KANAZAWA UNIVERSITY | Emerging Media Initiative

GSCI1801A

Information Science

Lecture 2: Flowcharts, Data, and Algorithms (Part 1)

Asst. Prof. Chawanat NAKASAN | 2021-10-12

Agenda

1. Flowcharts & Flow Control
   - Branching (Decision; if/if-else)
   - Loops (Iteration; while/do-while/for)
2. Data & Data Types
   - Primitive Data Types
   - Data Encoding
   - Composite Data Types
3. Algorithms
   - Time Complexity of Algorithms
   - (Next Class) Types of Algorithms
- (Next Class) Problem Solving

Where are we in the CS curriculum?

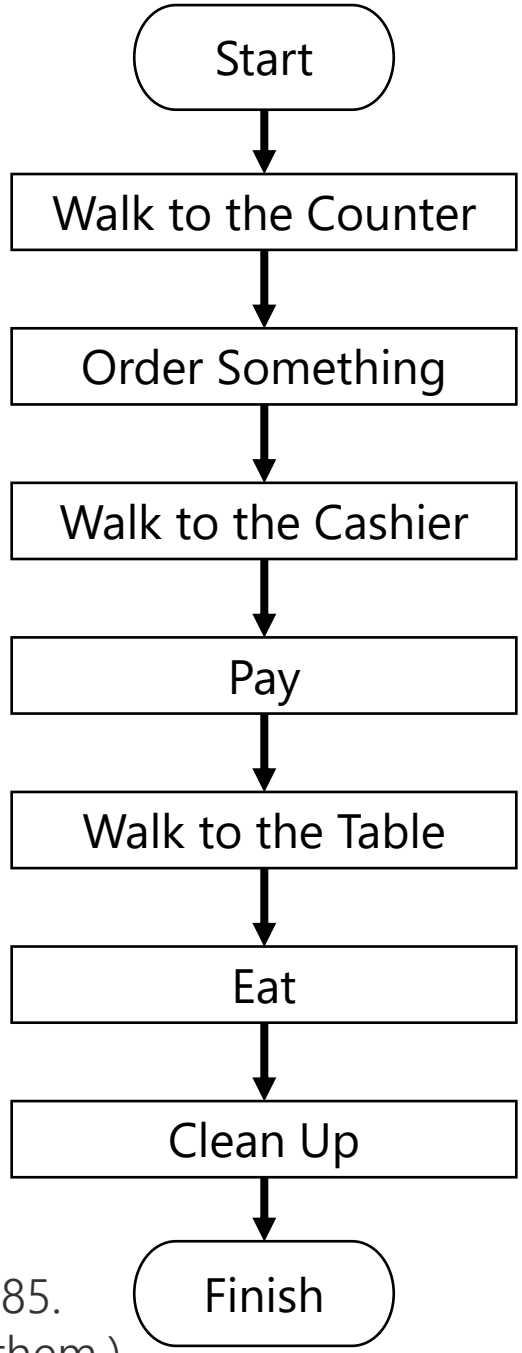
Year 1	Year 2	Year 3	Year 4
Programming	Architecture	Operating System	Security
GS classes	Algorithms	Data Communication	Ethics
General Data Science Lectures	Data Types	Networks	Advanced Networks

This is from CS curricula in general, not specific to Kanazawa University.

2.1.
Flowcharts &
Flow Control

Flowcharts?

- Show how something is done.
- A rounded rectangle ("capsule") represents the beginning and the end of the process.
- A (right-angle) rectangle represents individual steps of the process.



Full standard for flowcharts is written in ISO 5807:1985.
(ISO papers are expensive, so we usually don't buy them.)

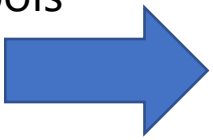
How to draw flowcharts?

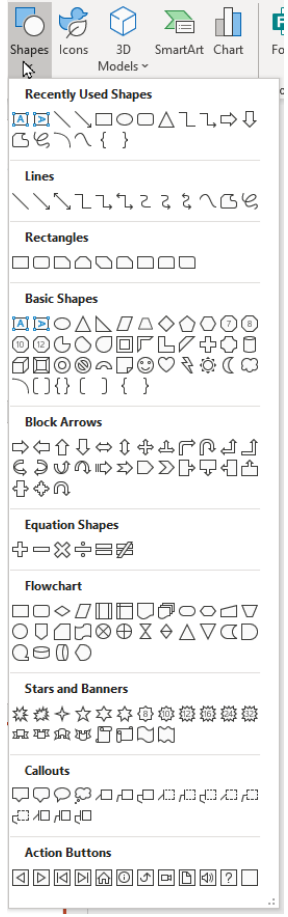
(This might be useful for your homework!)

- Paper: you can use rulers and plastic templates to help!
- General-purpose vector graphics software (InkScape, etc.)
- Dedicated diagramming software (draw.io, MS Visio, etc.)
- You can also draw using general-purpose office applications like MS Word and PowerPoint.

