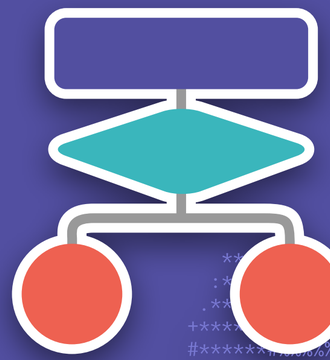
/* elice */

알고리즘의 정석

탐욕적 기법



엘리스 선생님

수강 목표

탐욕적 기법에 대해 알아봅니다

지금까지 과정에 대해서 복습해 봅니다

목차

탐욕적 기법

탐욕적 기법

순간의 최적의 선택이
궁극적으로 최적의 선택이다

단순한 방법도
문제 풀이에는 충분하다

[실습 1] 거스름돈

1원, 5원, 10원, 50원, 100원 짜리가 있을때,

돌려줘야 하는 동전의 최소 개수를 구하라

입력의 예

```
7
```

출력의 예

```
3
```

[실습 1] 거스름돈

1원 5원 10원 50원 100원

173원

[실습 1] 거스름돈

1원 5원 10원 50원 100원

173원

100원부터 시작해서 최대한 많이 나눠준다

[실습 2] 기울기가 가장 큰 두 점 찾기

기울기의 최댓값을 구하라

단, $1 \leq n \leq 100,000$

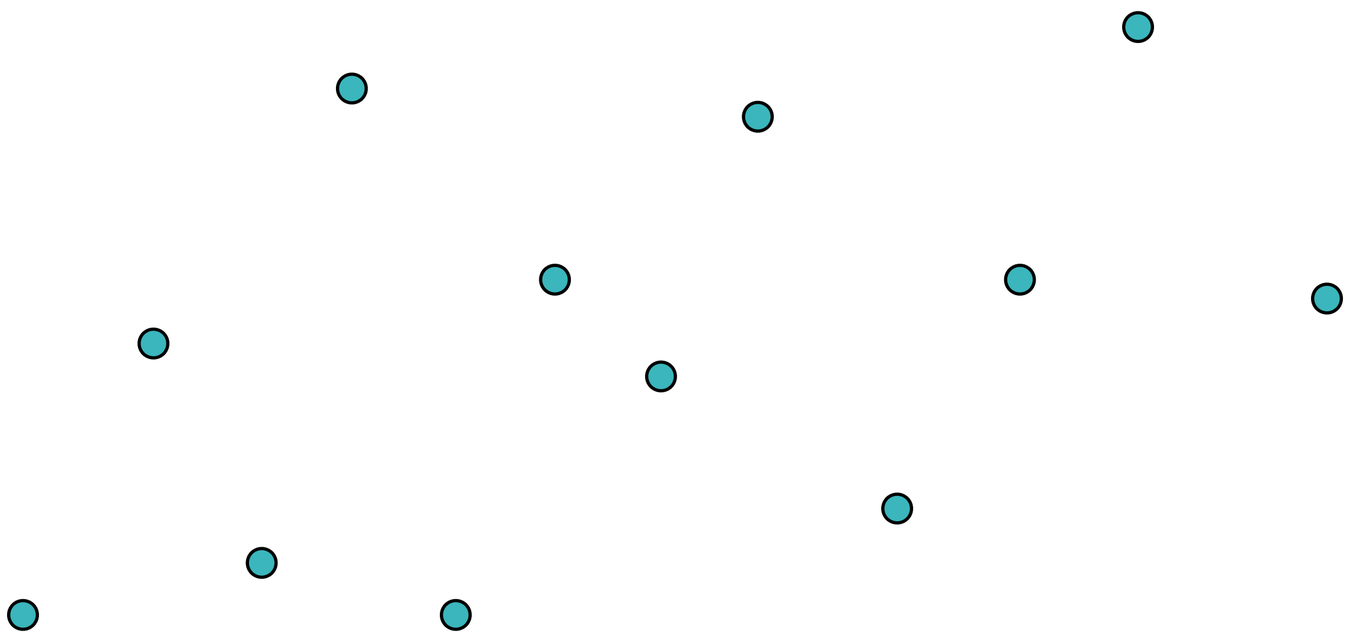
입력의 예

```
4
0 3
1 1
2 1
4 1
```

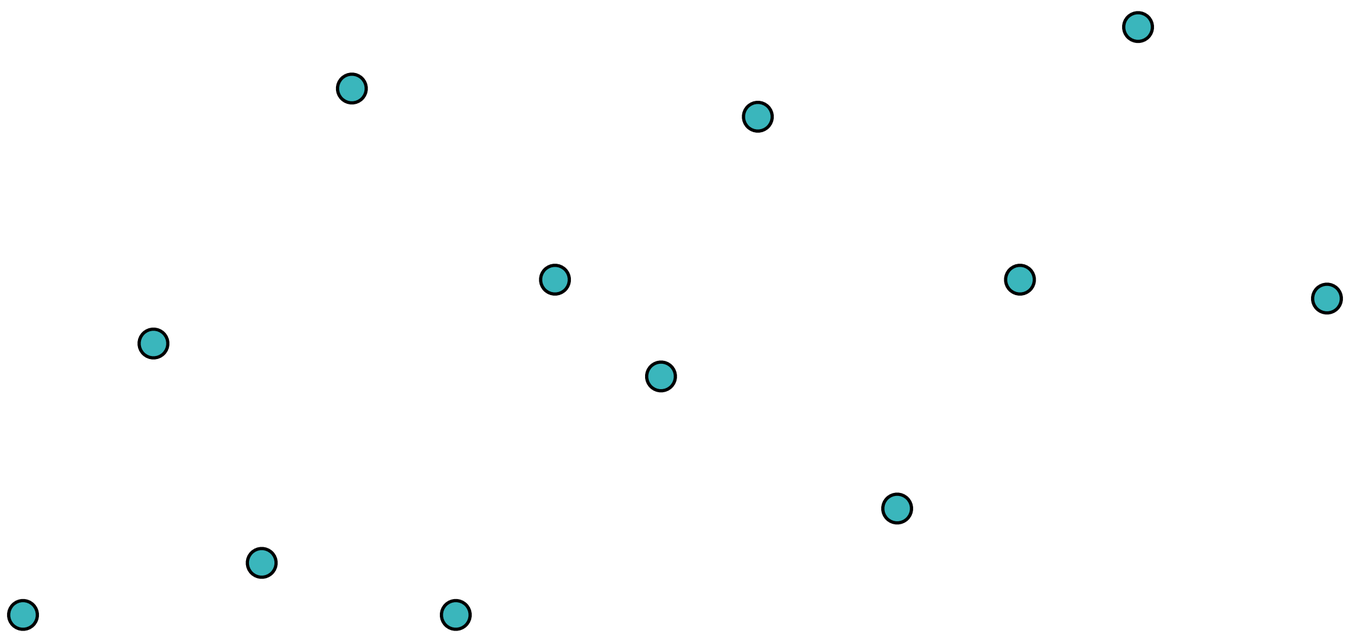
출력의 예

```
2.000
```

[실습 2] 기울기가 가장 큰 두 점 찾기

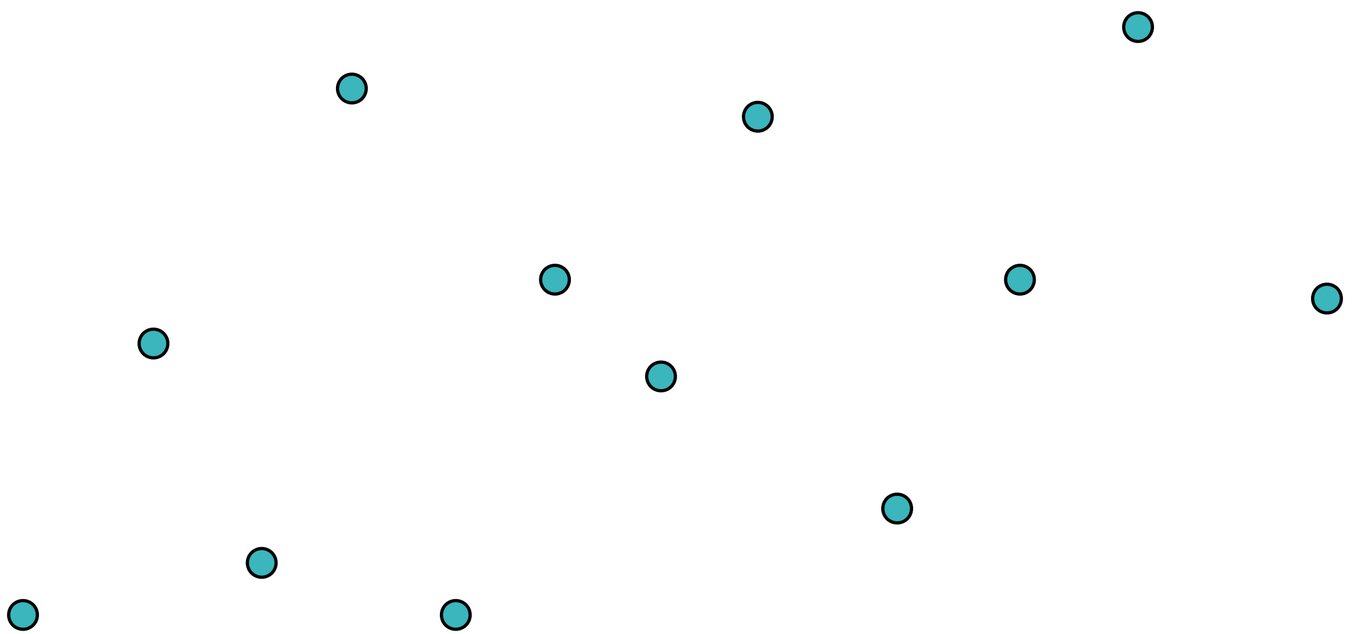


