Who am I

Prof. Wonjun Kim

Engineering Building C, 424-2

E-mail : wonjkim@konkuk.ac.kr

Research Interest :

Computer vision, image/video processing, pattern recognition, machine learning

Homepage : https://dcvl.konkuk.ac.kr/

(You can download lecture notes in my homepage)

Advanced Algorithms

(Lecture Note 1)

Introduction to Algorithm (1)

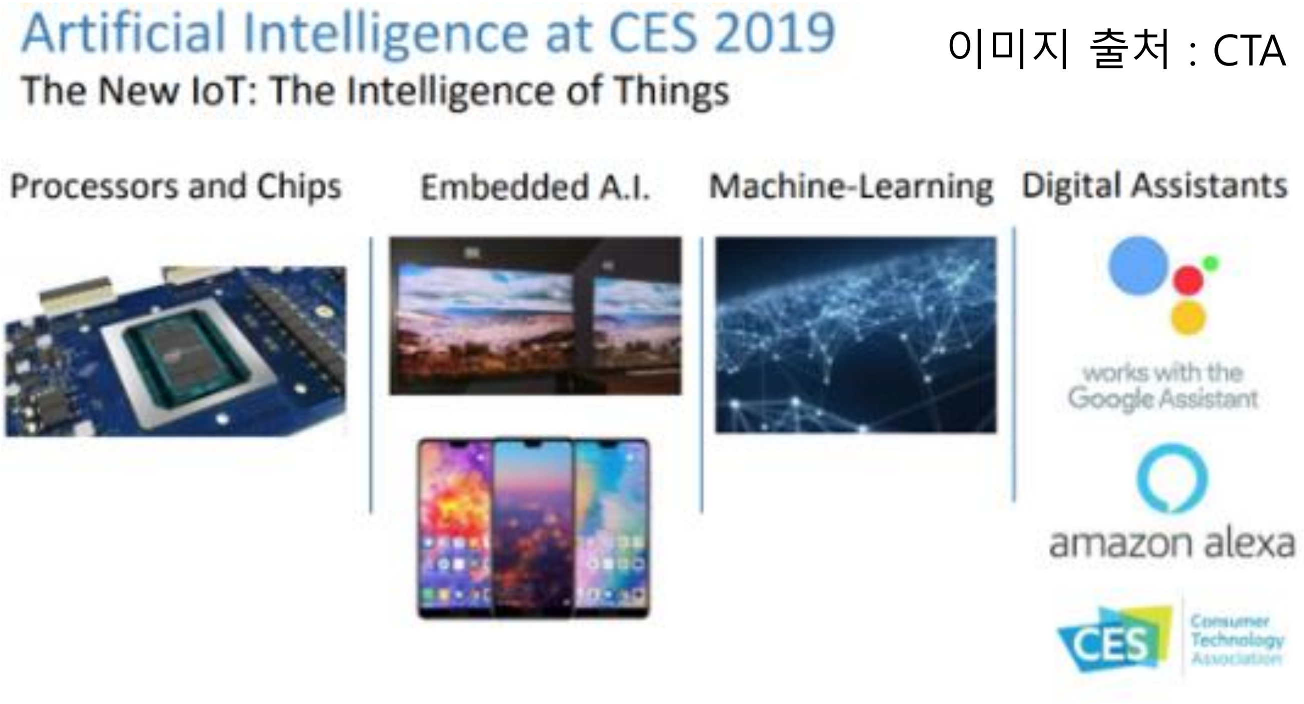
2019. 3. 5

Prof. Wonjun Kim

School of Electrical and Electronics Engineering

Before Starting … (1/2)

- **Algorithm is the heart of AI !**
  - **How can we make this robot work ?**



Artificial Intelligence at CES 2019

The New IoT: The Intelligence of Things

Processors and Chips

Embedded A.I.