- MS PowerPoint:

Use these symbols



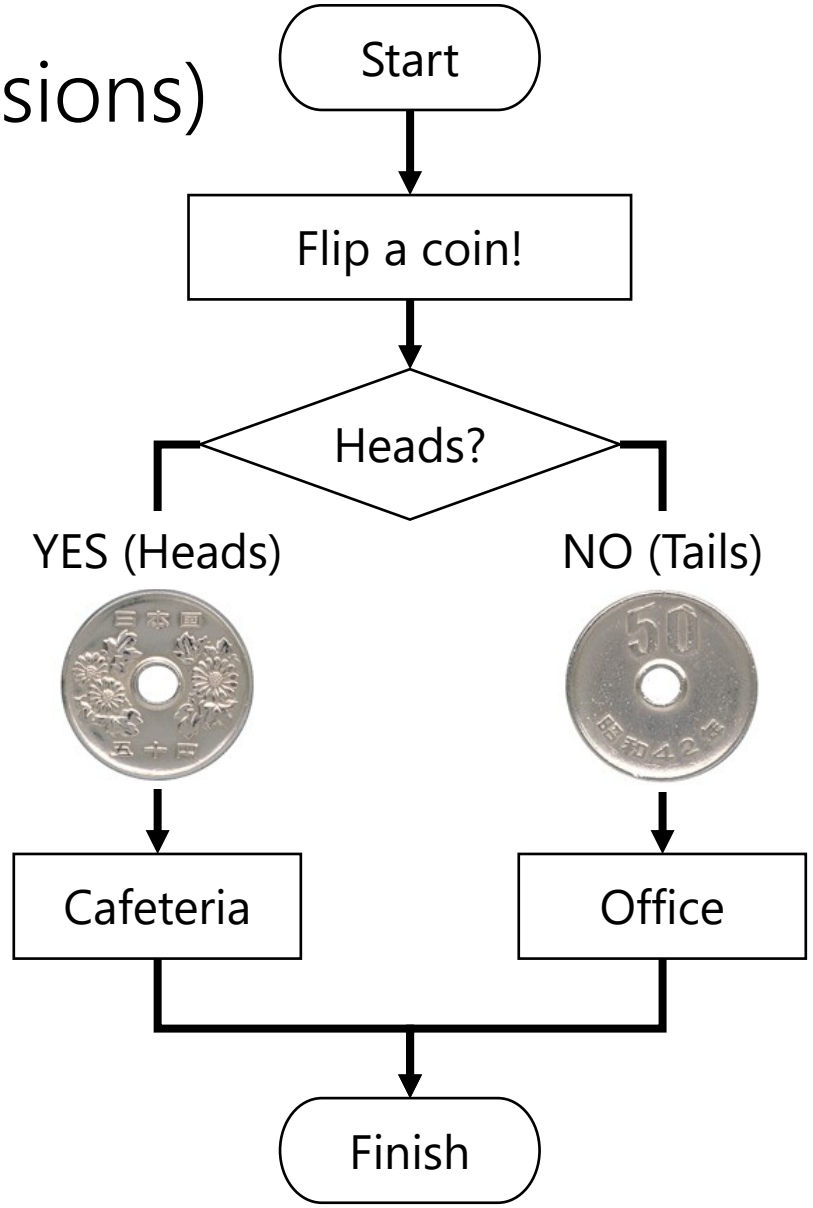


※ You can use any method for your report as long as it is submitted in PDF.

Branching (or Decisions)

- Where should I eat today?
- Use the rhombus (diamond) shape to indicate decisions.
- Decisions should only be YES or NO. Use more diamonds to create complex decision trees.

- Note 1: Vocabulary:
  - 表 = (omote) Obverse Side
  - 裏 = (ura) Reverse Side
- Note 2: A coin may also land on the edge, but it is usually ignored in probability discussions.

