Problem solving homework#1

Largest N Numbers

2017314712 김우진

1. 과제 목적

이차원 배열 A[N]N]이 있다. 배열안에는 N^2의 수가 있으며 이 수들은 각각 다른 크기의 제한이 없는 양수들이며 N^2수중 가장 큰 N개의 수를 찾아야한다. 시간의 복잡성을 구해야 하며 알고리즘은 배열의 초기상태를 정의할 필요 없다.

1. 알고리즘

가장 큰 수 N개를 찾기 위해 일단 행단위로 A[N][N]을 N개의 A[N]으로 분리한다. 그 후 각각의 배열을 merging sort를 이용해서 정리한다. 그럼 N개의 배열이 정리가 되고 N개의 배열 오른쪽에 각행의 가장 큰 수가 놓이게 된다. 그 후 각 **오른쪽 숫자들만 따로 모아 B[N]을 만든다**. B[N]의 원소 순서는 행의 순서대로 정리한다. B[N]에서 원소들의 비교를 통해 가장 큰 원소를 추출하고 추출한 원소의 빈자리는 추출된 원소의 배열 A[N]의 두번째로 큰 원소를 넣는다. 그 후 이과정을 N-1번 더 반복한다. (밑에 그림 참고)

1. 시간 복잡도

N개의 A[N]배열 merging sort : N\*O(nlogn)

B[N]에서 가장 큰 수 찾기: N-1

따라서 시간 복잡도는 N\*O(nlogn)이다.

Problem solving homework#2

Minimum Height

2017314712 김우진

1. 과제 목적

두개의 연속된 숫자들의 집합이 있다. 집합의 원소는 각각 다르며 두 집합의 원소는 동일하지만 나열되었을 때 그 순서가 다르다. Ghost leg로 나열된 두 수를 가장 작은 높이로 연결하는 것이 목적이다.

1. 알고리즘

먼저 각각의 수들마다 오른쪽 또는 왼쪽으로 얼마나 이동해야 되는지 파악한다. **이동해야하는 양을 배열 A[N]**에 저장하고 **방향 값을 배열 B[N]**에 저장한다. 왼쪽부터 줄을 긋기로 하는데 각 줄마다 층수를 비교하기 위해 번호를 부여하기로 하겠다. 첫번쨰로 주은 선을 0이라하고 그 **밑에 그어질수록 -1**씩 하며 **위에 그려질수록 +1**을 부여한다. 그 후 왼쪽 줄부터 가지고 있는 층의 줄을 기록한다. 이 과정은 배열대신 **linked list**를 이용한다. 왼쪽부터 줄부터 차례대로 방향 및 이동해야 하는 양을 확인 한 후 줄을 긋는다. 1. 오른쪽으로 줄을 그어야 하는 경우 중심줄의 최대 층 줄보다 한층 더 높은 곳에서 줄을 긋기 시작한다. 2. 왼쪽으로 줄을 그어야 하는 경우는 밑에 그림으로 설명하겠다. 이 과정을 마지막 줄까지 이어서 한다.