Machine-Learning

Digital Assistants

works with the Google Assistant

amazon alexa

CES Consumer Technology Association

이미지 출처 : CTA

Hardware accelerator for AI

Smartphone

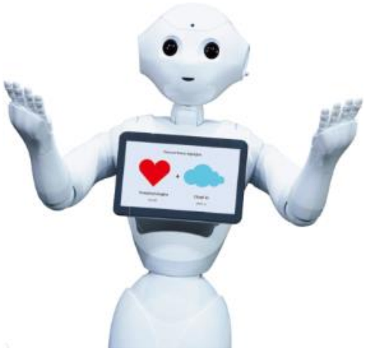
Algorithm

New platform

Before Starting ... (2/2)

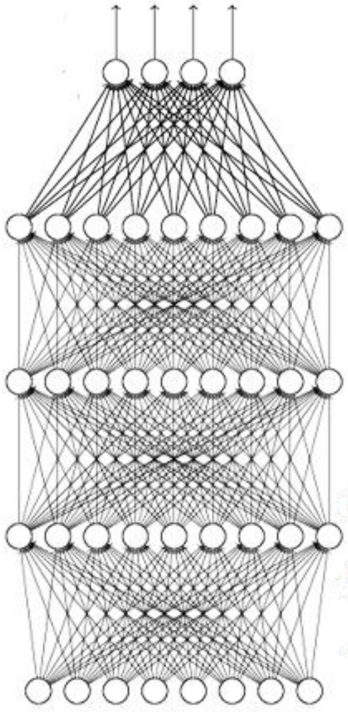
- **Algorithm is the heart of AI and computer vision !**

Echo (Amazon)



Pepper (Softbank)

Target output



Deep learning structure

What makes it possible ?

"Algorithm"

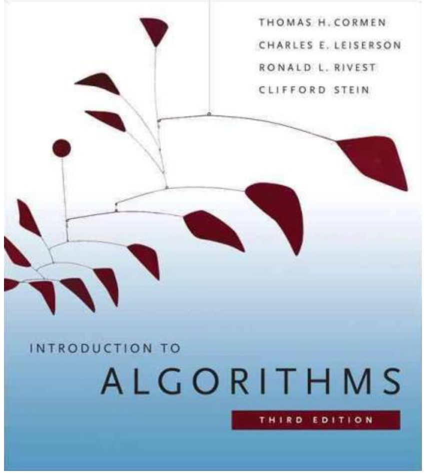
Class Schedule

What we learn in this class …

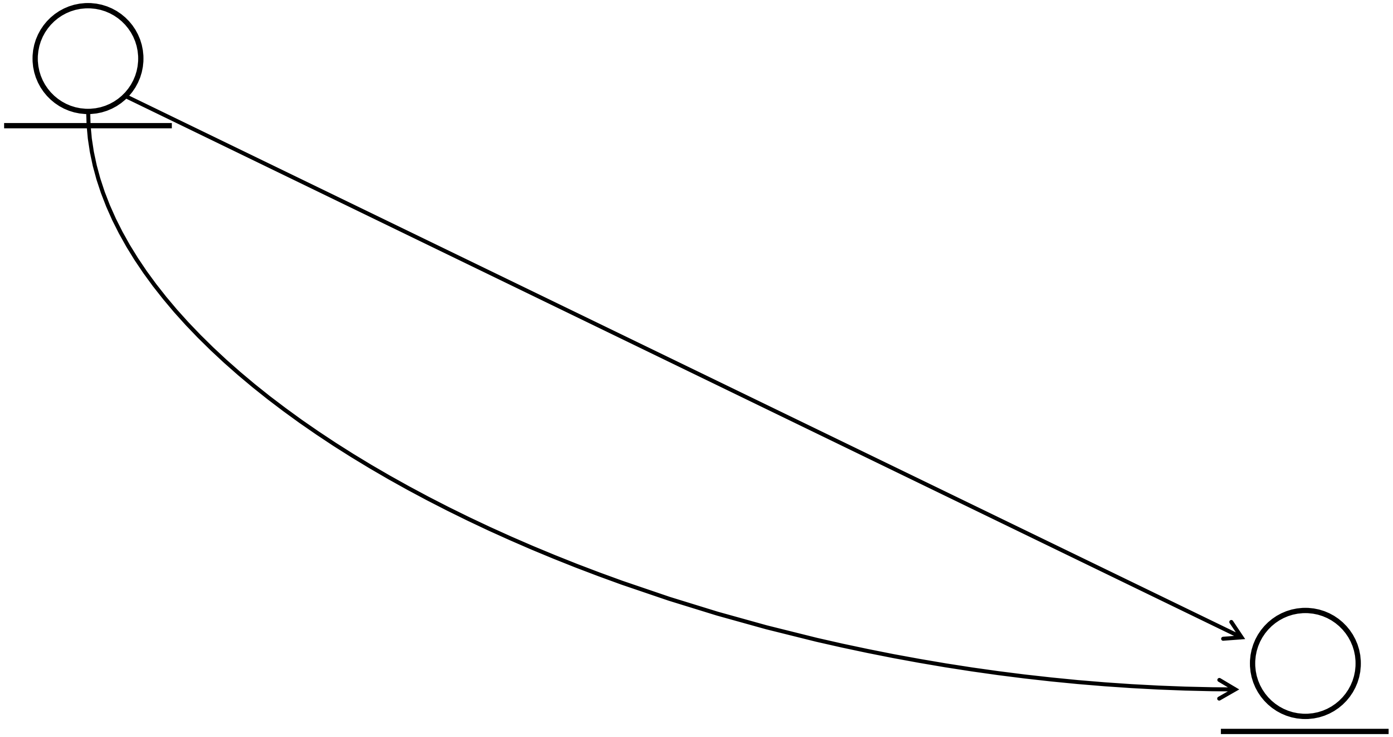
- Introduction to algorithms
- Warming-up : maze algorithm
- Recursion
- Recursion-based algorithms
- Graph theory
- Weighted graph
- Network theory
- Dynamic programming (if possible)
- Practical examples

