알고리즘 설계과제 1

[도시 설계 프로그램 설계 1차]

1. 과제 개요

인하왕국은 여러 도시들을 건설하였다. 하지만 현재 도시들 간에는 비포장도로들로 연결되어 있어, 도보로만 이동할 수 있다. 자동차로 이동할 수 있게 임의의 도로를 선택하여 포장도로로 개선하고자 한다. 만약 도시 A, B 사이에 포장도로가 있고, B, C 사이에 포장도로가 있다면, A에 살고 있는 시민들은 자동차를 타고 바로 도시 C로 갈 수는 없지만, 자동차로 도시 B를 거쳐 도시 C에 도착할 수 있다. 따라서 이런 경우는 도시 A, B, C는 자동차로 왕래가 가능하다. 국왕은 최소 비용으로 모든 도시들을 자동차로 왕래할 수 있게 연결하고자 한다.

강의시간에 배운 Kruskal의 최소신장트리(Minimum Spanning Tree, 이하 MST)를 찾는 알고리즘을 이용하여 모든 도시들을 자동차로 왕래할 수 있도록 도로들을 포장하는데 드는 최소 비용을 계산하는 프로그램을 설계하시오. 단, 도로는 모두 왕복도로를 의미하고, 비포장도로를 포장도로로 개선하는데 드는 비용은 도로의 길이와 같다고 가정한다. 이때 다음의 조건을 이용하여 MST를 찾는 프로그램을 구현한다.

- **조건 1** – Union-Find(또는 Disjoint set) 자료구조는 연결리스트표현(linked-list representation)으로 구현하며, 두 집합의 union은 크기가 더 작거나 같은 집합을 다른 집합의 tail에 연결하는 가중치-유니온 휴리스틱(weighted-union heuristic)을 이용한다. 이때 union 되는 두 집합의 크기가 동일하다면, 두 집합 중 leader가 더 작은 집합으로 union된다. 즉, leader가 각각 A, B인 서로 다른 두 집합 A, B에 대해, $|A|<|B|$이면, union(A, B)는 원소의 수가 더 적은 A가 B의 tail에 연결되고, 새로운 집합의 leader는 B이다. 만약 $|A|=|B|$이면, union(A, B)는 B가 leader id가 더 작은 A의 tail에 연결되고, 새로운 집합의 leader는 A이다.

  ※ 그래프는 자유롭게 구현 가능

  ※ Union-Find를 forest 기반으로 구현할 경우, 부정행위로 처리됨
- **조건 2** – 최소 비용으로, 비포장도로를 포장도로로 개선하여 모든 도시들을 연결하는 방법은 여러 가지 존재할 수 있다. 즉, MST를 찾는 어떤 단계에서는 최소 비용으로 공사할 수 있는 비포장도로가 여러 개 존재할 수 있다. 이 경우, 더 많은 시민들의 편의성을 위해, 포장하려는 도로의 양쪽 끝단의 두 도시의 인구수의 합이 더 큰 비포장도로를 우선적으로 개선한다. 만약 양쪽 끝단의 두 도시의 인구수의 합도 같은 경우, 양쪽 끝단의 도시 번호 중 더 작은 번호끼리 비교하여, 더 작은 도시 번호를 포함하는 비포장도로를 먼저 개선한다.
- **조건 3** – STL은 vector, list, string, 표준입출력, algorithm 헤더의 sort만 허용한다.

2. 수행할 기능

(0) 데이터 입력받기

프로그램을 실행하면 다음의 왕국의 정보를 표준입출력으로 입력받는다.

1) 첫 번째 줄에는 설립된 도시들의 개수 $N(2 \le N \le 50{,}000)$, 도로들 사이에 존재하는 비포장도로의 개수 $M(1 \le M \le 500{,}000)$, 질의의 수 $q(1 \le q \le 25{,}000)$가 공백으로 구분되어 주어진다.

2) 이후 N개의 줄을 통해 도시 정보가 아래와 같이 한 줄에 주어지고, 각 필드는 공백으로 구분된다.

