2024 겨울학기 동국대학교SW역량강화캠프

18일차. 기말고사

문제

- 동국대는 가위바위보를 좋아해

문제

부트캠프를 진행하기 위해 동국대를 드나들던 유섭이는 재미난 사실을 발견하였는데, 바로 동국인들은 서로를 만나면 인사 대신 가위바위보를 한다는 것이었다.

동국인들은 가위, 바위, 보 중 자신이 좋아하는 손 모양을 가지고 있으며, 이후 가위바위보 게임에서 자신이 좋아하는 손 모양을 낸다. 그러나 동국인들이 누구인가, 역사를 걸으면 동국이 보이고, 동국을 걸으면 역사가 된다는 역사를 쓰는 자들이 아닌가. 가위바위보에서 진 동국인은 역사를 새로 쓰기 위해 즉시 자신이 좋아하는 손 모양을 상대방을 이길 수 있는 손 모양으로 바꾸며, 이후 가위바위보 게임에서는 바꾼 손 모양을 낸다고 한다.

승부욕이 강한 유섭이는 제자들을 모두 이기기 위해 조교인 구연쌤에게 수업이 끝나는 시점에 학생들이 좋아하는 손 모양을 모두 알아 오라고 시켰다. 성실한 구연쌤은 수업 전 학생들이 좋아하는 손 모양을 모두 조사하였지만, 불행히도 수업 도중 가위바위보를 한 학생이 생겨버렸다. 다행히 유능한 구연쌤은 시간 순으로 가위바위보를 한 학생들의 쌍을 완벽히 파악하였다.

이제는 여러분이 유능한지 증명할 차례이다. 수업 전 구연쌤이 조사한 내용과 수업 중 가위바위보를 한 학생들의 정보가 주어졌을 때, 수업 후 학생들이 좋아하는 손 모양을 알아내어라.

입력

첫째 줄에는 학생들의 수 N과 수업 중 가위바위보를 한 횟수 M이 빈 칸을 사이에 두고 주어진다. $(2 \leq N \leq 10^5, 0 \leq M \leq 10^5)$

풀이

학생들의 이름이 문자열로 주어지고, 학생들이 좋아하는 손 모양을 관리해야 하기 때문에 Map을 이용하여 학생들을 관리해주면 된다. (Python은 dictionary)

Rock, scissors, paper를 문자열 자체보다 0, 1, 2로 저장하면 관리가 수월하다.
0 > 1
1 > 2
2 > 0
(x+1) = y 이면 x 승
x = (y+1) 이면 y 승

이름순으로 정렬하여 출력해야 하므로 Java는 TreeMap을 이용하는게 수월하며, Python은 items를 정렬하여 출력해야 한다.

코드

```
st = new StringTokenizer(br.readLine());
int N = Integer.parseInt(st.nextToken());
int M = Integer.parseInt(st.nextToken());

TreeMap<String, Integer> students = new TreeMap<>();
for(int i=0; i<N; i++){
    st = new StringTokenizer(br.readLine());
    String name = st.nextToken();
    String like = st.nextToken();
    int like_num=0;

    if(like.equals("rock")) like_num = 0;
    if(like.equals("scissors")) like_num = 1;
    if(like.equals("paper")) like_num = 2;

    students.put(name, like_num);
}
```

```
for(int i=0; i<M; i++){
    st = new StringTokenizer(br.readLine());

    String student1 = st.nextToken();
    String student2 = st.nextToken();

    int num1 = students.get(student1);
    int num2 = students.get(student2);

    if((num1 + 1) % 3 == num2) students.replace(student2, (num2+1)%3);
    else if(num1 == (num2 + 1) % 3) students.replace(student1, (num1+1)%3);
}
```

문제

- 젠가의 신

문제

종범이는 젠가의 신이다.
어떠한 상황에서도 나무토막을 빼던 종범이는 일렬로 세운 젠가에서도 나무토막을 빼는 경지에 이른 것이다.

나무토막을 일렬로 세웠기에 맨밑 혹은 중간에 있는 나무토막을 빼게 되면 위에 있는 나무토막들이 한칸씩 내려오게 되며, 뺀 나무토막은 즉시 젠가의 맨 위에 올려야 한다.