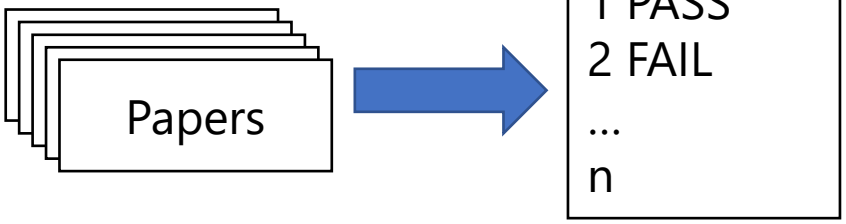


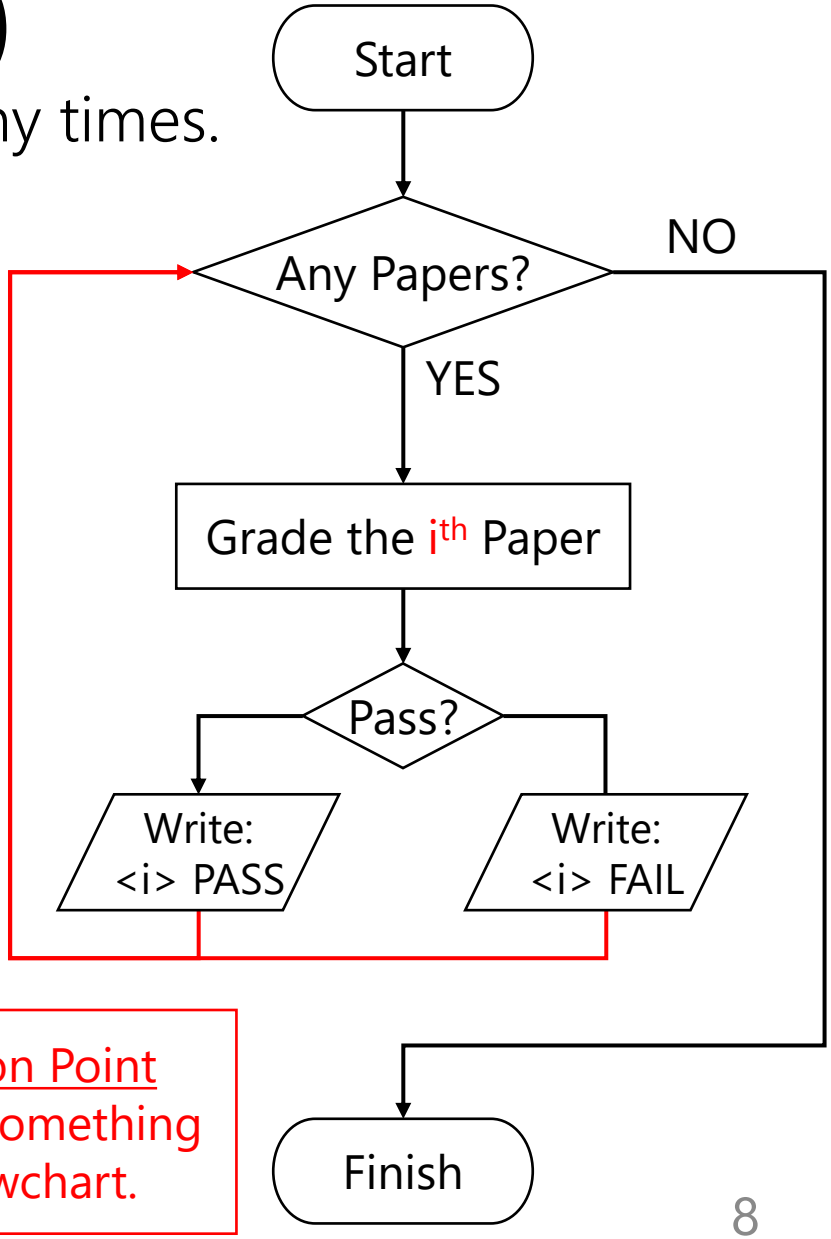
Coin image: Ministry of Finance

Loops (Iterations)

Repeating the same action many times.

- I need to grade papers, each labeled *i*, and record them as pass/fail.
  - In programming and maths, variable "i" is usually used to indicate index of things.
  - If you need more indices (indexes), use "j" and "k".



