# Homework #4

# Due: 12/14

**HW4**

이번 과제는 Greedy Algorithm, Min-Cut Max-Flow Theorem 관련 5개의 코딩 문제로 구성되어 있습니다. **(Total 100 points)**

자동 채점으로 점수를 집계하기에, 사소한 실수로 정답이 오답 처리되는 안타까운 경우가 잦습니다.

아래 주의사항들과, 별도 첨부 파일 **CodingAssignmentGuidelines.pdf** 에 명시되어 있는 주의사항들을 꼭 숙지해주세요.

**Submission Format**

모든 제출 파일은 **학번_문제번호.py** 로 작성해주시고, 하나의 **HW4_학번.zip** 파일로 묶어서 제출해 주세요. 코드는 Python(.py extension)으로 작성하시기 바랍니다.

| |
|---|
| 파일 1: 학번_1.py |
| 파일 2: 학번_2.py |
| 파일 3: 학번_3.py |
| 파일 4: 학번_4.py |
| 파일 5: 학번_5.py |

→ **HW4_학번.zip** 압축 후 제출

(**파일명이 다르면 채점이 안됩니다!**)

**Recursion Limit**

Python 에서는 재귀호출의 default depth 를 1,000 번으로 제한하고 있습니다. 이로 인해, 재귀호출 횟수를 초과하여 에러가 발생할 수 있습니다. 본인의 코드를 재귀함수로 구현한다면, sys 모듈을 이용해 아래와 같이 depth 를 1,000,000 으로 넉넉하게 설정해주세요.

```
import sys
sys.setrecursionlimit(10**6)
```

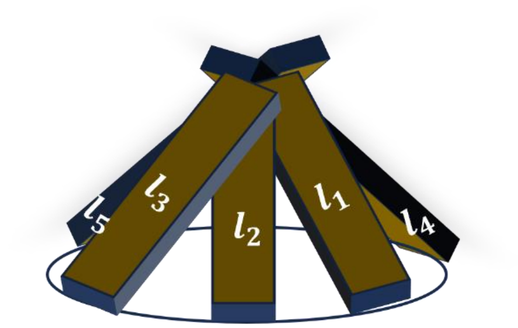
**Note That**

**H43 는, 사용 권장한 sys 모듈 외에는 다른 어떠한 import 도 허용하지 않습니다**. 이는 특정 자료구조에서 오는 시간적 이점을 최소화하고, 독립적인 알고리즘 설계를 장려하기 위한 목적입니다. 텍스트 검사로 **sys 외의 다른 import 가 발견되면 해당 문제는 0 점 처리**할 예정입니다.

1. **달구의 캠프파이어 (10 points)**

달구가 공터에서 캠프파이어를 하려고 한다. 다양한 길이의 m 개의 나무토막이 있고, 달구들은 모든 장작을 **원뿔형의 형태**로 쌓아 불을 붙이려 한다. **양 옆에 위치한 나무토막의 길이 차이 중 최댓값(A)**을 통해 나무토막이 얼마나 안정적으로 쌓였는지 알 수 있다. **A 의 값이 작으면 작을수록 쌓인 나무토막이 안정적**이다. 완벽한 캠프파이어를 위해 **모든 나무토막을 최대한 안정적**으로 쌓으려 한다. 이 때 A 의 값을 출력하는 프로그램을 작성하시오.

예시: 5 개의 장작이 주어졌을 때, 아래 그림과 같이 쌓는다. $l_2$ 의 길이를 가지는 나무토막 기준으로 양 옆에 위치한 장작은 $l_3, l_1$의 길이를 가지는 나무토막이다.




<u>조건:</u>

| |
|---|
| 모든 나무토막의 길이는 전부 다르다. |
| 입력으로 들어오는 나무토막의 길이는 정렬이 안 된 채로 들어온다. |

<u>Input Format</u>

| |
|---|
| 첫째 줄에 장작의 수 m 이 주어진다. ($3 \le m \le 100000$, m 은 정수) |
| 둘째 줄에 모든 장작들의 길이 $l_i$가 공백으로 구분되어 주어진다. ($1 \le l_i \le 100000$, $l_i$는 정수) |