※ Language : C or (C++)

Lecture Note



Introduction (1/6)



Introduction (2/6)

▪ What is the Algorithm ?

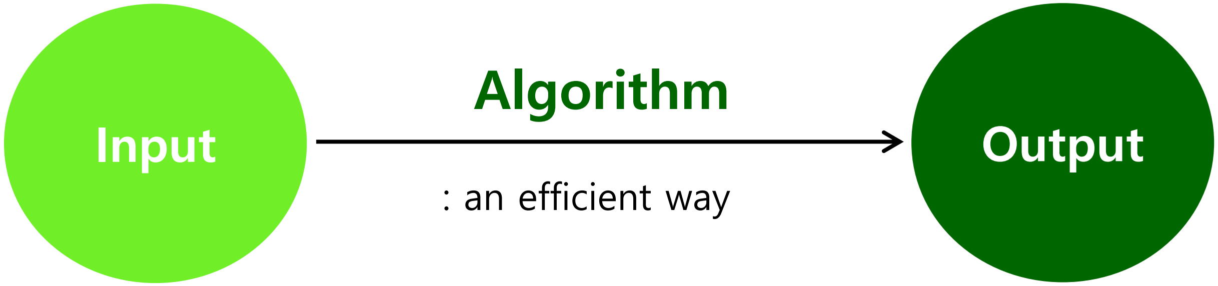
An algorithm is any well-defined computational procedure that takes some values (or a set of values) as input and produces some values (or a set of values) as output

