



# Introduction to Information Systems

## - Understanding the digital world

**5** Algorithms, Programming and Programming  
languages

Liang Zhao

ILA, Doshisha University

12001102, Fall, 2023



# Today's schedule

- Review of Mini test #4 (5')
- Answers to the coding tasks (10')
- Mini test #5 (5')
- Algorithms & Programming (25')
- Programming languages (45')

# Last coding task 1

The next program calculates  $1 + 2 + \dots + N$  for a number **N given by the user.**

Toy Machine Simulator ->

<http://www.cs.princeton.edu/courses/archive/fall18/cos109/toysim.html>



- |                     |                     |
|---------------------|---------------------|
| 1. <b>_GET</b>      | 11. <b>E_LOAD S</b> |
| 2. <b>_STORE N</b>  | 12. <b>_PRINT</b>   |
| 3. <b>L_LOAD N</b>  | 13. <b>_STOP</b>    |
| 4. <b>_IFZERO E</b> | 14. <b>N</b>        |
| 5. <b>_ADD S</b>    | 15. <b>S 0</b>      |
| 6. <b>_STORE S</b>  |                     |
| 7. <b>_LOAD N</b>   |                     |
| 8. <b>_SUB 1</b>    |                     |
| 9. <b>_STORE N</b>  |                     |
| 10. <b>_GOTO L</b>  |                     |

## Last coding task 2

Write a program that prints 1, 2, ..., 99 **in that order** with loop – an elegant answer by Jacky.

Toy Machine Simulator ->

<http://www.cs.princeton.edu/courses/archive/fall18/cos109/toysim.html>



1. L\_load a
2. L\_add 1
3. L\_store a
4. L\_print
5. L\_sub c
6. L\_ifzero b
7. L\_goto L
8. a 0
9. b stop
10. c 99

# Algorithm: procedure to solve a problem/task

The efficiency of a program depends on the algorithm.

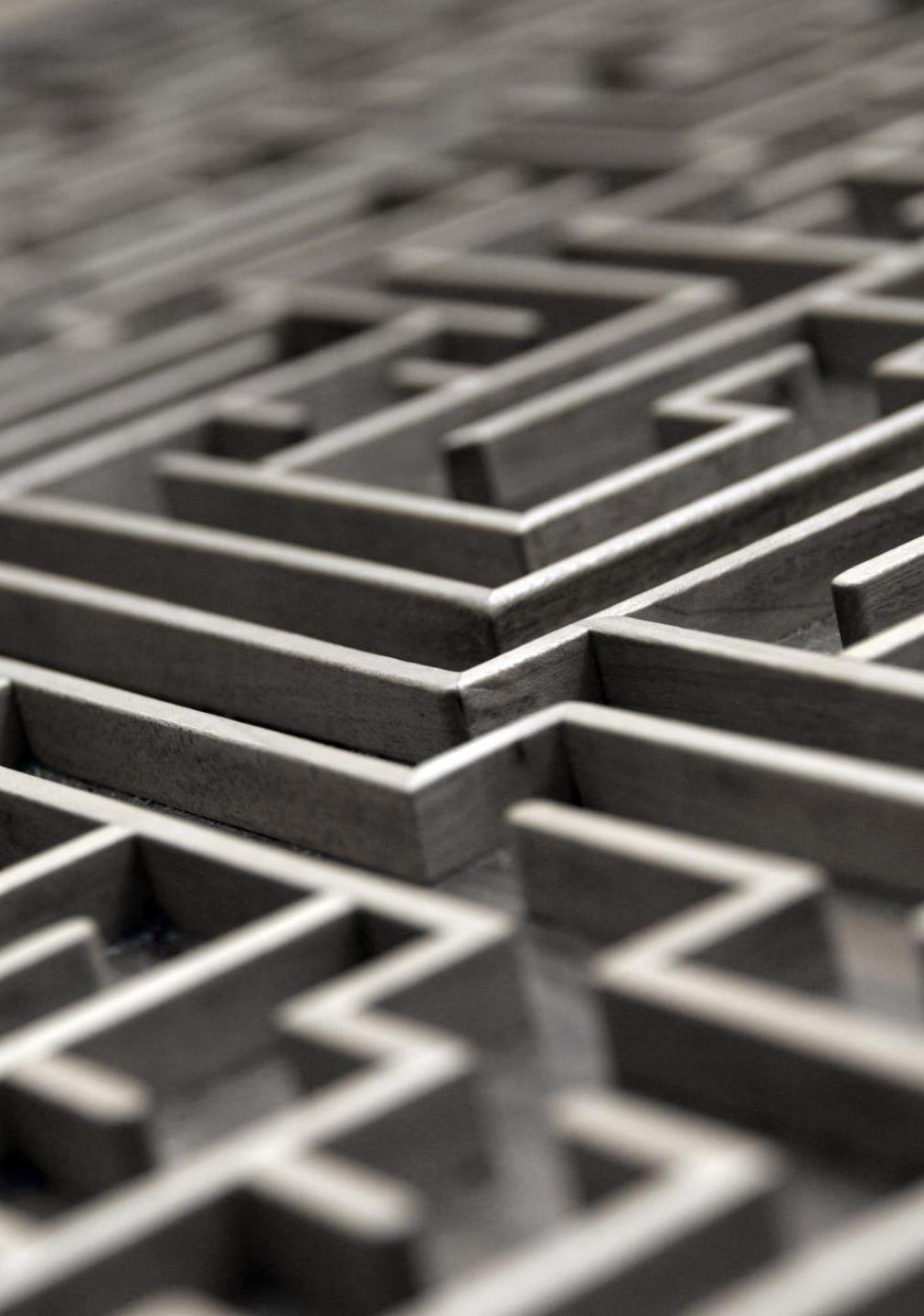
```
_GET  
_STORE A  
  
99 times {  
    _ADD A  
    ...  
    _ADD A  
    _PRINT  
    _STOP  
    A}
```