[실습 2] 기울기가 가장 큰 두 점 찾기

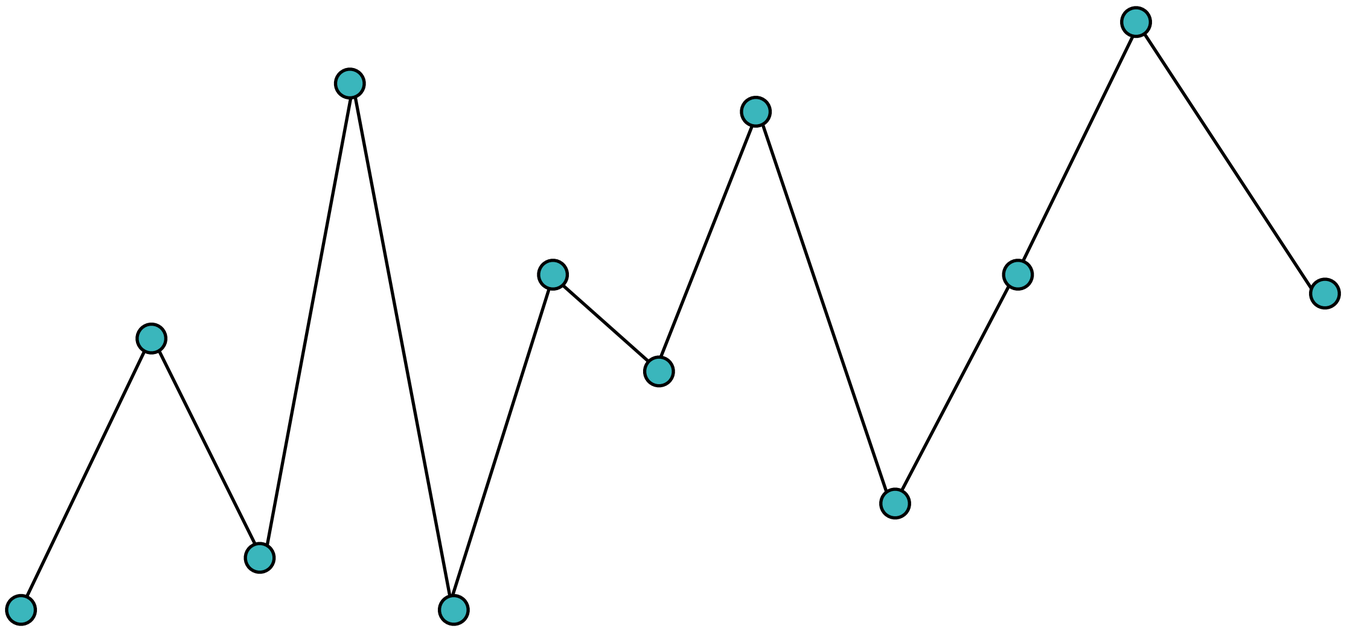


$O(n^2)$

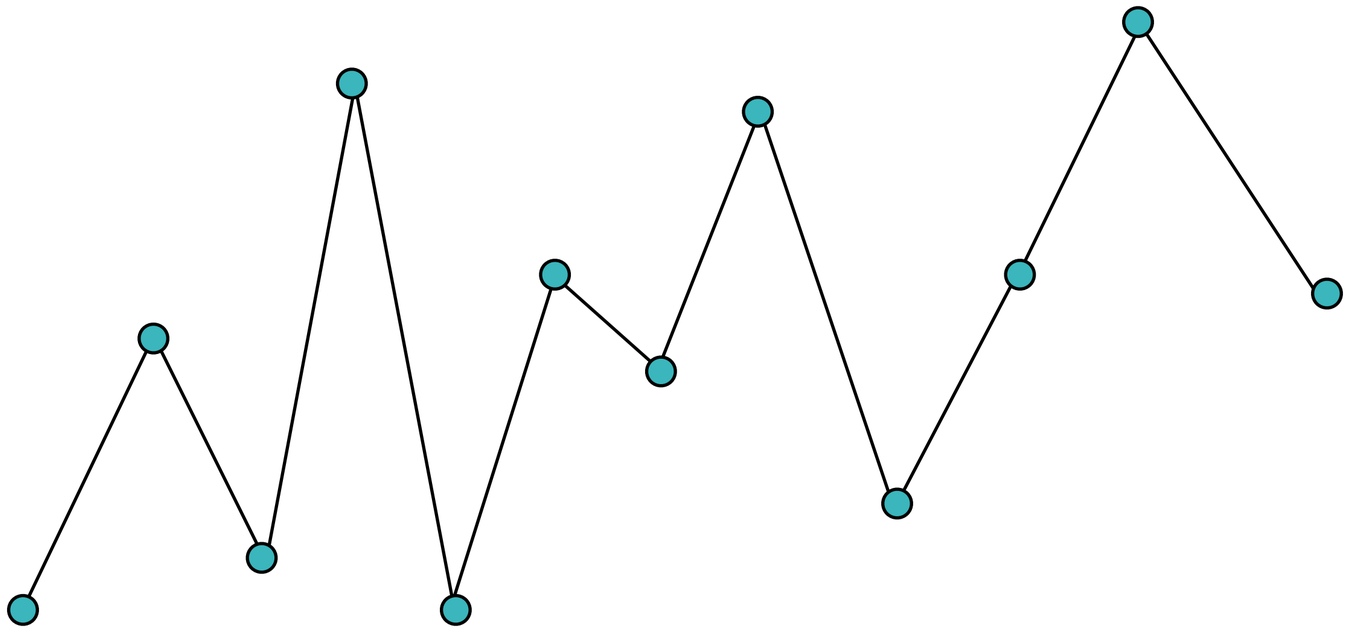
[실습 2] 기울기가 가장 큰 두 점 찾기



[실습 2] 기울기가 가장 큰 두 점 찾기

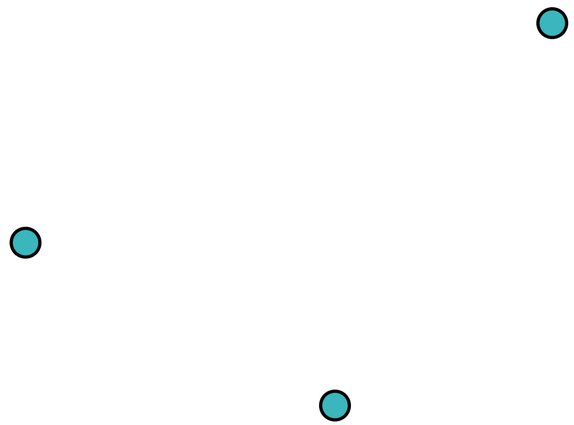


[실습 2] 기울기가 가장 큰 두 점 찾기

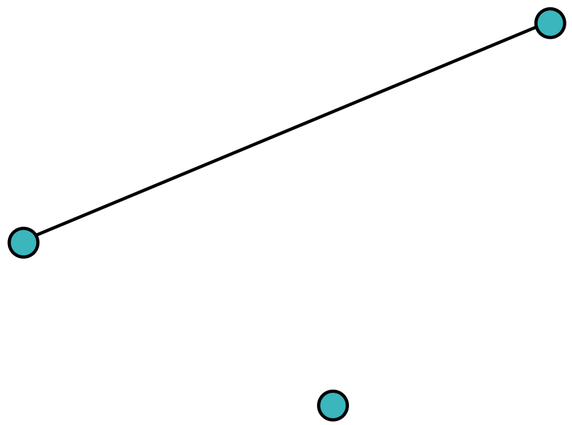


x축으로 인접한 두 점만 고려해도 충분하다

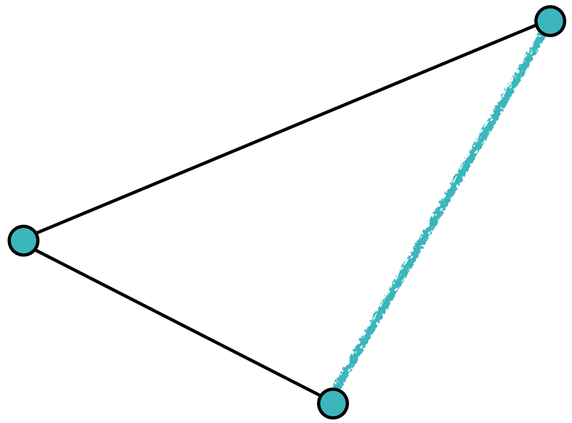