<u>Output Format</u>

| |
|---|
| 첫째 줄에 A 의 값을 출력한다. |

Sample Input

```
7

15 2 10 7 12 4 1
```

Sample Output

```
6
```

Greedy Algorithm 추천 / timeout per a testcase = 0.87 s

## 2. **달구의 건축사업 (25 points)**

달구는 건축사업을 한다. 달구의 회사에는 B 명의 직원이 있다. 모든 직원은 시작지점(A)에서 동시에 출발한다. 이때, 50*50 크기의 지도를 보고 움직이는데, 지도에는 시작지점(A)와 집을 지어야 하는 위치(B), 장애물이 있는 위치(C)가 2 차원 좌표(x, y)로 주어진다. 모든 직원은 가로 또는 세로 방향으로 1 씩 움직일 수 있고, 경우에 따라 한 직원이 여러 집을 지을 수도 있고 아예 집을 짓지 않을 수도 있다. 지도가 주어졌을 때, **모든 직원의 최소 움직임으로 모든 B 에 집을 지으려 한다.** 이 때 아래 조건들을 만족하며 **모든 직원이 움직인 거리의 합**을 출력하는 프로그램을 작성하시오.

조건:

| |
|---|
| 집을 짓는 시간은 배제한다. |
| 모든 직원들은 상, 하 좌, 우로만 움직일 수 있다. (대각선으로는 움직이지 못한다) |
| 반드시 모든 직원이 움직일 필요는 없다. |
| 장애물이 있는 곳은 지나갈 수 없다. |
| 지도에 나와있는 영역 밖으로는 움직일 수 없다. (경계는 가능하다) |
| 지도는 (0,0), (N,0), (0,N), (N,N)을 꼭짓점으로 잇는 사각형이다. |
| 장애물에 의해 B 에 도달하지 못하는 경우는 없다고 생각한다. |

Input Format

| |
|---|
| 첫째 줄에 집을 지어야 하는 곳의 수 b, 장애물의 개수 c 가 공백으로 구분되어 주어진다. ($1 \le b \le 100$, $1 \le c \le 100$, b, c 는 정수) |
| 둘째 줄에 시작 지점(A)의 좌표 $(x_A, y_A)$가 공백으로 구분되어 주어진다. |
| 셋째 줄부터 b+2 번째 줄까지 각 줄에 집을 지어야 하는 위치(B)의 좌표 $(x_B, y_B)$가 공백으로 구분되어 주어진다. |
| b+3 번째 줄부터 b+c+2 번째 줄까지 각 줄에 장애물이 있는 위치(C)의 좌표 $(x_C, y_C)$가 공백으로 구분되어 주어진다. (각 좌표들은 지도를 벗어나지 않는다.) |

Output Format

모든 B 에 집을 다 지었을 때, **모든 직원들이 움직인 거리의 합**을 출력한다.

Sample Input

```
4 5
4 3
0 1
1 1
3 7
7 2
2 0
2 1
3 6
4 7
5 3
```

Sample Output

```
17
```

Greedy Algorithm 추천 / timeout per a testcase = 12.59 s

### 3. **달구의 대피 (10 points)**

대구에 재난 상황이 발생해 디지스트에 있는 모든 달구가 학교 버스를 타고 대구 밖의 대피소에 모이려 한다. 대구에는 여러 개의 노드들이 있으며 각 노드들은 도로를 통해 연결되어 있다. **각 도로들은 최대로 수용할 수 있는 버스의 수가 정해져 있으며, 최대한 많은 달구들을 대피시키기 위한 최적의 경로**를 계획해야 한다. 이 때 아래 주어진 조건에 조건들을 만족하며 **대피소에 도착할 수 있는 최대 버스 수를** 출력하는 프로그램을 작성하시오.

조건:

디지스트는 0, 대피소는 n-1 의 노드 번호를 가진다.

학교 버스는 모두 같은 도로로 움직인다.

주어지는 도로의 방향은 일방통행이다.

특정 두 노드 사이의 도로는 0~2 개다.

디지스트에서 대피소로 바로 이어지는 도로는 없다고 생각한다.

디지스트에서 대피소로 갈 수 없는 경우는 없다고 생각한다.

<u>Input Format</u>

첫째 줄에 구역의 수 n 과 도로의 수 m 이 공백으로 구분되어 주어진다. ($3 \le n \le 250$, $2 \le m \le 500$, n, m 은 정수)

둘째 줄부터 m+1 번째 줄까지 각 도로의 정보가 공백으로 구분되어 주어진다. 도로 정보는 세 개의 정수로 구성되어 있으며, 이는 도로로 연결된 두 구역 시작 구역 $a_i$, 종료 구역 $b_j$, 해당 도로의 차량 용량 $c_i$을 나타낸다. ($1 \le c_i \le 1000$, $c_i$는 정수)

<u>Output Format</u>

첫째 줄에 대피소에 도착할 수 있는 최대 버스 수를 출력한다.

