[Python 트랙] 2회차 - 알고리즘

| Background

- ✓ 배열에 대한 이해와 활용
- ✓ 트리 순회에 대한 이해와 활용

| Goal

- ✓ 반복문을 이용하여 배열의 요소에 접근할 수 있다.
- ✓ 필요한 조건에 따라 트리를 순회할 수 있다.
- ✓ 문제의 조건을 정확히 이해하고 해결할 수 있다.

| 환경 설정

1) Pycharm(Python3.7이상)을 이용해서 코드를 작성하고 결과를 확인한다.
   - 새로운 Pycharm 프로젝트를 생성 후 코드를 작성한다.

2) 파일 이름 및 제출 방법
   - 1, 2번 문제에 대한 소스 파일은 Algo문제번호_지역_반_이름.py로 만든다.
   - pypy의 경우 프로젝트와 파일이름에 한글을 사용할 수 없으므로 algo1.py, algo2.py 로 만들고 제출시 아래와 같이 변경한다.
   - 3번은 텍스트 파일로 작성한다.

     Algo1_서울_1반_이싸피.py
     Algo2_서울_1반_이싸피.py
     Algo3_서울_1반_이싸피.txt

   - 위 3개의 파일만 지역_반_이름.zip으로 압축하여 제출한다.

     서울_1반_이싸피.zip

   (탐색기에서 파일 선택 후 오른쪽 클릭 – 보내기 – 압축(zip)폴더 선택)

3) 채점
   - 주석이 없는 경우, 주석이 코드 내용과 맞지 않는 경우, 지정된 출력 형식을 만족하지 않는 경우 해당 문제는 0점 처리될 수 있다.
   - import를 사용한 경우 해당 문제는 0점 처리될 수 있다. (import sys도 예외 없음)

4) 테스트케이스는 부분적으로 제공되며, 전체가 공개되지는 않는다.

5) 각 문제의 배점이 다르므로 표기된 배점을 반드시 확인한다.
   - 1번 50점, 2번 30점, 3번 20점

성실과 신뢰로 테스트에 볼 것 (부정 행위시 강력 조치 및 근거가 남음)

※ 소스코드 유사도 판단 프로그램 기준 부정 행위로 판단될 시,
0점 처리 및 학사 기준에 의거 조치 실시 예정

[Python 트랙] 2회차 - 알고리즘

| 문제 1 : 정원에 나무심기 (배점 50점)

김싸피는 집의 정원에 나무를 심으려고 한다. 정원은 사각형 모양이며 나무는 세로 줄로 심을 예정이다. 정원의 가장 왼쪽 세로 줄부터 나무를 심어야 하며 앞으로 나무가 커질 것을 고려하여 한 줄씩 띄어 심으려고 한다. 각 위치에 심을 나무의 가격은 알고 있다고 가정할 때 정원에 나무를 심기 위한 총 비용과 심은 나무의 수를 구해보자. 또한 심은 나무 중 가장 비싼 나무의 가격과 해당 나무의 열 번호를 계산해보자. 만약 가장 비싼 나무가 여러 개 심어져 있는 경우 가장 큰 열의 번호를 계산한다.

예를 들어 정원의 크기가 3 X 3이고 각 위치에 심을 나무의 가격이 다음 그림과 같다면 나무를 심는 총 비용은 5+7+4+3+8+5 = 32가 되고 심은 나무의 수는 6이 된다. 또한 가장 비싼 나무의 가격은 8이며 해당 나무가 심어진 열은 3이 된다.

5	2	3
7	1	8
4	6	5

또 다른 예를 들어 보면, 다음 그림과 같이 정원 크기가 4 X 5 인 경우 나무를 심는 총 비용은 3+9+7+5+4+7+2+8+8+5+9+6 = 73이 되고 심은 나무의 수는 12가 된다. 또한 가장 비싼 나무의 가격은 9이며 해당 나무는 1열과 5열에 심어져 있지만 열 번호가 더 큰 5가 계산된다.

3	2	4	1	8
9	5	7	6	5
7	3	2	5	9
5	1	8	4	6

정원 크기인 N X M 리스트가 주어질 때 나무를 심는 총 비용과 심은 나무의 수, 심은 나무 중 가장 비싼 나무의 가격, 가장 비싼 나무가 심어진 열을 출력하는 프로그램을 만드시오.

[Python 트랙] 2회차 - 알고리즘

[입력]

첫 줄에는 테스트케이스 개수 T가 주어진다.(1<= T <=10)
다음 줄부터 테스트 케이스의 첫 줄엔 정원의 크기인 행의 개수 N과 열의 개수 M이 주어진다.(3 <= N <= 20, 3 <= M <= 20)

그 다음 줄부터는 정원 영역이 N 줄에 걸쳐 각 행 별로 M개의 자연수가 공백으로 구분되어 주어진다. 주어지는 자연수는 100이하이다.