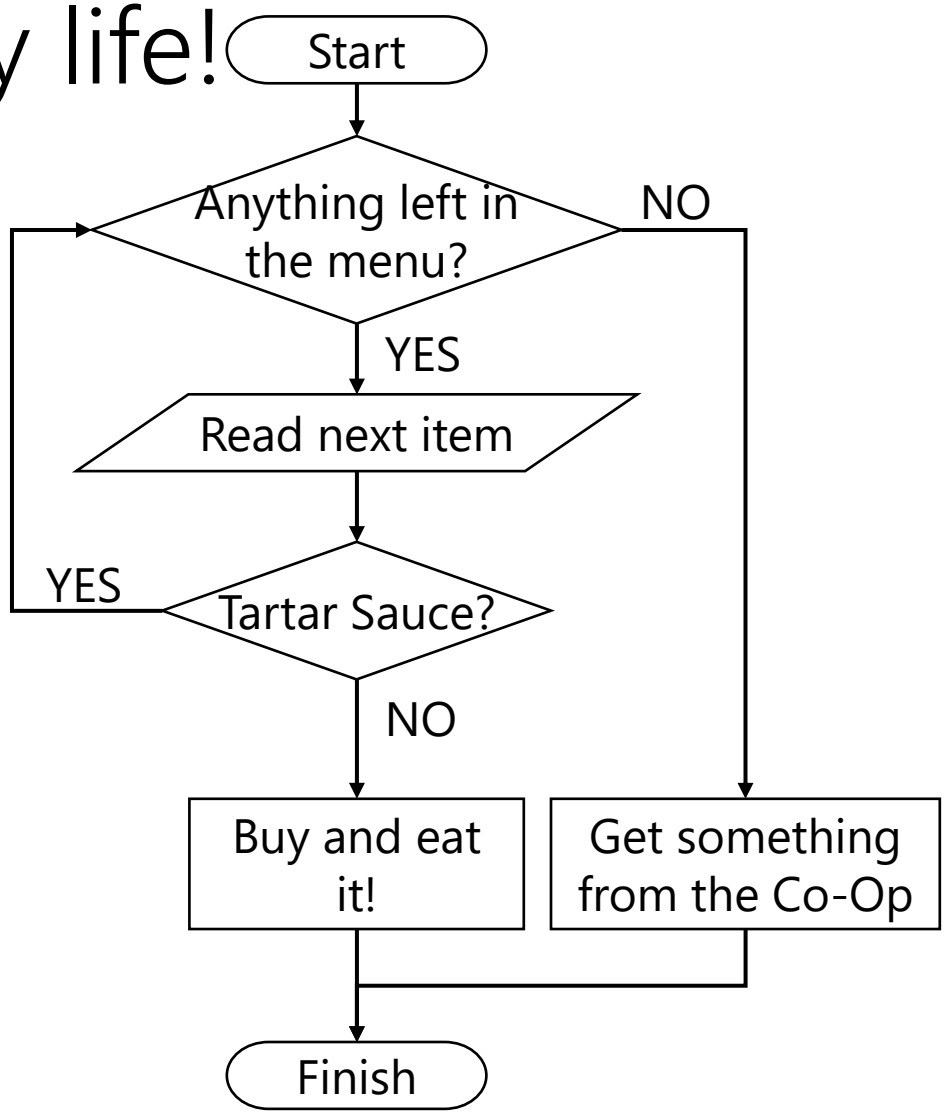


Use this symbol for input/output.

Participation Point
I'm missing something in this flowchart.

Flowcharts in daily life!

- **I hate tartar sauce.**
- I read the menu until I find one that does not include tartar sauce.
- What's your process for choosing what to eat?
- Is this accurate? What am I missing? (Hint: No, and it's not in the flowchart. It's a "business logic" problem.)



Loops within Loops

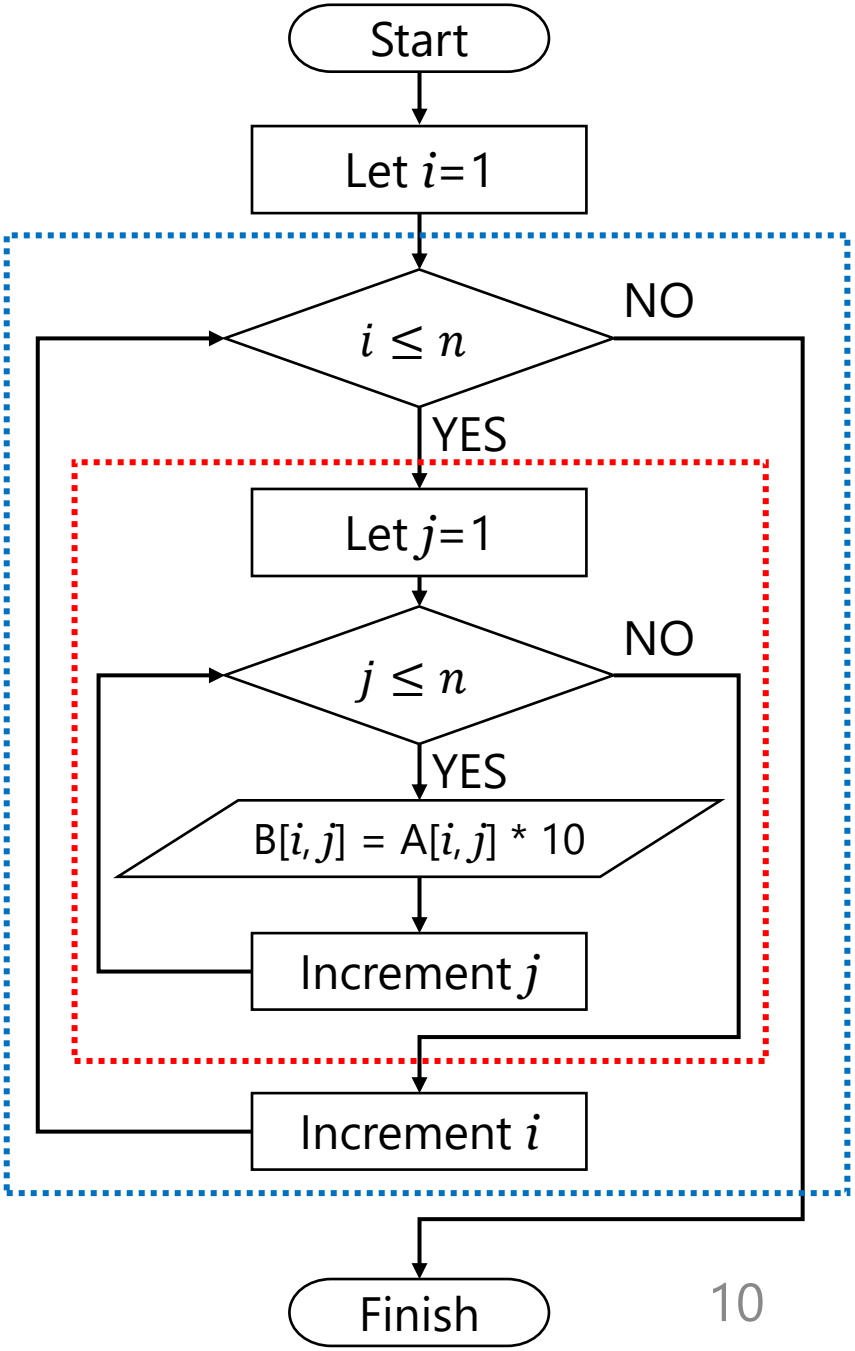
- Sometimes you have many levels of operations, such as iterating through a table.
- This flowchart shows how to copy and multiply values from Table A to Table B.