- **Simple example**

  - Task (problem) : compute the average of mid term scores

  - Input : mid-term scores

  - Output : average score of all the scores ↓ **How to compute ?**



Input

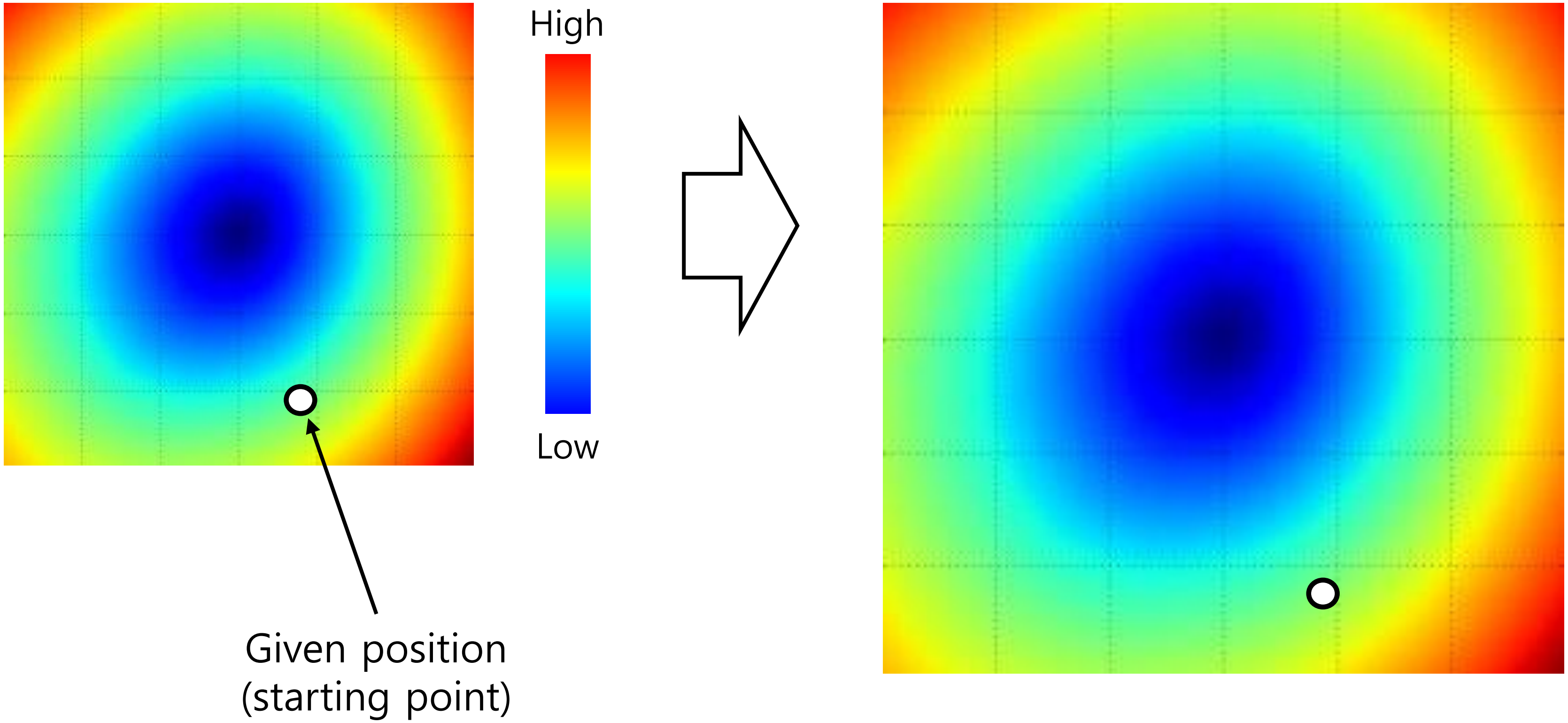
Algorithm

: an efficient way

Output

Introduction (3/6)

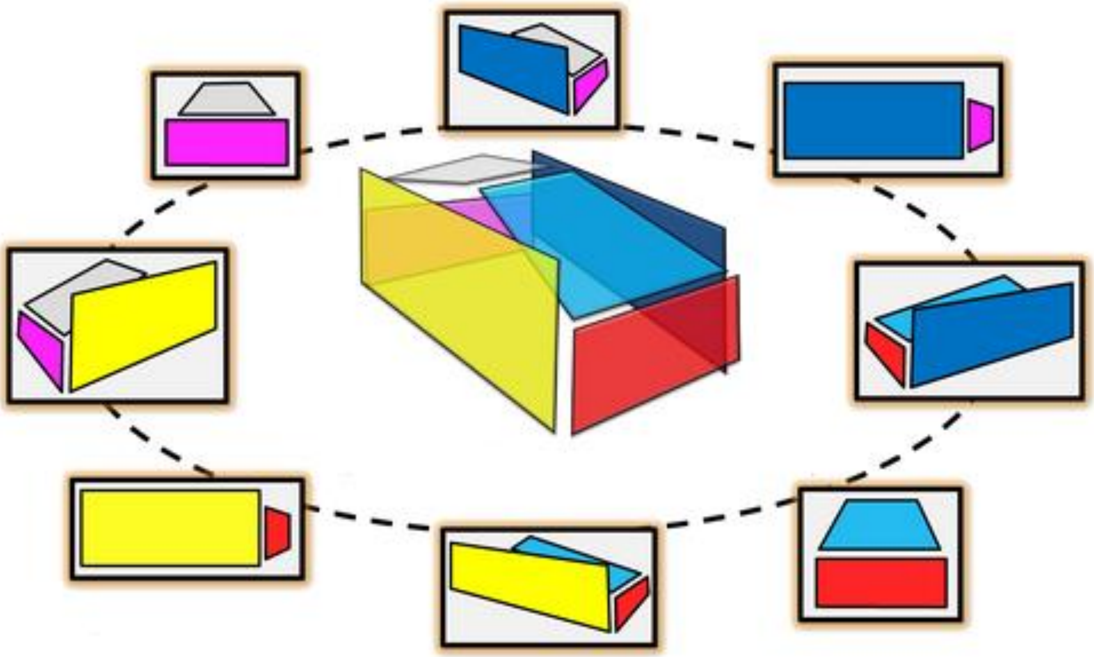
- **Needs for “desirable” algorithms**
  - **Goal : to find the lowest position on a given 2D space**