[실습 2] 기울기가 가장 큰 두 점 찾기



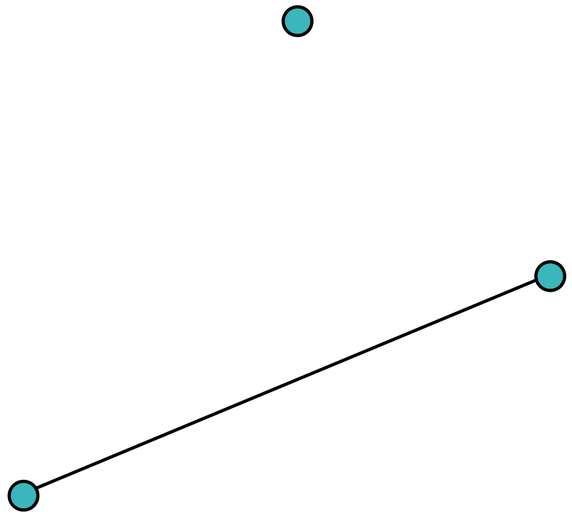
[실습 2] 기울기가 가장 큰 두 점 찾기



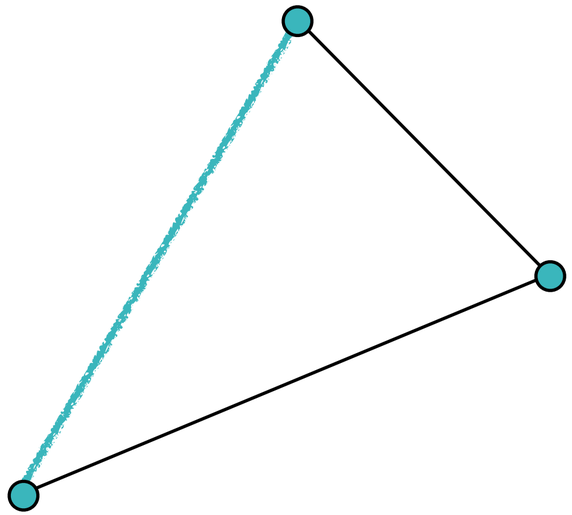
[실습 2] 기울기가 가장 큰 두 점 찾기



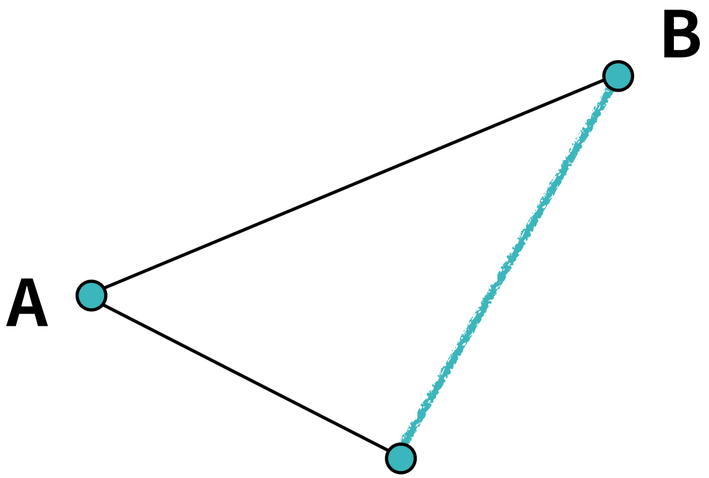
[실습 2] 기울기가 가장 큰 두 점 찾기

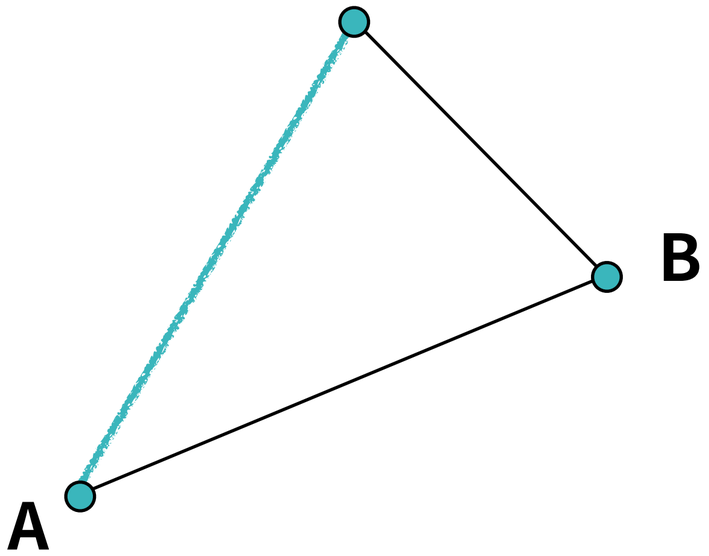


[실습 2] 기울기가 가장 큰 두 점 찾기



[실습 2] 기울기가 가장 큰 두 점 찾기