종범이가 가진 젠가에는 1부터 N까지 숫자가 붙어 있는데, 일렬로 세운 젠가에서도 나무토막을 빼는 경지에 이른 종범이는 이번에는 자신의 능력을 사용하여 나무토막을 1번 나무토막부터 N번 나무토막까지 차례로 쌓이도록 만들어보고자 한다.

그러나 젠가는 엄청난 집중력을 요구하는 게임으로 젠가의 신 종범이도 나무토막을 빼는 것에 상당한 피로를 느낀다. 따라서 종범이는 나무토막을 빼는 횟수를 최소로 하여 피로를 최소화 하고자 한다. 종범이를 도와 현재 젠가의 상태가 주어졌을 때, 몇개의 나무토막을 추가로 빼면 되는지 알려주자.

입력

첫째 줄에는 젠가에 사용된 나무토막의 개수를 나타내는 하나의 정수 N이 주어진다. ($1 \leq N \leq 10^5$)

풀이

K개의 나무토막을 빼야 한다면, 그 순서도 알아내야 하는가?

한번 뺀 나무토막을 또 빼는게 유리한 경우가 있는가?

풀이

K개의 나무토막을 빼야 한다면, 그 순서도 알아내야 하는가? NO

한번 뺀 나무토막을 또 빼는게 유리한 경우가 있는가? NO

어떤 나무토막이 빼야 하는 나무토막인지만 결정해주면 된다.
= 건드리지 않아도 되는 나무토막이 무엇인지 결정해주면 된다.

한번 빼기로 한 나무토막은 더 이상 신경 쓰지 않아도 된다.

풀이

맨 아래 1번 나무토막이 있게 하려면?
1번 나무토막 아래에 있는 나무토막은 모두 빼야 한다.

1번 나무토막 위에 2번 나무토막이 있게 하려면?
1번 이후 2번 나무토막 아래에 있는 나무토막을 모두 빼면 된다.

3번 나무토막은?
…

그런데 만약 x번 나무토막을 고려해야 하는데, x번 나무토막이 빼기로 결정한 나무토막이라면?
X번 나무토막은 기존의 나무토막들보다 위로 가게 된다.
즉, (x-1)번 나무토막보다 위에 있던 모든 나무토막들은 적어도 한번은 빼야 한다.

앞에서부터 1, 2, 3, 4, … 를 찾다가 처음으로 찾을 수 없는 숫자부터는 전부 빼야 하는 번호에 해당한다.

코드

```
int N = Integer.parseInt(br.readLine());
int next=1;

st = new StringTokenizer(br.readLine());
for(int i=0; i<N; i++){
    int x = Integer.parseInt(st.nextToken());
    if(x == next) next++;
}

System.out.print(N-next+1);
```

문제

- 형스킨라빈스 주니원 포장 주문

문제

형준이는 형스킨라빈스 주니원이라는 가게를 개업하였다.

형스킨라빈스 주니원 가게는 아이스크림 전문점인데, 아이스크림을 매장에서 먹을 수도 있고, 포장이나 배달을 할 수도 있다.
그 중 포장이나 배달 시, 아이스크림과 함께 형준아이스를 넣어 아이스크림이 녹지 않고 목적지까지 갈 수 있도록 해준다.

형준이가 개발한 형준아이스는 특별한 성질을 가지고 있다. 크기가 C인 형준아이스는 바깥으로 나가면 C분 동안 아이스크림이 녹지 않게 해주다 C분을 넘으면 그 기능을 상실하지만, 실내에 들어가는 즉시 초기 상태로 돌아가 다시 실외에서 C분 동안 아이스크림이 녹지 않게 해준다.

설명을 들으면 알겠지만 형준이가 개발한 형준아이스는 꽤 비싸서 형준이는 포장 주문이 들어오면 가능한 작은 크기의 형준아이스를 사용하고자한다.
다행히도 포장 손님들은 아이스크림이 녹지 않고 목적지까지 가는 경로가 존재한다면 시간이 얼마나 걸리든 해당 경로로 간다고 한다.