필드1	필드2	필드3
도시 번호	도시명	도시의 인구수

① 도시 번호: 1~999,999 사이의 정수 **(기준 키. 유일함)**
② 도시명: 알파벳 20글자 이내 (예: "HongGilDong")
③ 도시의 인구수: 1~1,999,999 사이의 정수

3) 이후 M개의 줄을 통해 비포장도로의 정보가 한 줄로 주어지며 각 필드는 공백으로 구분된다.

필드1	필드2	필드3
도시 1	도시 2	두 도시 간의 도로 길이

① 도시 1: 도시 번호
② 도시 2: 도시 번호
③ 두 도시 간의 도로 길이: 1~10,000 사이의 정수

(1) 중간정보 출력: 특정 도시가 포함된 연결리스트의 내부 정보 파악

- 질의형식 : "N v k"
  - N: 도시 v가 포함된 집합에서 헤드로부터 k번째 노드를 출력하는 질의(query)
  - v: 도시 번호
  - k: 정수
- 출력형식 : "x y" 또는 "no exist"
  - x: 도시 v가 포함된 linked-list의 헤드로부터 k번째 도시의 번호
  - y: 도시 v가 포함된 linked-list의 헤드로부터 k번째 도시명
  - ✓ no exist: v가 포함된 linked-list의 헤드로부터 k번째 도시 정보가 존재하지 않을 때 "no exist" 출력

(2) 중간정보 출력: 특정 도시가 포함된 연결리스트의 크기

- 질의형식 : "L v"
  - L: 도시 v가 포함된 linked-list의 총 element의 개수에 대한 질의
  - v: 도시 번호
- 출력형식 : "S"
  - S: 도시 v가 포함된 linked-list의 총 element의 개수

(3) 중간단계: Kruskal 알고리즘에서 하나의 반복(iteration) 진행

-질의형식 : “I”

- I: Kruskal 알고리즘에서 하나의 반복을 실행하기 위한 질의

-출력형식 : “ID S” 또는 “not union” 또는 “ID S v D”

- ID: 해당 반복에서 union에 성공했다면, union 된 새로운 집합의 set id(leader)
- S: union 된 새로운 집합의 크기
- ✓ not union: 만약 해당 반복에서 두 집합에 대해 union을 실행할 경우 cycle이 발생하면, “not union”을 출력하여 예외처리를 한다.
- ❖ ID S v D: 만약 해당 반복에서 union에 성공하였고 MST를 찾았다면, "ID S v D"(ID, S, v, D는 각각 union 된 새로운 집합의 set id(leader), union 된 새로운 집합의 크기, 최종 완성된 MST 집합의 leader, MST의 모든 포장도로들의 길이의 합)를 출력하고 프로그램을 종료한다.

(4) 중간정보 출력: 두 도시가 같은 집합에 속하는지 확인

-질의형식: “F V1 V2”

- F: 도시 V1, V2가 같은 집합인지 물어보는 질의
- V1: 도시 번호 1
- V2: 도시 번호 2

-출력형식: “True s” 또는 “False v w”

- True s: 두 도시가 같은 집합일 경우, “True s”(s는 집합의 leader)을 출력
- ✓ False v w: 두 도시가 다른 집합일 경우, “False v w”(v, w는 각각 V1, V2를 포함하는 집합의 leader)를 출력

(5) 중간정보 출력: 임의의 도시를 포함하는 집합의 모든 포장도로의 길이

-질의형식: “W v”

- W: 도시 v를 포함하는 집합의 모든 포장도로들의 길이의 합에 대한 질의
- v: 노드 번호

-출력형식: “D”

- D: 도시 v를 포함하는 집합의 모든 포장도로들의 길이의 합

(6) 알고리즘의 나머지 단계 연속수행 및 프로그램 종료

-질의형식: “Q”

- Q: MST를 생성할 때까지 Kruskal 알고리즘을 진행하라는 질의

-출력형식: “v D”

- v: MST의 leader
- D: MST의 모든 포장도로들의 길이의 합

3. 입출력 제한사항

(1) 입력파일에서 도시정보는 최대 50,000개, 도로정보는 최대 500,000개, 질의는 최대 25,000개가 입력된다.

(2) 프로그램은 총 2초의 제한시간 이내에 수행되어야 한다.