	1	2	...	n
1	5	3	...	2
2	7	5	...	2
	⋮	⋮	⋱	⋮
n	1	7	...	5

Table A

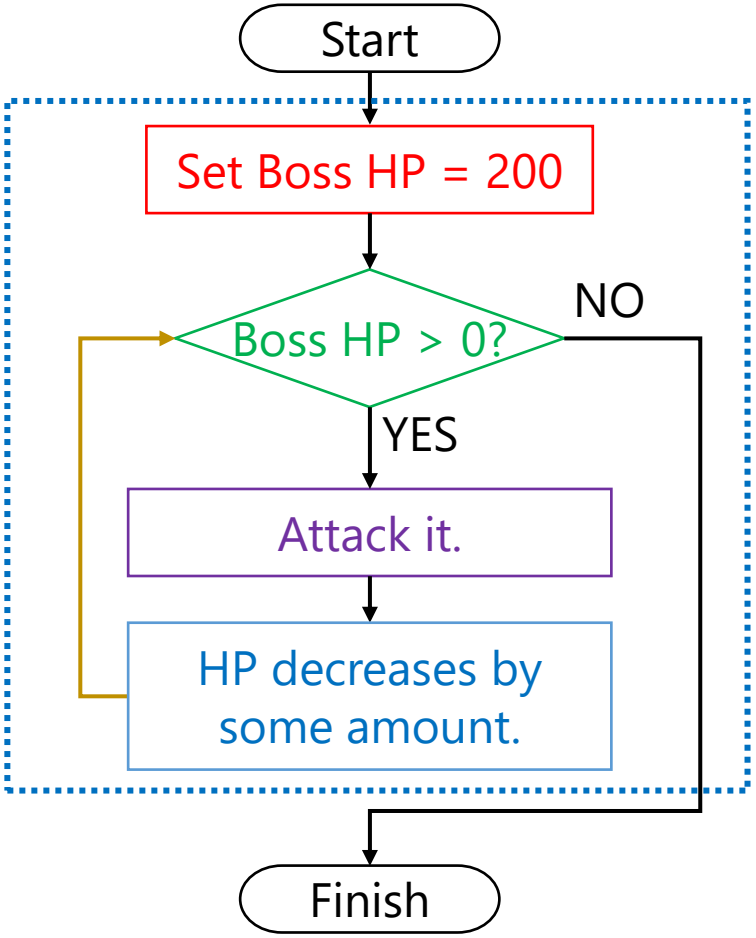
	1	2		n
1	50	30	...	20
2	70	50	...	20
	⋮	⋮	⋱	⋮
3	10	70	...	50

Table B



Essence of Loops

- A loop consists of four elements:
  - Initialization
  - Condition
  - Action
  - Modification
  - Repetition
- (Only Repetition is necessary to make a loop.)
- In programming, there are many ways to apply these elements.
- But, if you understand what loops are, you can write them in any language! ☺



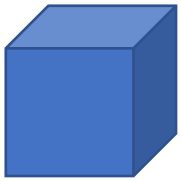
Summary of Flow Control

- Branching splits your procedure into two paths.
- Loops create repetition allowing you to do the same (or similar) thing many times.