위와 같이 두 점 **A, B 사이에 점 C**가 존재하면,
A, B의 기울기는 **최댓값이 될 수 없다**

[실습 3] Fractional knapsack

가방에 최대한 가치가 높은 물건들을 담아보자

단, $1 \leq n \leq 100,000$

입력의 예

```
4 10
3 10
2 7
4 9
5 13
```

출력의 예

```
30.000
```

[실습 3] Fractional knapsack

가방에 최대한 가치가 높은 물건들을 담아보자

단, $1 \leq n \leq 100,000$

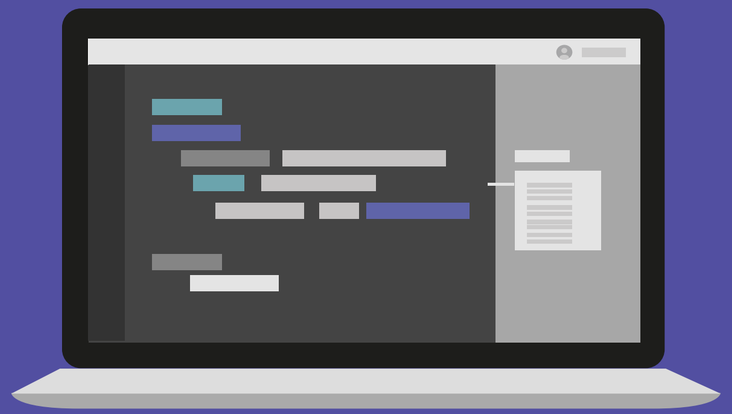
탐욕적 기법

단순한 방법도