형준이를 도와 형스킨라빈스 주니원 주변 지도에 대한 정보가 주어졌을 때, 필요한 형준아이스의 최소 크기를 구해주자.

입력

첫째 줄에 건물의 개수와 건물을 잇는 도로의 개수를 의미하는 두개의 정수 N, M이 공백을 사이에 두고 주어진다. ($2 \leq N \leq 10^5, N-1 \leq M \leq 2 \times 10^5$)

풀이

형스킨라빈스 주니원에서 v번 건물까지 필요한 형준아이스의 최소 크기를 x로 안다고 해보자.

v번 건물에 오는 순간 형준아이스가 초기 상태로 돌아가기 때문에 v번 건물과 연결된 도로의 길이가 x 이하인 건물들은 모두 x 크기의 형준아이스로 갈 수 있다.

연결된 도로의 길이가 x 초과인 건물들은 해당 v건물을 통해 가려면 해당 도로의 길이만큼의 형준아이스가 필요할 것이다.

즉, v건물과 u건물을 연결하는 도로의 길이를 d라고 한다면, 다음과 같은 관계식이 성립할 것이다.
d[u] = min(x[u], max(x[v], d))

이 값들은 우선순위큐를 이용해 그래프 순회를 통해 구해줄 수 있다.

코드

```
PriorityQueue<Edge> pq = new PriorityQueue<>();
pq.offer(new Edge(1, 0));
while(!pq.isEmpty()){
    Edge tmp = pq.remove();
    int v = tmp.getTo();
    int d = tmp.getDis();

    if(v == N){
        System.out.print(d);
        return;
    }

    if(chk[v]) continue;
    chk[v] = true;

    for(Edge e: edge[v]){
        int to = e.getTo();
        int dis = e.getDis();

        pq.offer(new Edge(to, Math.max(d, dis)));
    }
}
```

문제

- 형스킨라빈스 주니원 배달 주문

문제

형준이는 형스킨라빈스 주니원이라는 가게를 개업하였다.

형스킨라빈스 주니원 가게는 아이스크림 전문점인데, 아이스크림을 매장에서 먹을 수도 있고, 포장이나 배달을 할 수도 있다.
그 중 포장이나 배달 시, 아이스크림과 함께 형준아이스를 넣어 아이스크림이 녹지 않고 목적지까지 갈 수 있도록 해준다.

형준이가 개발한 형준아이스는 특별한 성질을 가지고 있다. 크기가 C인 형준아이스는 바깥으로 나가면 C분 동안 아이스크림이 녹지 않게 해주다 C분을 넘으면 그 기능을 상실하지만, 실내에 들어가는 즉시 초기 상태로 돌아가 다시 실외에서 C분 동안 아이스크림이 녹지 않게 해준다.

설명을 들으면 알겠지만 형준이가 개발한 형준아이스는 꽤 비싸서 형준이는 배달 주문이 들어오면 가능한 작은 크기의 형준아이스를 사용하고자한다. 배달 특성상 제한 시간이 존재하며, 직업 정신이 투철한 배달 기사님은 제한 시간 안에 아이스크림이 녹지 않고 목적지까지 가는 경로가 존재한다면 해당 경로로 간다고 한다.

형준이를 도와 형스킨라빈스 주니원 주변 지도에 대한 정보가 주어졌을 때, 필요한 형준아이스의 최소 크기를 구해주자.

입력

첫째 줄에 건물의 개수와 건물을 잇는 도로의 개수, 배달 제한 시간을 의미하는 3개의 정수 N, M, T가 공백을 사이에 두고 주어진다. ($2 \leq N \leq 10^5, N-1 \leq M \leq 2 \times 10^5, 1 \leq T \leq 10^9$)

풀이

문제를 바꿔 생각해보자.

시간제한이 아니라, 오히려 형준아이스의 크기가 정해지고, N번 건물까지의 최단경로를 구하라고 한다면?
도로의 길이가 형준아이스 크기보다 큰 간선을 제외한 그래프에서 최단경로를 구해주면 된다.