[출력]

각 줄에 #과 1부터인 테스트케이스번호를 출력하고 나무를 심는 총 비용, 심은 나무의 수, 가장 비싼 나무의 가격, 가장 비싼 나무가 심어진 열을 빈칸으로 구분하여 출력하시오

[입력 예시]

```
3
3 3
5 2 3
7 1 8
4 6 5
4 5
3 2 4 1 8
9 5 7 6 5
7 3 2 5 9
5 1 8 4 6
5 6
40 15 33 52 12 37
70 23 73 12 34 54
23 10 37 15 45 80
50 60 48 24 19 20
55 28 32 66 80 42
```

(algo1_sample_in.txt 참고)

[출력 예시]

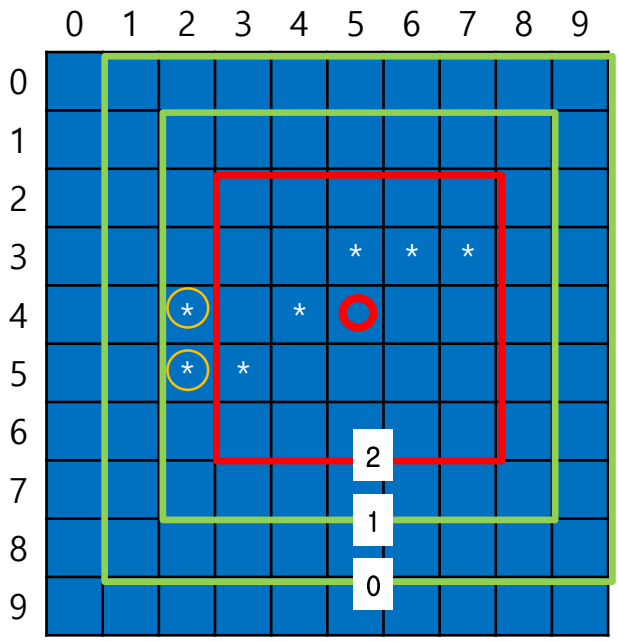
```
#1 32 6 8 3
#2 73 12 9 5
#3 651 15 80 5
```

(algo1_sample_out.txt 참고)

[Python 트랙] 2회차 - 알고리즘

| 문제 2 : 별자리 사진 찍기 (배점 30점)

김싸피는 최근 은하23 스마트폰을 구매하였다. 김싸피는 구매한 스마트폰이 별 사진을 잘 찍을 수 있다는 이야기를 들었고 별자리 사진을 찍으려고 한다. 김싸피는 별자리 사진을 최대한 크게 찍기 위해서 확대 기능을 사용하려고 한다. 별자리가 빠짐없이 나오면서 최대 크기로 별자리 사진을 찍기 위해서 몇 번 확대해야 하는지 출력하는 프로그램을 작성하시오.
(확대 시 별자리 비율은 커지지만 촬영 영역은 작아진다.)



- 김싸피가 바라보고 있는 **하늘의 전체 크기는 N*N** 이며, 하늘에는 단 하나의 별자리만 존재 하고 최소 1개의 별로 구성되어 있다. 확대하지 않고 사진을 찍을 경우 찍을 수 있는 영역은 **초점(A, B)을 중심**으로 하며 **영역의 크기는 K*K** 이다.
  (1 ≤ K ≤ N ≤ 20, 단, K는 홀수)
- 한 번 확대 할 때마다 촬영 가능영역의 한 변 **길이는 2씩 줄어 든다**. 위 예시에서 K=9 이고 한 번 확대 시 영역의 크기는 7*7, 두 번 확대 시, 영역의 크기는 5*5 이다. 영역의 최소 크기는 1*1이다.
- 위 예시는 N= 10, K = 9, 초점(A,B) = (4,5) 일 때의 예시이다. 확대 하지 않았을 때, 모든 별을 찍을 수 있지만, 1번 확대 하였을 경우에도 마찬가지로 모든 별을 찍을 수 있으니 **최대 확대 횟수인 1을 출력**한다. 2번 확대하는 경우 (4,2) 와 (5,2)에 위치한 별이 촬영영역에 포함되지 않으므로 조건에 맞지 않다.

[Python 트랙] 2회차 - 알고리즘

[입력]

첫 줄에 테스트케이스 수가 주어진다.

각 테스트 케이스의 첫 줄에 N, K, A, B가 띄어쓰기로 구분되어 주어진다. 이후 N개의 줄에 N개의 문자가 띄어쓰기로 구분되어 주어진다. 이 때 '.'은 빈 하늘, '*'은 별을 의미한다.

(1 ≤ K ≤ N ≤ 20, 단, K는 홀수 , 0 ≤ A, B < N)

[출력]

각 줄마다 " #T " (T는 테스트 케이스 번호)를 출력한 뒤, 최대 확대 횟수를 출력한다.

단, 주어진 조건에서 모든 별을 촬영할 수 없을 경우 -1을 출력한다.

[입력 예시]

```
3
10 9 4 5
..........
..........
..........
.....***..
..*.*.....
..**......
..........
..........
..........
..........
10 9 5 5
..........
.*...*....
..........
..***.....
..........
..*.......
.*...*....
..........
..........
..........
8 7 5 4
.........
....*....
.........
..*...*..
.....*...
.........
....*....
.........
```

[출력 예시]

```
#1 1
#2 0
#3 -1
```

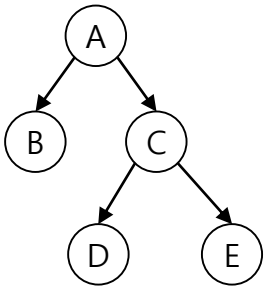
[Python 트랙] 2회차 - 알고리즘

| 문제 3 : 순회 (배점 : 20점)

(1) 이진 트리를 중위 순회하며, 방문한 정점의 이름을 출력(print(v))하는 유사(pseudo) 코드를 완성하시오. 방문한 정점과 자식 정점에 대한 설명이 필요한 경우, v와 v의 왼쪽 자식 정점(또는 v.left), v의 오른쪽 자식 정점(또는 v.right)으로 표현하고, 각 행에 주석 형식으로 간단한 설명을 추가해야 한다.

```
def inoder(v):
```

(2) 다음의 트리를 정점 A부터 전위순회하는 경우, 정점의 방문순서를 (예)와 같이 나타낼 수 있다. A부터 중위순회, 후위순회하는 경우의 방문 순서를 각각 같은 방식으로 표시하시오.



예) 전위 순회 A B C D E