Draw your algorithm !

Introduction (4/6)

- Needs for "desirable" algorithms – cont'd

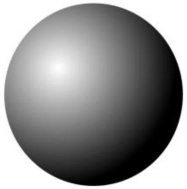


Multiview 3D reconstruction

* Image is from Google Image

* Why is infinite-dimensional ?

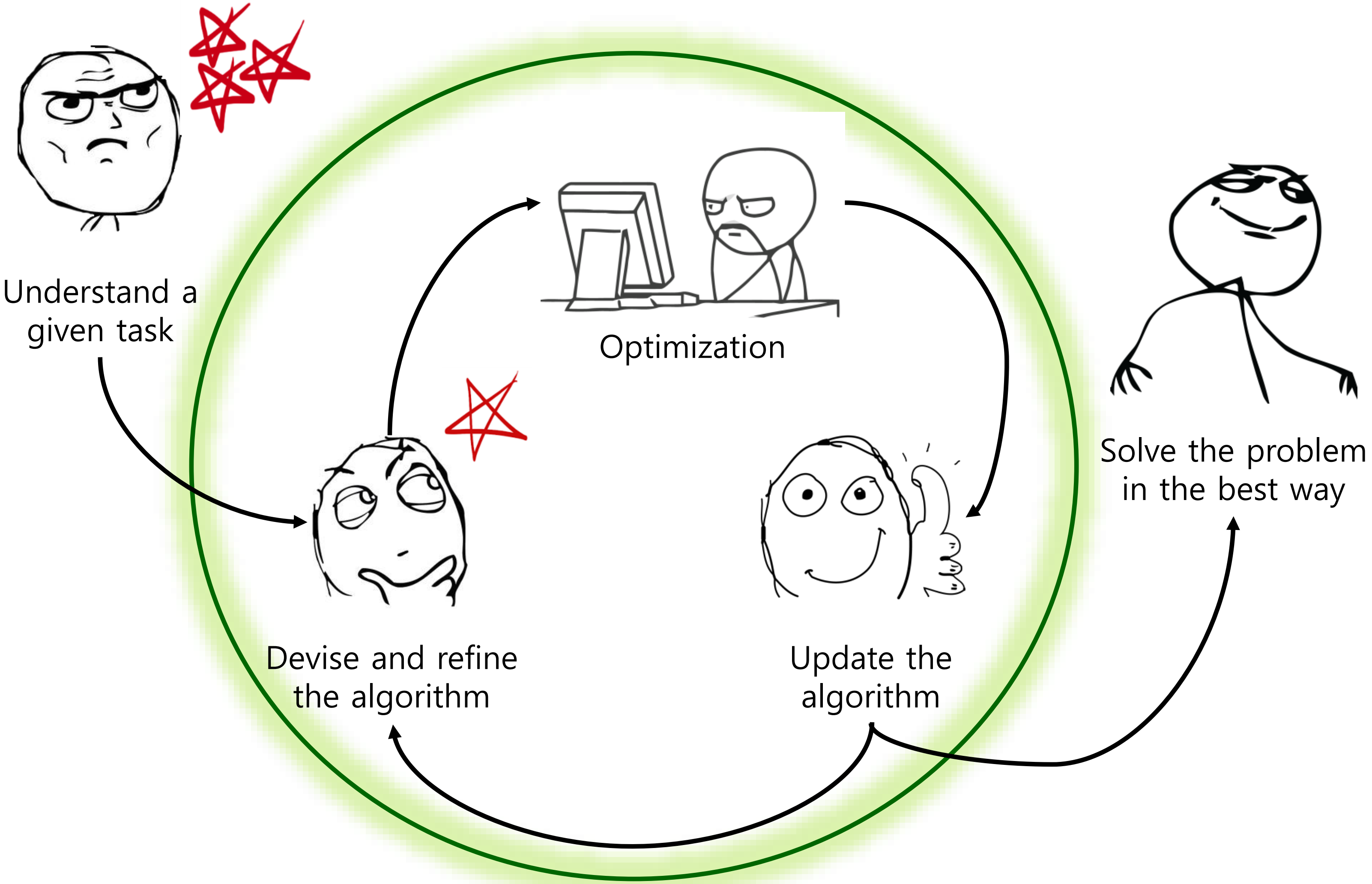
(Think about a sphere !)



