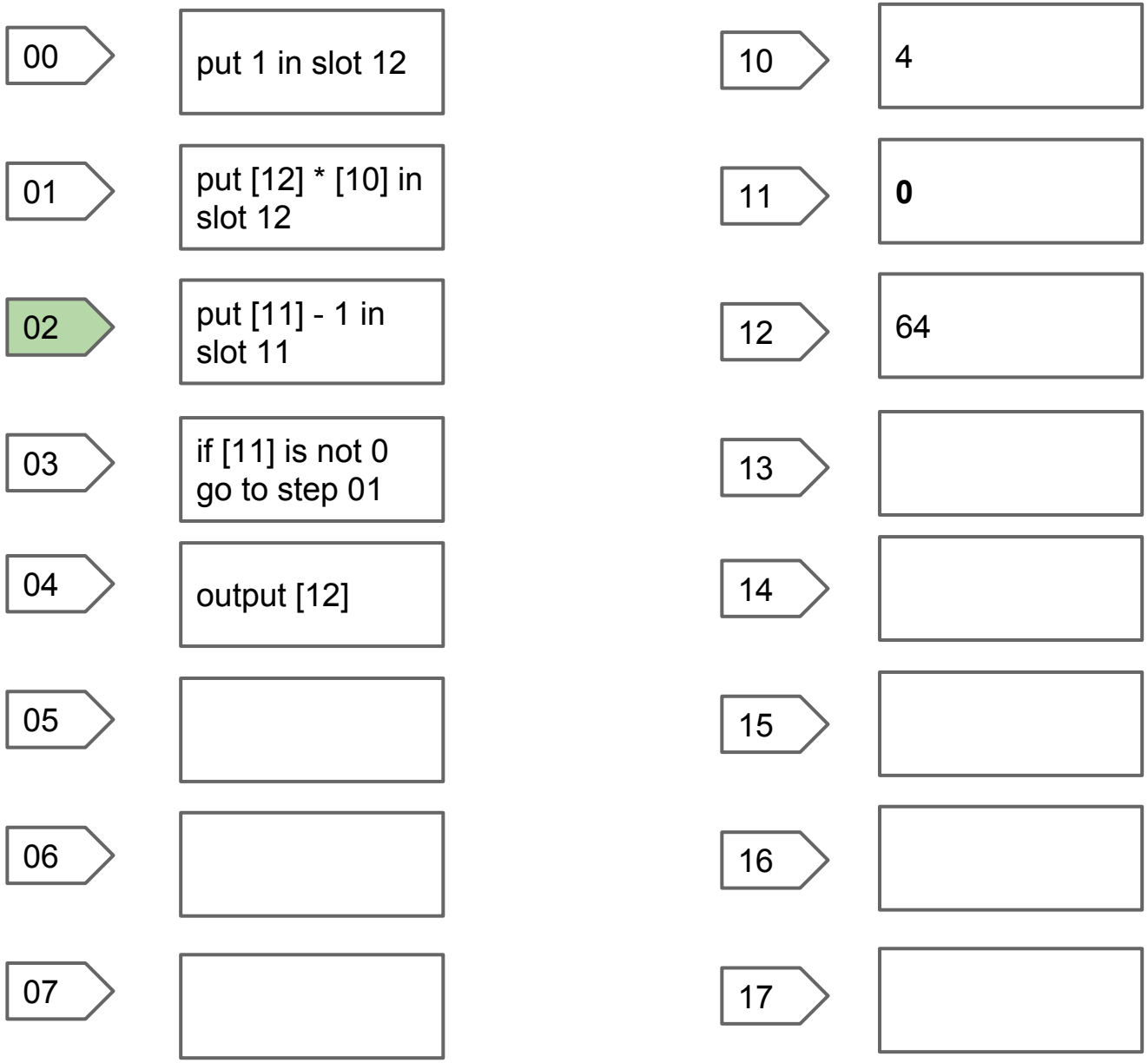

00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
2
12
4
13
14
15
16
17

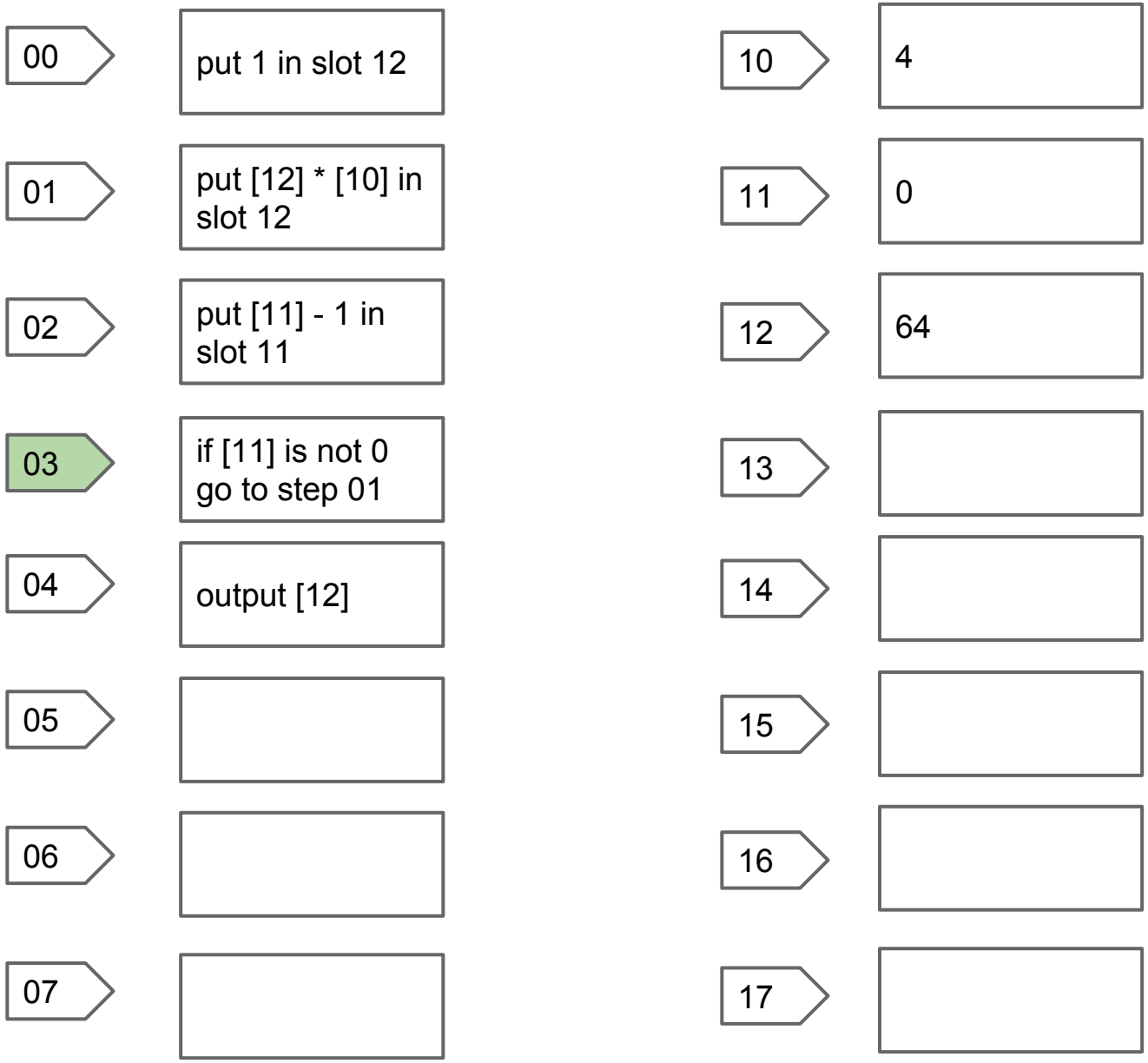

00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
2
12
16
13
14
15
16
17

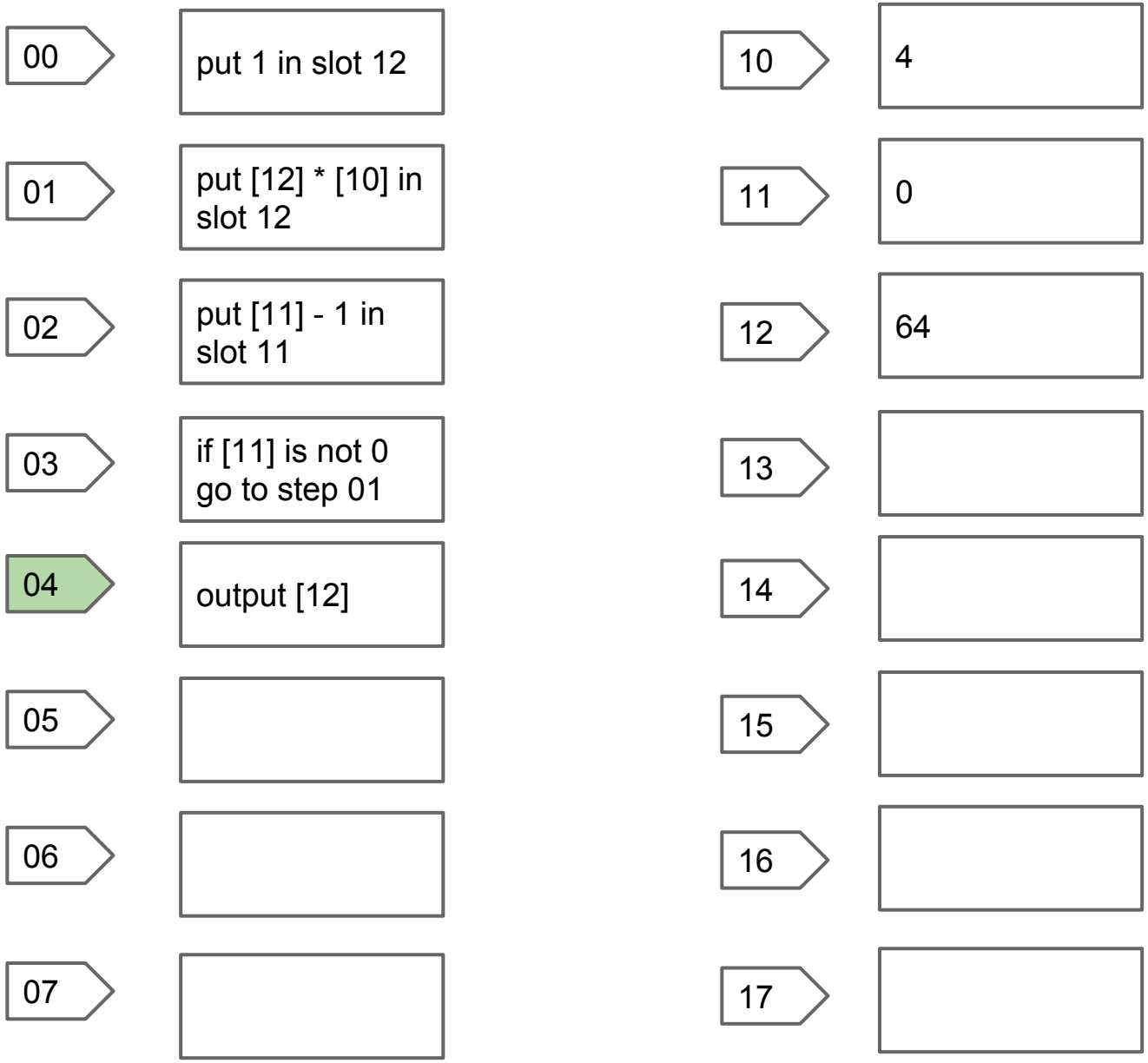

00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
1
12
16
13
14
15
16
17


00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
1
12
16
13
14
15
16
17


00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
1
12
64
13
14
15
16
17


00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
0
12
64
13
14
15
16
17


00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
0
12
64
13
14
15
16
17


00
put 1 in slot 12
01
put [12] * [10] in slot 12
02
put [11] - 1 in slot 11
03
if [11] is not 0 go to step 01
04
output [12]
05
06
07
10
4
11
0
12
64
13
14
15
16
17

64

# What did that do?

# Very simple instructions

**(we would be pretty frustrated)**

# Compiler

Is a program that takes complicated instructions, and **translates** them to simple instructions.

# Interpreter

Is a program that takes complicated instructions and **follows** them.

# Writing complicated instructions

- More complicated for the computer.  Less complicated for you.
- The rules for writing them is called a **programming language**
- We'll be using **Python**
- Specifically, Python is an interpreted language