Sample Input

```
5 5
0 1 3
3 4 4
1 2 3
2 3 5
2 4 6
```

Sample Output

```
3
```

Min-Cut Max-Flow Theorem 추천 / timeout per a testcase = 0.36 s

4. **달구의 경찰과 도둑 (25 points)**

달구가 경찰과 도둑 놀이를 하고 있다. 경찰팀은 도둑이 시작지점부터 종료지점에 도착할 때까지 도둑을 잡아야 승리한다. 총 n 개의 거점이 존재하고, 시작지점과 종료지점도 거점에 포함된다. 도둑의 이동 경로는 거점과 거점끼리 연결된 길을 통해 움직일 수 있다. 경찰팀은 도둑을 확실하게 잡기 위해 전략을 세웠다. 전략은 **도둑이 종료지점에 도착하려면 경찰이 배치된 도로를 반드시 지나가게끔 특정 도로에 인원을 배치**하는 것이다. 다만, 각 **도로의 폭에 따라서 그 도로를 막으려면 특정 수 이상의 인원을 배치해야 한다**. 이때 **전략에 맞게 최소로 인원을 배치할 때, 총 몇 명의 인원이 배치**되는 지 출력하는 프로그램을 작성하시오.

조건:

시작지점은 0 번 node 번호를, 종료지점은 n-1 번의 node 번호를 가진다.

모든 도로는 일방통행이다.

특정 두 노드 사이의 도로는 0~2 개다.

시작 지점에서 종료 지점으로 바로 연결되는 길은 없다.

시작지점에서 종료지점으로 도착하지 못하는 경우는 없다.

Input Format

첫째 줄에 거점의 수 n 과 도로의 수 m 이 공백으로 구분되어 주어진다. ($3 \le n \le 100$, $2 \le m \le 2{,}000$, n, m 은 정수)

둘째 줄부터 m+1 번째 줄까지 도로의 정보가 공백으로 구분되어 주어진다. 도로 정보는 3 개의 정수로 구성되어 있으며, 이는 도로로 연결된 두 거점 시작 거점 $a_i$, 종료 거점 $b_j$, 도로에 배치해야 하는 최소 경찰의 인원 수 $c_i$가 순서대로 주어진다. ($1 \le c_i \le 100{,}000$, $c_i$는 정수)

Output Format

최소로 경찰을 배치했을 때, 총 배치된 인원 수를 출력한다.

Sample Input

```
5 6
0 1 2
2 4 2
3 1 20
0 3 15
2 3 13
0 2 19
```

Sample Output

```
2
```

Min-Cut Max-Flow Theorem 추천 / timeout per a testcase = 0.43 s

5. **달구의 체력단련 (25 points)**

달구는 최근 체력을 증진시키기 위해 학교와 기숙사를 왕복하며 뛴다. 러닝루트를 n 개의 노드로 구성된 undirected unweighted graph 로 매핑할 수 있다. 학교와 기숙사도 노드다. 매번 같은 경로를 뛰는 게 지겨웠던 달구는 **한번 방문한 노드는 다시 가지 않는다**. 이 때 주어진 조건을 만족하며 **학교에서 기숙사를 최대한 많이 왕복할 수 있는 횟수**를 출력하는 프로그램을 작성하시오.

조건:

학교는 0, 기숙사는 n-1 의 노드 번호를 가진다.

학교 노드와 기숙사 노드를 직접 연결하는 길은 없다.

학교에서 기숙사 또는 기숙사에서 학교로 이동하는 편도 한번을 횟수 한 번으로 생각한다.

모든 길은 양방향 통행이 가능하다.

특정 두 노드 사이의 길은 0~1 개다.

학교와 기숙사는 여러 번 방문 가능하다.

<u>Input Format</u>

첫째 줄에 학교, 기숙사와 모든 쉼터의 개수의 합 n 과, 각 노드들과 학교, 기숙사가 연결된 길의 개수 m 이 공백으로 구분되어 주어진다. ($3 \le n \le 300$, $2 \le m \le 10{,}000$, n, m 은 정수)

둘째 줄부터 m+1 번째 줄까지 각 줄에 길의 시작 지점과 종료 지점이 공백으로 구분되어 주어진다. (두 지점은 0~n-1 의 정수)

<u>Output Format</u>

첫째 줄에 학교와 기숙사를 오간 최대 횟수를 출력한다.

Sample Input

```
4 5
2 3
0 1
1 2
0 2
1 3
```

Sample Output

```
2
```

Min-Cut Max-Flow Theorem 추천 / timeout per a testcase = 3.36 s