> 100 operations

```
1. _GET      9. _STORE S  
2. _STORE A  10. _GOTO L  
3. L_LOAD N 11. E_LOAD S  
4. _IFZERO E 12. _PRINT  
5. _SUB 1    13. _STOP  
6. _STORE N  14. A  
7. _LOAD S   15. S_0  
8. _ADD A    16. N_100
```

Only 17 operations

```
1. _GET      10. _ADD A  
2. _STORE A  11. _STORE A  
3. _ADD A    12. _ADD A  
4. _STORE A  13. _ADD A  
5. _ADD A    14. _ADD A  
6. _STORE A  15. _ADD A  
7. _ADD A    16. _PRINT  
8. _ADD A    17. _STOP  
9. _ADD A    18. A
```

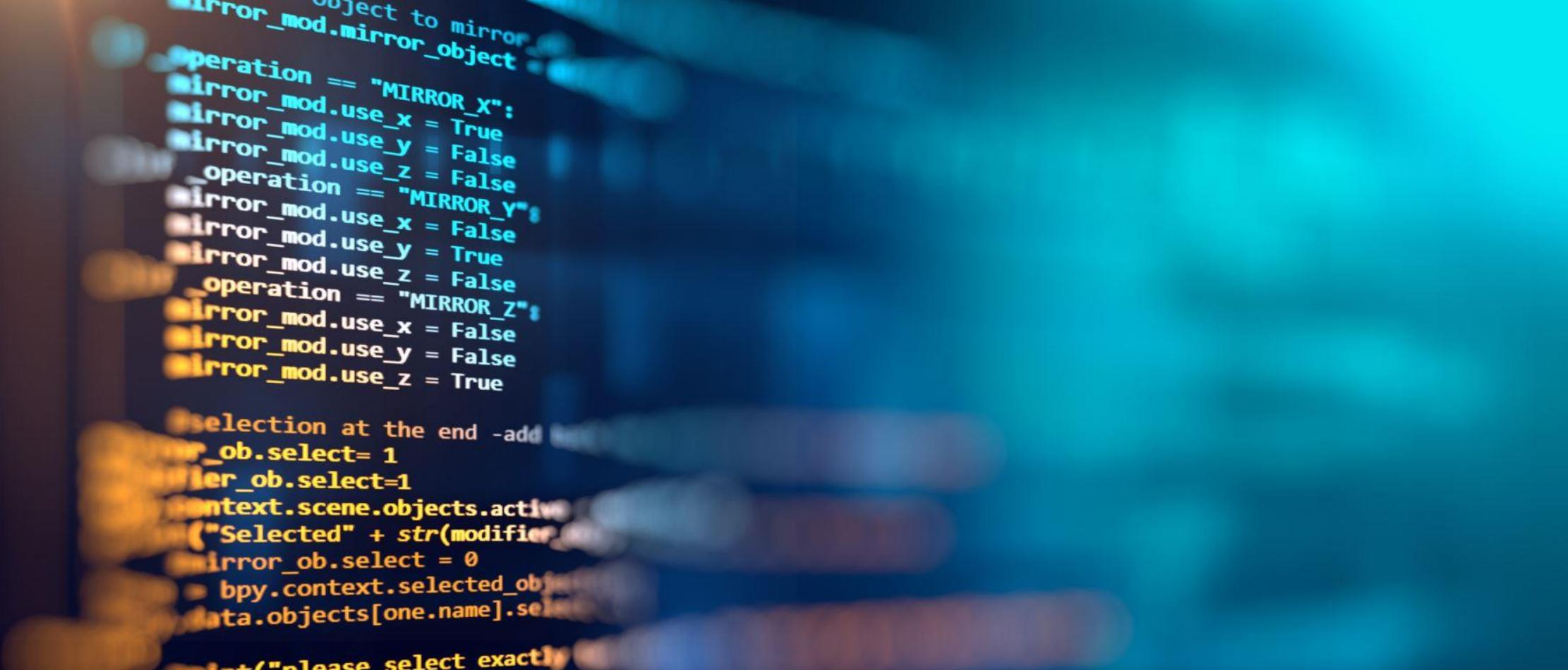


# Hard problems and complexity

- **Complexity:** How difficult an algorithm is. It is evaluated by the **#steps** and **#memory** used.
- <https://www.youtube.com/watch?v=Q4gTV4r0zRs>  
(See how hard a hard problem can be. It is in Japanese with English subtitles. About 8 minutes)

# Program

- A collection of instructions to **implement** an algorithm.
- There exist many dialects, called **programming languages**.
- The one we learned is **assembly**. There are > 200 languages.