형준아이스를 이분탐색해주면 된다.

코드

```
int l = 0;
int r = (int)1e9;
while(l < r){
    int m = (l+r)/2;

    int res = check(m);
    if(res == -1 || res > T) l = m+1;
    else r = m;
}

System.out.print(l);
```

```
static int check(int limit){
    int[] dist = new int[N+1];

    PriorityQueue<Edge> pq = new PriorityQueue<>();
    pq.offer(new Edge(1, 1));
    while(!pq.isEmpty()){
        Edge tmp = pq.remove();
        int v = tmp.getTo();
        int d = tmp.getDis();

        if(dist[v] > 0) continue;
        dist[v] = d;

        for(Edge e: edge[v]){
            int to = e.getTo();
            int dis = e.getDis();

            if(dis > limit) continue;
            pq.offer(new Edge(to, d + dis));
        }
    }
    return dist[N]-1;
```

문제

- 동국대에서도 클라이밍을 할 수 있어요

문제

동국대에는 동국인들도 모르는 신기한 사실이 있다. 바로 동국대 학생회관 옆면에는 클라이밍 구조물이 설치되어 있다는 것이다! (진짜다)

3년이나 동국대를 다녔지만 이 사실을 처음 안 정현이는 소스라치게 놀랐고, 수소문 끝에 클라이밍을 할 수 있는 기회를 얻었다.

동국대 학생회관 옆에 설치된 클라이밍 구조물에는 N개의 홀드가 설치되어 있으며, 클라이밍은 이 바위 모양의 홀드를 손이나 발로 딛거나 끌어당겨 목표 지점에 도달하는 스포츠이다.
기존 클라이밍은 전략에 따라 밑에 있는 홀드를 잡기도 하고, 왼쪽과 오른쪽 홀드를 번갈아 잡으며 진행하지만, 정상만을 바라보는 정현이에겐 후퇴란 없다. 왼쪽 아래에 있는 정현이는 오른쪽 위를 목표로 삼았으며, 이전에 선택한 홀드보다 더 아래나 왼쪽에 있는 홀드는 절대로 잡지 않는다고 한다. 또한 욕심 많은 정현이는 최대한 많은 홀드를 선택하여 정상을 향하고자 한다.

설치된 홀드에 대한 정보가 주어졌을 때, 정현이가 선택할 수 있는 홀드의 최댓값을 구해주자.

홀드에 대한 정보는 음이 아닌 두 정수 x_i, y_i로 주어지며, 이는 처음 정현이의 위치에서 오른쪽으로 x_i만큼, 위로 y_i만큼 떨어진 위치를 의미한다.

입력

첫째 줄에 홀드의 개수를 의미하는 하나의 정수 N이 주어진다. ($1 \leq N \leq 10^5$)

풀이

K개의 홀드를 결정했을 때, 이들의 x좌표와 y좌표가 모두 같거나 증가하는 순서가 돼야 한다.

x좌표와 y좌표 2개나 있으니 머리 아프다.
x좌표 순으로 정렬을 하자. (같으면 y좌표 순으로도 정렬해주자.)

x좌표순으로 정렬을 했기 때문에 k개의 홀드를 결정하면 그 순서가 정렬한 리스트의 순서와 일치해야 할 것이다.

그렇다면 우리는 해당 리스트에서 y좌표만 고려했을 때, 같거나 증가하는 부분 수열을 구하면 된다
LIS 를 구해주면 된다.

LIS의 정의는 증가하는 부분 수열이며, 이 문제에서는 같거나 증가하는 부분수열을 구해야 함에 주의하자.

코드

```
Collections.sort(cor);

int[] lis = new int[N];
ArrayList<Integer> K = new ArrayList<>();
K.add(0);
int LIS=0;
for(int i=0; i<N; i++){
    int y = cor.get(i).getY();
    lis[i] = upper_bound(K, y);

    if(K.size() == lis[i]) K.add(y);
    K.set(lis[i], y);

    LIS = Math.max(LIS, lis[i]);
}

System.out.print(LIS);
```