그리디 알고리즘

그리디 알고리즘은 **기준에 따라 좋은 것을 선택하는 알고리즘** 으로 문제에서 '가장 큰 순서대로', '가장 작은 순서대로'와 같은 기준을 제시해준다. 정렬 알고리즘을 사용했을 때 만족시킬 수 있으므로 정렬과 그리디는 주로 짝을 지어 출제됩니다.

특징

- 사전에 외우고 있지 않아도 풀 수 있을 가능성이 높다
- 문제 출제의 폭이 넓고 암기가 아닌 그 때마다의 상황으로 문제를 해결해가야한다.

예시 1) 거스름돈 문제

'가장 큰 화폐 단위부터' 돈을 거슬러 주는 것

화폐의 종류만큼 반복문을 수행해야 하기에 화폐의 종류가 K개라면 시간 복잡도는 O(K)입니다.

거슬러주어야하는 금액 N은 시간복잡도에 영향을 주지 않습니다.

그리디 알고리즘의 정당성

그리디 알고리즘은 '최적의 해'를 찾을 가능성이 낮다.

탐욕적으로 문제에 접근했을 때 정확한 답을 찾을 수 있다는 보장이 있을 때 효과적인 방법

거스름돈 문제를 예로 들어보자

→ 갖고 있는 동전 중 큰 단위가 항상 작은 단위의 배수였기에 **작은 단위의 동전들을 종합해 다른 해가 나오게 할 수 없다**.

케이스로 500, 400, 100을 이용해 800원을 거슬러주는 문제에서 그리디로 접근하는 경우 [500, 100, 100, 100]이 되지만 사실 [400, 400]이 정답이 되기 때문이다.

실제로 동전의 단위가 서로 배수 형태가 아니라 무작위로 주어진 경우에는 그리디 알고리즘으로는 해결할 수 없다.

즉, 대부분의 그리디 알고리즘 문제에서는 **최소한의 아이디어를 떠올리고 이것이 정당한지 검토할 수 있어야 옳은 답을 도출**할 수 있습니다.

예시 2) 큰 수의 법칙

다양한 수로 이루어진 배열에서 주어진 수들을 M번 더하여 가장 큰 수를 만드는 방법 (단, 서로 다른 인덱스에 해당하는 수가 같은 경우에도 서로 다른 것으로 간주)

풀이 방법

⇒ 가장 큰 수와 두번째로 큰 수를 고르고 가장 큰수 * K번 + 두번쨰로 큰 수의 과정을 M에 도달할 때 까지 반복한다.

⇒ "**반복되는 수열에 대한 파악**"이 필요하다.

예시 3) 숫자 카드 게임

M x N 배열의 카드에서 각 행마다 가장 작은 수를 찾은 뒤에 그 수 중에서 가장 큰 수를 찾는 문제

list에서 min 함수와 2중 반복문 구조를 이용하면 쉽게 구할 수 있었던 문제

예시 4) 1일 될 때까지

주어진 N에 대해서

1. N에서 1을 뺀다
2. N을 K로 나눈다

위 두가지 조건을 만족하고 N을 계산하면서 1이 될 때까지 반복

이 문제의 포인트는 "최대한 많이 나누기"입니다. 이 포인트를 갖고 조금 더 자세하게 구현 계획을 세워보면..

1. N이 K의 배수가 될 때까지 1씩 빼기
2. N을 K로 나누기

이걸 계속 for문으로 돌리면 데이터 크기에 따라 시간초과가 날 수 있다.

N이 100억 이상의 큰 수면 시간초과가 발생할 수 있다.

빠르게 동작하려면 N이 K의 배수가 되도록 효율적으로 한번에 빼는 방식으로 사용하는 것이 좋다.