

## Algorithm Project

주제: 10000개의 20차원 벡터로 이루어진 도형에서 가장 volume이 큰 행렬의 volume 찾기

input: input.csv, 20 X 10000 행렬 (20 X 1 벡터가 10000개 있는 것과 같음),  
input file의 첫째 행은 인덱스 표시 (2행부터 21행까지가 실제 벡터 값)

output: 최대 volume, 해당 벡터의 집합 (벡터 인덱스 출력), running time

### ■ 과제 설명

- $n \times 1$  벡터  $m$ 개로 이루어진  $n \times m$  크기의 행렬  $A$ 의 volume,  $V$ ,는 다음과 같이 정의

$$V = \sqrt{|\det(A^T A)|}$$

여기서  $A^T$ 는 python의 transpose()를 이용하면 됨 (즉,  $A$ 가 numpy array로 정의됐을 때,  $A.transpose()$  or  $\text{numpy.transpose}(A)$ ), 마찬가지로  $\det(B)$ 의 값은  $\text{numpy.linalg.det}(B)$ 를 사용하면 됨

\*  $a > 0$ 이고 scalar 값일 때,  $\det(a) = a$

예)  $A = \begin{pmatrix} 15 \\ 26 \\ 37 \\ 48 \end{pmatrix}$ 라고 하면

$$V = \sqrt{\left| \det \begin{pmatrix} (15)^T & (15) \\ (26)^T & (26) \\ (37)^T & (37) \\ (48)^T & (48) \end{pmatrix} \right|} = \sqrt{\left| \det \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{pmatrix} \begin{pmatrix} 15 \\ 26 \\ 37 \\ 48 \end{pmatrix} \right|} = \sqrt{\left| \det \begin{pmatrix} 29 & 70 \\ 70 & 174 \end{pmatrix} \right|} = \sqrt{146}$$

- input에서  $A$ 를 구성하여 최대  $V$  값을 구하고 해당  $A$ 를 출력함
- 각자 제안 알고리즘에 대한 pseudo code 작성
- 각자 제안 알고리즘에 대한 time complexity 계산

### ■ 주의 사항

- 실행 시간 최소화 고려
- 평가 시에 최대 volume 값, 실행 시간을 고려하여 다음과 같은 performance metric이 클수록 좋은 성능을 가짐
  - \* performance metric: volume / running time (us)
- 실행 시간은 교수 컴퓨터로 측정
- sorting 함수는 python에 있는 자체 함수가 아닌 각자 구현한 함수 사용
- 모든 코드는 python으로 작성 (다른 언어 사용 불가)
  - \* 속도 측정을 위해서는 프로그램 언어가 동일해야 함

■ 제출 자료

- 결과 보고서 (hwp or word)
  - \* pseudo code 포함
  - \* time complexity 분석 포함
  - \* 알고리즘 및 결과 설명
- 소스코드
  - \* .py 코드 제출