Role of the algorithm ?

Introduction (5/6)

- **Procedure to make the algorithm**



Understand a given task

Optimization

Devise and refine the algorithm

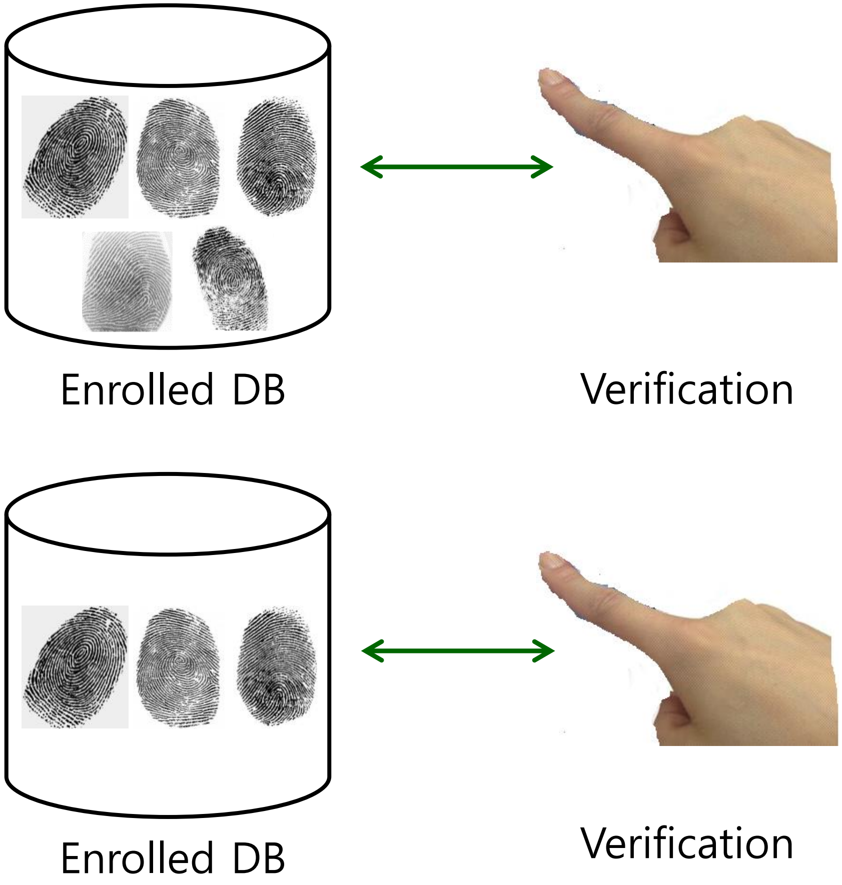
Update the algorithm

Solve the problem in the best way

Introduction (6/6)

- For optimization, we always think about ( ) !
  - Trade-off between the processing time and the memory

[Fingerprint recognition (S10)]



Enrolled DB

Verification

Enrolled DB

Verification

* Images are from Google Image

Warming-up Example (1/5)

Integer multiplication

Normal operation

(1) Multiplication twice

(2) Summing up all the values

```
   45
 x 37
 ----
  315
 135
 ----
 1665
```

$= 45 \times (30+7)$

$= (45 \times 30) + (45 \times 7)$

$= 1350 + 315$

$= 1665$

※ Complexity :

Multiplication > addition

a la russe algorithm

(1) If A is odd, copy B to C

(2) A=A/**2** (integer division), B=Bx**2**

(3) Repeat (1)(2) until A becomes 1,
summing up all C values

A	B	C
45	37	37
22	74	-
11	148	148
5	296	296
2	592	-
1	1184	1184

➔ 37+148+296+1184 = 1665

Warming-up Example (2/5)

- Integer multiplication (implementation)
  - Compare the processing time of two cases

Make your C codes !

