flowers for you !! Wk 1 - Lec

10%. weekly quizzes (wk 2)
30%. assignments (2 coding, 3 writing)
60%. final exam (in campus)
* 노트 정리를 잘하고 과제를 잘하자!

Lecture - ask Qs thru mentimeter
Solve tut sheet before tutorials ∨

Lec 1

Algorithm Analysis

3 Abstractions
- Computational problem
  - defines the task ☆
    - what input is & what output should be) provided.
- Algorithm
  - step-by-step recipe to ge from input to output) my job
- Correctness & complexity Analysis
  - proof that the algorithm solves the problem
  - analytical bound on resources used

Pseudocode : will resemble skeletal python code.

ex)
1. Comp problm: return max of array
2. algorithm:

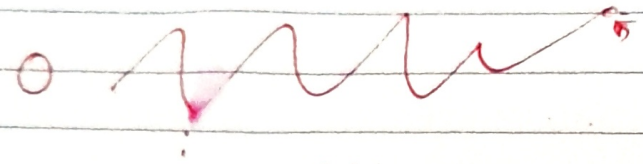
```
max ← -∞
for i ← 0 to n-1 do
    if A[i] > max:
        max ← A[i]
return max
```

3. correctness:
invariant:
After k^{th} iteration, max stores the maximum of the first k elements
build upon this invariant to prove the algorithm.

*** high-level English description is Mandatory!
~~(& pseudocode)~~.
pseudo → concrete description!

ex computational problem #2

- daily fluctuations of a stock price in an Array A.
- find ~~best~~ single-trade outcome.



$0 \le i \le j < n$

buy sell

$A = [3, -2, -5, -8, 3, 9, 15, -3, 9]$

$B = [0, 3, 1, -4, -12, -9, 0, 15, 12, 21]$

$A = [A_0, A_1, A_2, A_3 \dots A_{n-1}]$

$$B = \begin{cases} B[0] = 0 & \text{if } i = 0 \\ B[i] = A[i-1] + B[i-1] & \text{if } i > 0 \end{cases}$$

$\therefore B = [0, A_1, A_1+A_2, A_1+A_2+A_3, \dots]$

$A = [1, 2, 3]$

$B = [0, 1, 3, 6]$ $n = 3$

$i = 0$ $j = 2$

prof =

```
1  max_gap ← 0
2  buy_sell_point ← (None, None)        n = size of A
3  B ← new array of size n+1
4  B[0] ← 0
5  for i ← 0 to n-1 do
6      B[i+1] ← B[i] + A[i]
7  for i ← 0 to n-1 do
8      for j ← i+1 to n do
9          profit ← B[j] - B[i]
10         if profit > max_gap:
               max_gap ← profit
               buy_sell_point ← (i, j)
```

for j ← i to n-1 do

profit ← B[j+1] - B[i]

if

⇒ //

← (i, j+1)

shouldn't it be j+1 instead of j ?

Q efficiency?

bad: performs well on real-inputs.? who decides it? depends on hardware

bad: performs better than brute-force ← ill-defined

good: Runs in polynomial time

↳ should we count the exact steps? No!

↳ what matters is the leading term: Asymptotic growth Analysis

"worst-case scenario" Big-O notation

smaller

$O(n)$

$\Omega(n)$

$\Theta(n)$ ← bigger

equal