- These two concepts are extremely important for algorithm studies, coming *right ahead!*

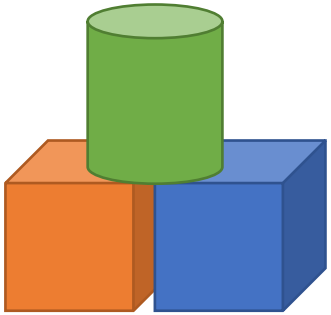
2. Data Types

Data Types



Primitive Data Type

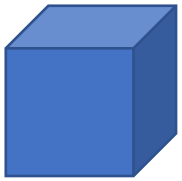
Usually just one thing, like a single number.



Composite Data Type

Consists of many primitive data elements together.

Data Types



Primitive Data Type

Usually just one thing, like a single number.

True

False

1

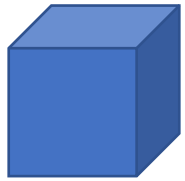
-200

'a'

3.141592

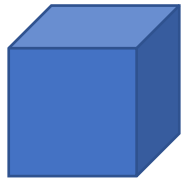
2021/10/02 13:15:00

Primitive Data Types



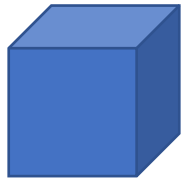
- Boolean Values
  - True or False
- Characters
- Numeric Data Types
  - Integers: already discussed
  - Real Numbers

Encoding gives meaning to data



- Computers understand only zeroes and ones.
- So, it is up to us to give data meaning.

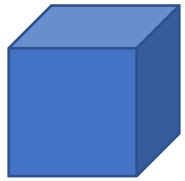
Using more bits, we can have a range of values.



0	0	0	0	0	0	0	0	=	0
1	1	1	1	1	1	1	1	=	255

0	1	0	0	0	0	0	1	=	65

We can also assign them specific meaning.

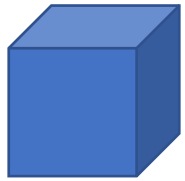


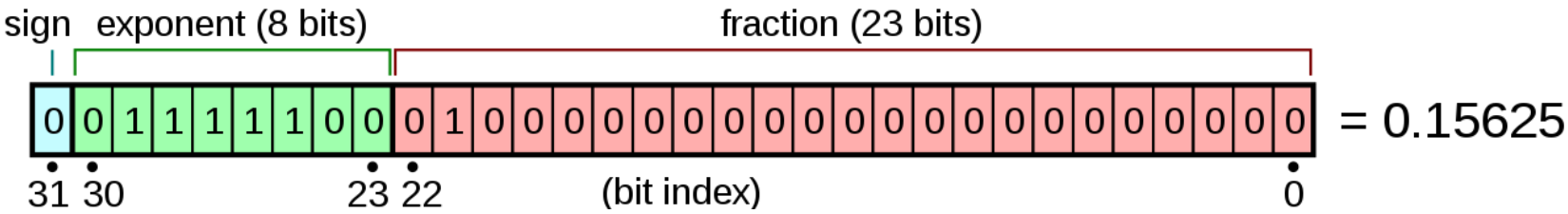
0	1	0	0	0	0	0	1	=	65

Binary	Octal	Decimal	Hexadecimal	Character
$100\ 0001_2$	101_8	65	41_{16}	A
$100\ 0010_2$	102_8	66	42_{16}	B
$100\ 0011_2$	103_8	67	43_{16}	C
$100\ 0100_2$	104_8	68	44_{16}	D
$100\ 0101_2$	105_8	69	45_{16}	E
$100\ 0110_2$	106_8	70	46_{16}	F

ASCII code table representation reproduced from https://en.wikipedia.org/wiki/ASCII

Real numbers are also made from bits and integers.





Picture of IEEE 754 representation of a number, https://en.wikipedia.org/wiki/IEEE_754
Graphics: Stannered / Wikipedia

Composite Data Types

- **Array**:
  - many of the same primitives, each of them are individual
- **Record**:
  - many of potentially different primitives, collectively forming a larger thing

- There are many other terms like lists, but today we'll discuss the most key terms.

Arrays:
many of the same primitives

65, 90, 65, 77, 73

can be translated to

'A', 'Z', 'A', 'M', 'I'

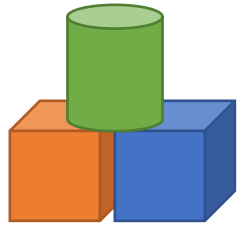
This is called an "array".