Warming-up Example (3/5)

▪ Appendix : time checker

- Compute the processing time as follows :

```
#include <Windows.h>

LARGE_INTEGER freq, start, stop;
double diff;

QueryPerformanceFrequency(&freq); // computer frequency
QueryPerformanceCounter(&start); // starting point

Algorithm

QueryPerformanceCounter(&stop); // stopping point
diff = (double)(stop.QuadPart - start.QuadPart)/ freq.QuadPart;
```

Measuring time

Warming-up Example (4/5)

- **Analysis of previous two approaches for multiplication**
  - **Which algorithm is easy to understand for human ?**
  - **Is it same for the computer ?**

➡ A la russe is more efficient for the computer, but not optimal

(x2 or /2) operations can be regarded as **bit shift** in the computer

- temp[0]>>1 : /2 operation
- temp[0]<<1 : x2 operation

➡ Even though a la russe requires more lines,
computers can handle this kind of simple operations very fast

※ Implement it using a bit shift and compare the processing time

Warming-up Example (5/5)

- **How about multiplication for large numbers ?**
  - **A la russe algorithm requires quite a lot lines !**
  - **Karatsuba's algorithm**

Given two numbers x and y :

$$x = x_1 B^m + x_0, \quad y = y_1 B^m + y_0$$

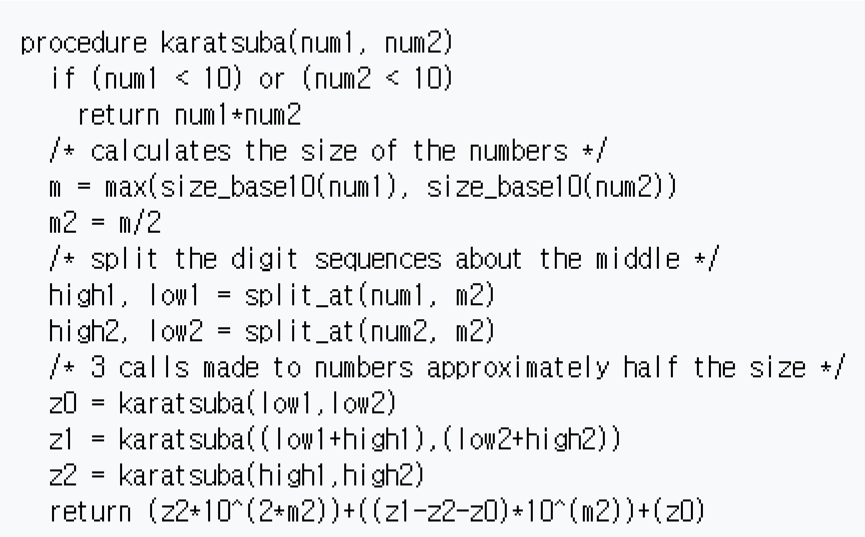
$$xy = (x_1 B^m + x_0)(y_1 B^m + y_0)$$
$$= z_2 B^{2m} + z_1 B^m + z_0$$

$$z_0 = x_0 y_0$$
$$z_1 = x_1 y_0 + x_0 y_1$$
$$z_2 = x_1 y_1$$

Why ?

$$z_1 = (x_1 + x_0)(y_1 + y_0) - z_2 - z_0$$

[Pseudo code for Karatsuba's alg.]



```
procedure karatsuba(num1, num2)
  if (num1 < 10) or (num2 < 10)
    return num1*num2
  /* calculates the size of the numbers */
  m = max(size_base10(num1), size_base10(num2))
  m2 = m/2
  /* split the digit sequences about the middle */
  high1, low1 = split_at(num1, m2)
  high2, low2 = split_at(num2, m2)
  /* 3 calls made to numbers approximately half the size */
  z0 = karatsuba(low1,low2)
  z1 = karatsuba((low1+high1),(low2+high2))
  z2 = karatsuba(high1,high2)
  return (z2*10^(2*m2))+((z1-z2-z0)*10^(m2))+(z0)
```

For example, 12345 x 6789 = ?

Try to implement it !

Summary

- Algorithm : find the best way to the solution
  - Need to understand a given problem clearly !
- Enjoy thinking with your programming