문제 풀이에는 충분하다

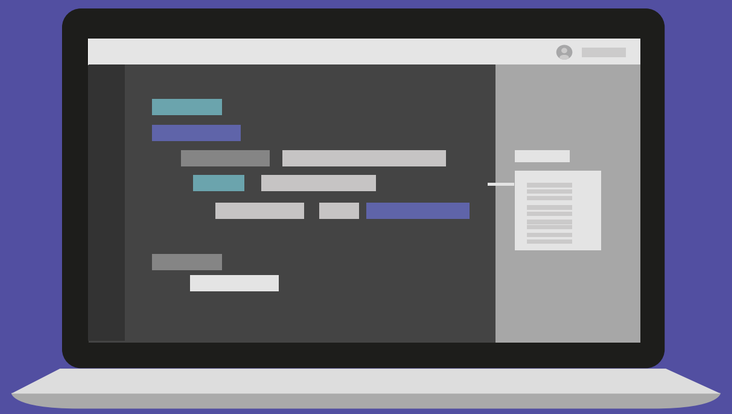
왜?를 증명하는 것이 가장 중요함

수학적 문제 해결력을 요구한다

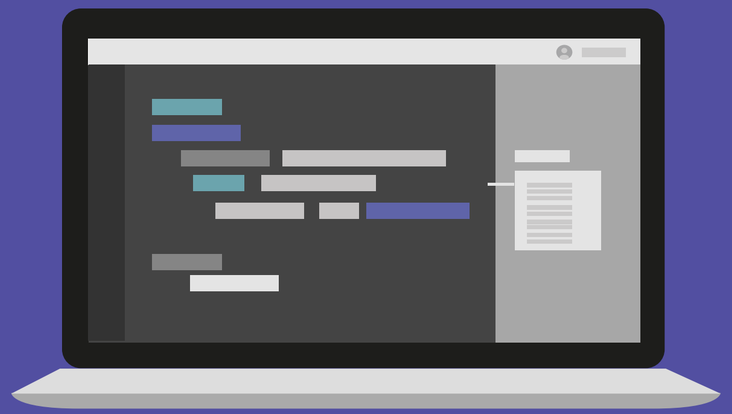
[실습 1] 거스름돈



[실습 2] 기울기가 가장 큰 두 점 찾기



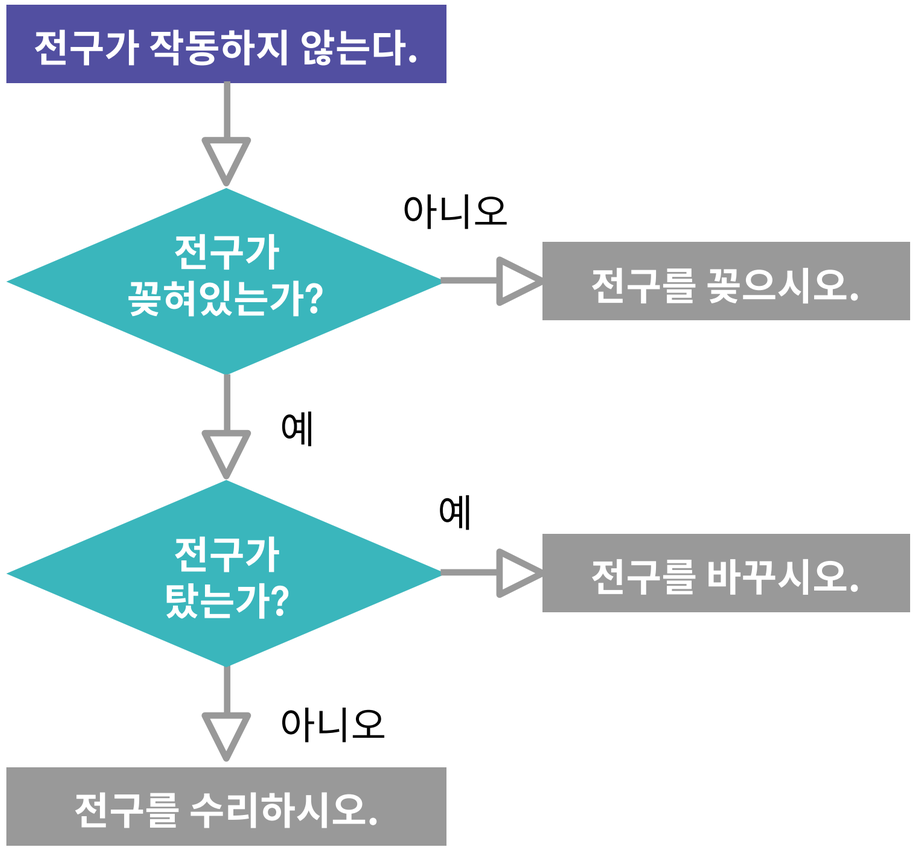
[실습 3] Fractional knapsack



과정 요약

알고리즘

문제를 해결하는 방법




```
def fixBulb(bulb) :
    if not bulb.isEmpty() :
        bulb.create()

    elif bulb.isBurnt :
        bulb.change()

    else :
        bulb.fix()
```

재귀호출

함수가 **자기 자신**을 호출

왜?

```
def Factorial(n) :
    if n == 0 :
        return 1
    else :
        return n * Factorial(n-1)
```

재귀적 계산 방법

Factorial(n) : n!을 반환하는 함수

Factorial(n) = Factorial(n-1) * n

n=1 일때는 **Factorial(n)**이 정상 작동한다.