(3) 제시한 입출력 형식대로 표준입출력을 사용하여 처리한다.

(4) 문제에서 설명되지 않은 예외처리를 해야 할 질의는 입력되지 않는다.

4. 프로그램 입출력 예

(1) 표준입출력 예시
파란색: 프로그램 표준입력 내용
빨간색: 프로그램 표준출력 내용

```
==========================================================
9 19 16
10 seoul 123
20 dajeon 325
40 newyork 951
30 tokyo 157
50 london 358
60 paris 675
70 kairo 6990
80 cicago 1254
90 sandiego 695
10 20 337
10 30 1404
20 30 1235
20 50 2342
30 50 1121
10 40 1846
40 30 802
50 60 946
50 70 1090
50 90 1258
60 70 184
70 80 144
70 90 187
10 90 2704
40 60 621
40 70 740
40 80 849
40 90 867
30 70 1391
I
```

70 2
I
70 3
I
70 4
F 60 80
True 70
F 90 10
False 70 10
N 90 2
60 paris
N 90 1
80 cicago
N 90 0
70 kairo
L 50
1
L 60
4
I
10 2
W 70
515
W 20
337
I
70 5
I
not union
Q
70 4456
(프로그램 종료)

(2) 파일 예시

input.txt
9 19 16 10 seoul 123 20 dajeon 325 40 newyork 951 30 tokyo 157 50 london 358 60 paris 675 70 kairo 6990 80 cicago 1254 90 sandiego 695 10 20 337

```
10 30 1404
20 30 1235
20 50 2342
30 50 1121
10 40 1846
40 30 802
50 60 946
50 70 1090
50 90 1258
60 70 184
70 80 144
70 90 187
10 90 2704
40 60 621
40 70 740
40 80 849
40 90 867
30 70 1391
I
I
I
F 60 80
F 90 10
N 90 2
N 90 1
N 90 0
L 50
L 60
I
W 70
W 20
I
I
Q
```

output.txt

```
70 2
70 3
70 4
True 70
False 70 10
60 paris
80 cicago
70 kairo
1
4
10 2
515
337
70 5
not union
70 4456
```

5. 주의 사항 (지키지 않으면 감점받을 수 있음)

(1) 채점 서버 환경

① OS : Ubuntu 18.04 (64bit)
② 허용된 개발 언어 : C, C++
③ gcc 버전 : gcc (Ubuntu 7.3.0-16ubuntu3) 7.3.0 (c++14 지원)

(2) 제출 파일

다음 파일들을 **"분반_학번_이름.zip"** 형식으로 압축합니다. (예 : 003_12059876_홍길동)
보고서 문서와 프로젝트의 이름도 **"분반_학번_이름"로 통일합니다.**

① **보고서**(I-class에 제출)
(a) 형식 : 아래아한글 문서(.hwp), MS Word 문서(.doc, .docx), PDF 문서(.pdf)
(b) 양식 : 첨부된 파일을 참조하세요.
(c) 파일명 : **"분반_학번_이름"로 통일**

② **소스 코드**(공지한 서버에 업로드)
(a) 하나의 소스코드 파일만으로 구현하여 서버에 업로드
(예: 003_12059876_HongGilDong.cpp) (소스코드 제출시, 이름은 반드시 **영어**로 표기)
(b) 소스코드에 반드시 주석이 기재되어 있어야 함.

(3) 기타

① 제출 마감은 **5월 27일 수요일 오후 11시 59분**입니다. **마감 후에 제출되는 과제는 받지 않습니다.**
② 마감 직전에는 학생들이 많이 몰리기 때문에 서버가 혼잡할 수 있습니다. 여유 있게 제출해 주세요.
③ 제출 후에는 제대로 제출되었는지 반드시 확인하시기 바랍니다.
④ 부정행위가 적발될 경우, 베낀 학생은 물론 **원본을 제공한 학생까지 0점이 아니라, 마이너스 점수가 부여됩니다.** 인터넷에 올라와 있는 소스코드나 참고서의 소스코드, 예전에 제출됐던 과제 등을 베껴서 제출해도 **부정행위로 처리됩니다.**