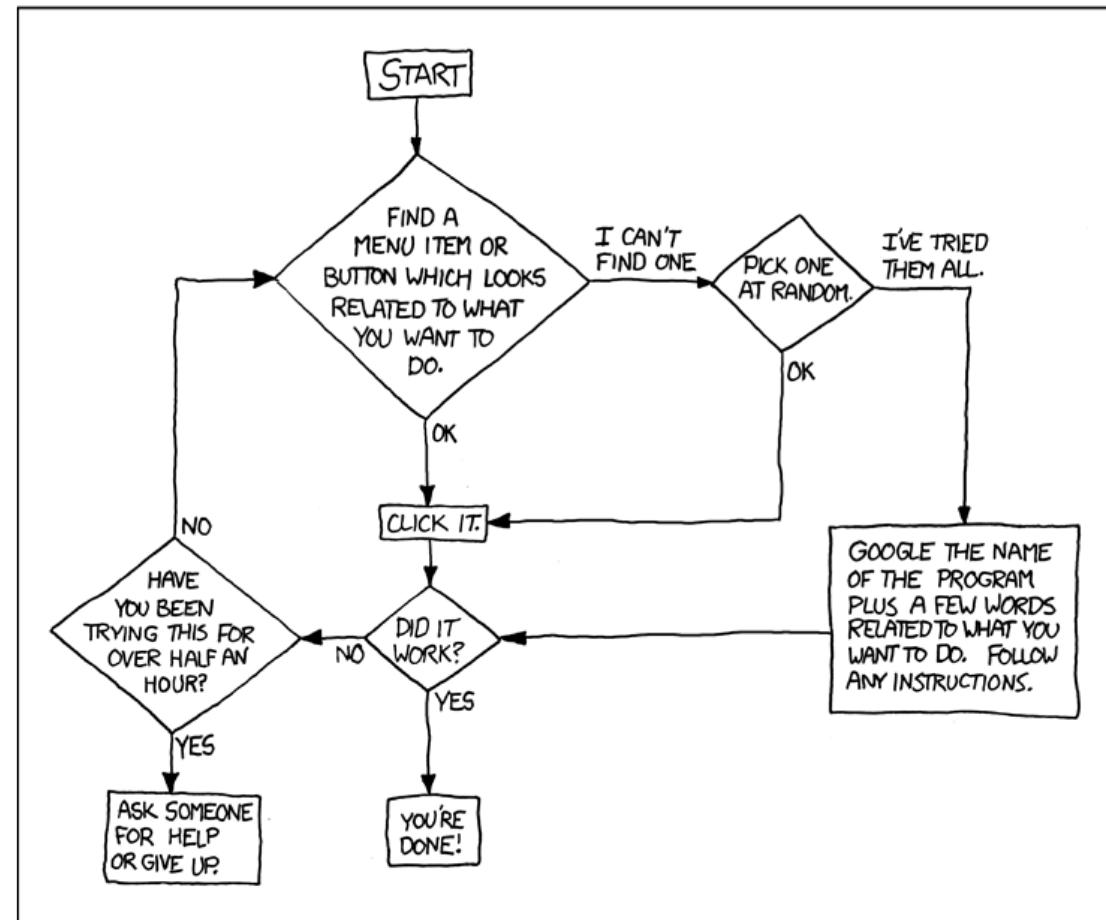


# Three fundamental structures of ALL programs (algorithms).

1. Sequential
2. Conditional
3. Iterative (-> loop)

DEAR VARIOUS PARENTS, GRANDPARENTS, CO-WORKERS, AND OTHER "NOT COMPUTER PEOPLE".

WE DON'T MAGICALLY KNOW HOW TO DO EVERYTHING IN EVERY PROGRAM. WHEN WE HELP YOU, WE'RE USUALLY JUST DOING THIS:



PLEASE PRINT THIS FLOWCHART OUT AND TAPE IT NEAR YOUR SCREEN.  
CONGRATULATIONS; YOU'RE NOW THE LOCAL COMPUTER EXPERT!

# Python: a high-level programming language



1. Write a program to calculate  $1 + 2 + \dots + N$  for a number  $N$  input by the user.

2. Write a program that prints 1, 2, ..., 99 in that order with loop.

<https://paiza.io/en/projects/new?language=python3>

## Program 1

```
1. n = int(input())
2. s = 0
3. for i in range(1, n+1):
4.     s = s + i
5. print(s)
```

Note: an indent = 4 spaces

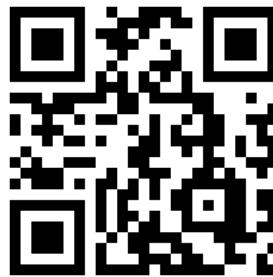
## Program 2

```
1. for i in range(100):
2.     print(i)
```

- Put the code into the code window (skip # lines).
- \* Put the input in the "Input" window.
- Click the "Run (Ctrl+Enter)" button to execute.

# Scratch (yet another programming language)

1. Go to <https://scratch.mit.edu>



2. "Start Creating" -> "Tutorials"



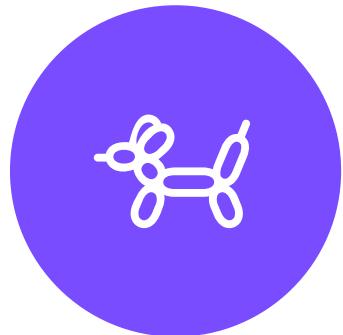
3. Open the next sample (click flag -> start, click cat -> stop)

<https://scratch.mit.edu/projects/750318208/>

Click "See inside" to investigate the code. Modify the program so that the cat runs faster, or slower, or says something different, or does something different, etc.

Notice how the three fundamental structures are implemented in Scratch.

# Homework



## WATCH VIDEOS

Watch the movies mentioned so far if you have not  
(You are not expected to understand everything)



## READ THE TEXTBOOK

**IF YOU HAVE NOT**