n = 1	n = 2	n = 3	n = 4	n = 5	n = 6
1! =	2! =	3! =	4! =	5! =	**6! =**

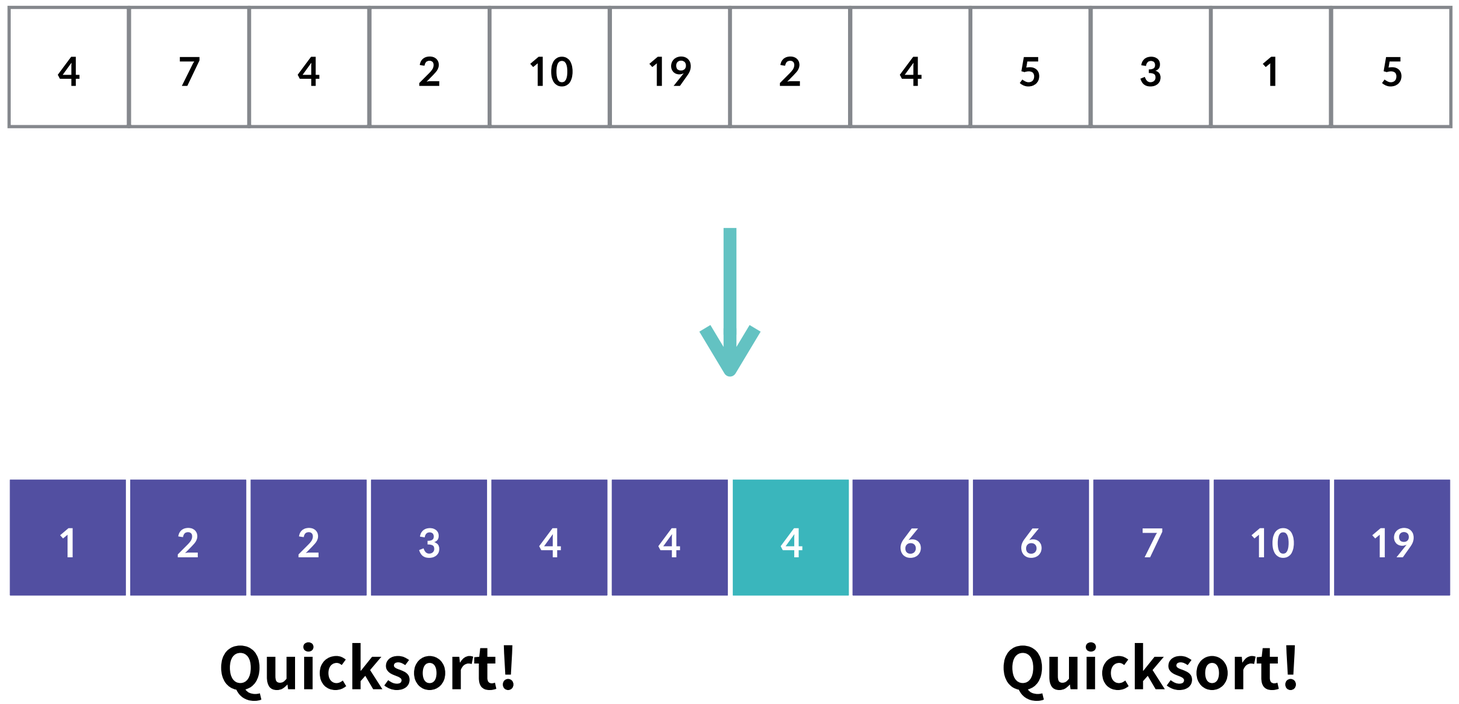
수학적 귀납법

명제 **P(n)**을 다음과 같이 증명하는 방법

1. **N = 1** 일 때 성립함을 보인다.
2. **P(k)**가 성립한다고 가정할 때, **P(k+1)**이 성립함을 보인다.
3. 따라서 **모든 자연수 n**에 대하여 P(n)이 성립한다.

퀵정렬 (Quick Sort)

재귀호출을 이용한 대표적인 정렬



완전 탐색 (Brute-Force)

가능한 모든 경우를 시도해 보는 것

가능한 모든 경우가 무엇인가?

완전 탐색의 중요성

문제가 주어지면

무.조.건

완전 탐색법으로 먼저 시도해야 한다.

Complexity Theory

문제 자체에도 복잡도가 존재한다

P class

NP-Complete class

알고리즘 과정에서 다루는 문제들

(거의 대부분) **P 문제**들만을 다룬다

알고리즘에서는 **고려해야 하는 경우**를
줄이는 방법을 배운다

하지만 대표적인 **NP-Complete 문제**는
알면 좋다

분할정복법

문제를 소문제로 **분할**

각각의 **소문제**를 해결

소문제의 해결 결과를 이용해 **전체 문제**를 해결

분할정복법

어렵다

분할정복법

분할정복법으로 해결할 수 있는 **대표적인 예제**

수학적 문제 해결 능력이 가장 중요

키보드 대신에 **노트와 펜**을 들고 생각

탐욕적 기법

단순한 방법도
문제 풀이에는 충분하다

왜?를 증명하는 것이 가장 중요함

수학적 문제 해결력을 요구한다

/* elice */

문의 및 연락처

academy.elice.io

contact@elice.io

facebook.com/elice.io

medium.com/elice