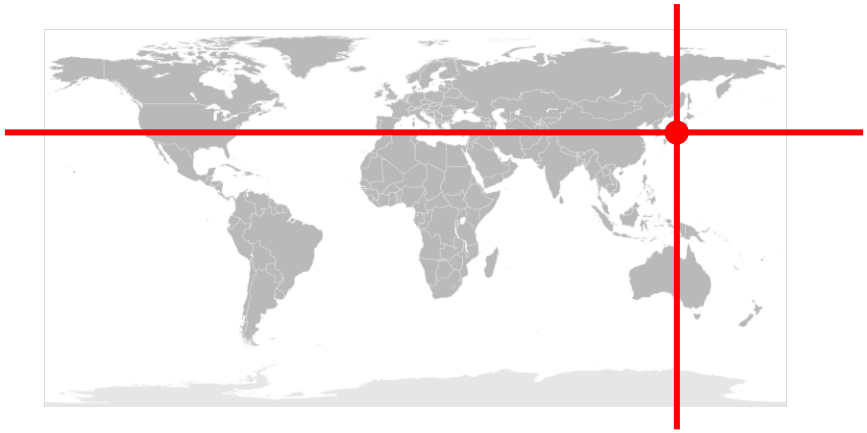
An array of characters is usually called a "string".

Dec	Char	Dec	Char
65	A	78	N
66	B	79	O
67	C	80	P
68	D	81	Q
69	E	82	R
70	F	83	S
71	G	84	T
72	H	85	U
73	I	86	V
74	J	87	W
75	K	88	X
76	L	89	Y
77	M	90	Z

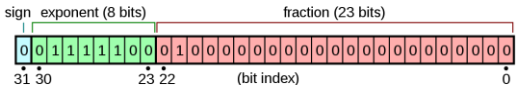
ASCII code table representation reproduced from https://en.wikipedia.org/wiki/ASCII

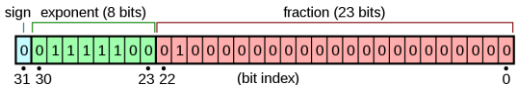
Lots of things in life can be translated to a set of values.





36.5495581,136.7096057





Coordinates can be represented using two real numbers.

Graphic Sources (excluding previously cited):
https://commons.wikimedia.org/wiki/File:BlankMap-World.svg; Google Maps
Image of floating-point representation is not accurate and used for presentation only.

Different types of data can be recorded together.

- Suppose we want to describe a food item at the cafeteria. This might be how it works:

```
{
    "name": "katsudon",
    "price": 400,
    "allergens": ["egg"],
    "restrictions": ["meat", "nonvegan", "pork"],
    "picture": "https://example.com/food/image.png"
}
```

A string

An integer

Arrays of strings

Another string, but this is a URL.

These braces { } group all the data into a single record.

This data format is called JSON (JavaScript Object Notation). You can learn more from https://www.json.org/

3. Algorithms

And analysis of time complexity

What is Algorithm?

*"a set of mathematical **instructions** or rules that, especially if **given to a computer**, will help to calculate an answer to a problem"*

– Cambridge Dictionary

*"a **procedure** for solving a mathematical problem (as of finding the greatest common divisor) in a **finite number of steps** that frequently involves repetition of an operation"*

– Merriam-Webster

*"a **list of rules** to follow in order to solve a problem"*

– BBC

Muhammad ibn Musa al-Khwarizmi, the mathematician namesake of "Algorithm".
Depicted on a Soviet stamp in 1983.

What is Algorithm?

procedures, instructions, or rules

finite number of steps

“given to a computer”

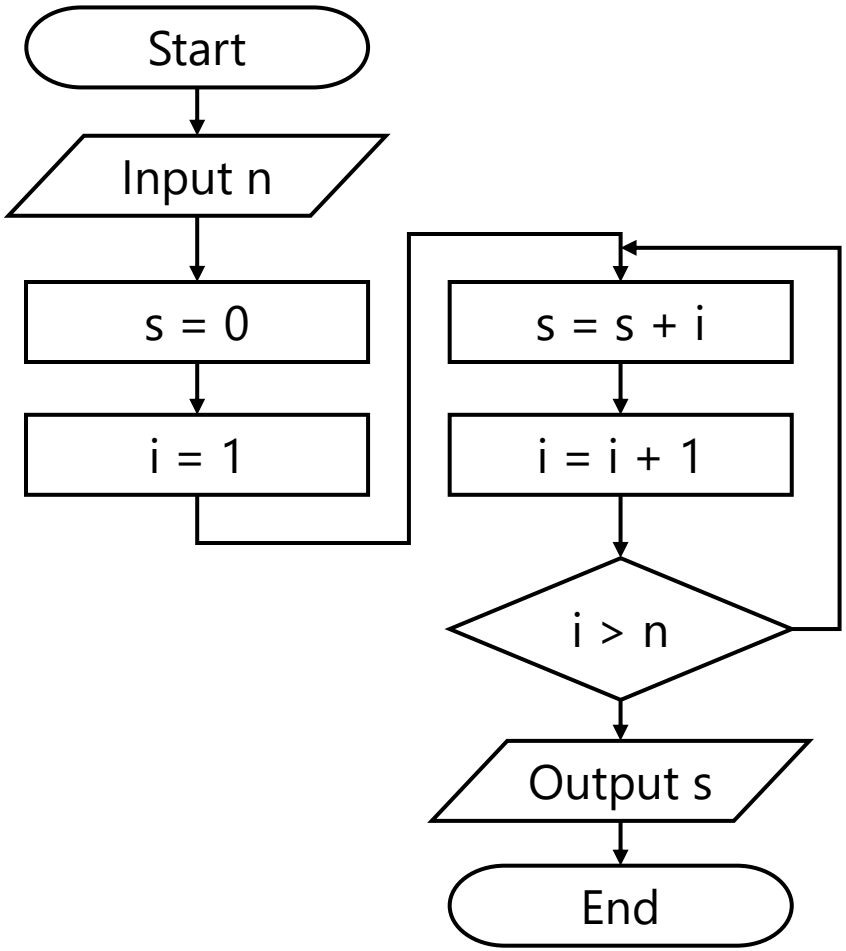
Important points:

- If you can draw it in a flowchart, it's already probably an algorithm.
- Algorithms can be expressed in many different ways.

- Time complexity analysis presented here will be very simple, intended for quick glimpses.

Suppose that we want to copy a string:

The textbook (Weng, 2021) example:

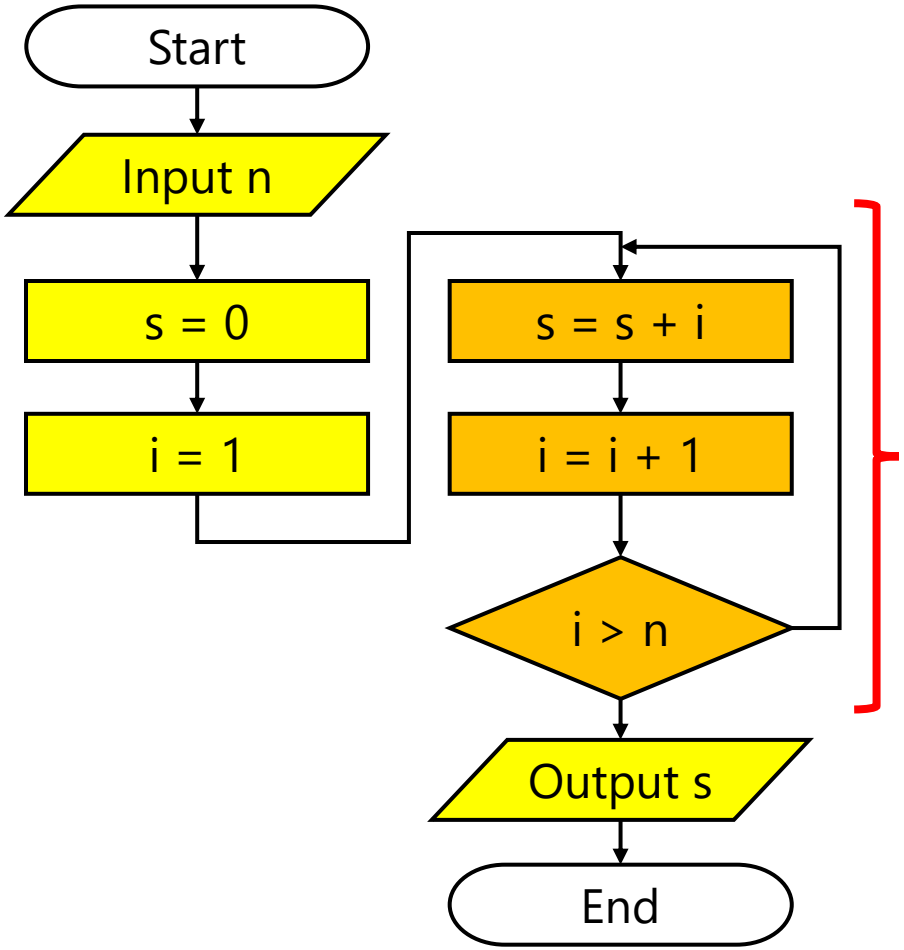


The very same algorithm written as Python code:

```
n = int(input().strip())
s = 0
i = 1
while i <= n:
    s = s + i
    i = i + 1
print(s)
```

Flowcharts and program codes have different purposes. These two represent the same algorithm.

This loop here can run many times.



We run the instructions (reach each box) **3n + 4** times in total.

This algorithm takes linearly increasing time to run compared to the size of n.

Describing Time Complexity

- Time complexity is popularly described using the “Big O Notation”.

- This is **basically** just the largest part of the time expression, without the multiplier in front.

- In our case, the previous algorithm is **O(n)**, because our time expression is **3n+4**.

O(n^2) example

- Instructions are run $n^2 + n + 1$ times.

	1	2	...	n
1	5	3	...	2
2	7	5	...	2
	⋮	⋮	⋱	⋮
n	1	7	...	5

Table A

	1	2		n
1	50	30	...	20
2	70	50	...	20
	⋮	⋮	⋱	⋮
3	10	70	...	50

Table B