[Python 트랙] 3회차 과목평가 – 알고리즘

| Background

- ✓ 리스트에 대한 이해와 사용
- ✓ 반복문에 대한 이해와 사용
- ✓ 이진 탐색에 대한 이해와 사용

| Goal

- ✓ 반복문을 이용하여 리스트의 요소에 접근할 수 있다.
- ✓ 이진 탐색을 활용할 수 있다.

| 환경 설정

1) Pycharm(Pypy3.7 또는 Python3.7)을 이용해서 코드를 작성하고 결과를 확인한다.

- **새로운 Pycharm 프로젝트를 생성 후 코드를 작성한다.**

2) 파일 이름 및 제출 방법

- 소스코드는 pypy에서의 한글 오류 방지를 위해 Algo1.py, Algo2.py로 작성한다.

- 디버깅이 끝나면 제출전에 파일이름을 다음 형식으로 바꾼다.

Algo1_서울_1반_이싸피.py

Algo2_서울_1반_이싸피.py

- 3번은 텍스트 파일로 작성한다.

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)

(edu.ssafy.com 사이트에 업로드)

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] 3회차 과목평가 – 알고리즘

| 문제 1 : 최대값은 무엇?(배점 50점)

가로, 세로 길이가 N인 2차원 리스트가 있다.
각 행에서 길이 K 인 특정 구간의 합을 구하려고 한다. 이 때, 모든 행의 구간 합 중 최대 값을 출력하는 프로그램을 작성하라.

[제약사항]
파이썬 내장함수 max, min, sum은 사용하지 않는다.
각 행에서의 구간이란, R (행 인덱스) ≤ 열의 인덱스 ≤ R + K – 1 인 구간을 의미한다. 단, **구간의 범위가 리스트의 범위를 벗어나는 경우, 리스트의 마지막 인덱스 까지만 계산한다.**

아래는 N = 5 인 2차원 리스트의 각 행에서 길이 K = 3인 구간을 나타낸 그림이다.

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

예) 구간의 길이 K 가 3일 때,

0 행의 구간 인덱스 : 0~2 , 구간 합 : 6
1 행의 구간 인덱스 : 1~3 , 구간 합 : 9
2 행의 구간 인덱스 : 2~4 , 구간 합 : 12
3 행의 구간 인덱스 : 3~4 , 구간 합 : 9
4 행의 구간 인덱스 : 4 , 구간 합 : 5

위 예시에서 최대 구간합은 2행의 12이다. 따라서 12을 출력한다.

[Python 트랙] 3회차 과목평가 – 알고리즘

[입력]

첫 줄에 테스트케이스 수가 주어진다.

각 테스트 케이스의 첫 줄에 N, K 가 띄어쓰기로 구분되어 주어진다.

이후 N 줄에 거쳐 N개의 정수 A_i가 주어진다.

(2 ≤ N ≤ 20 , 1 ≤ K ≤ N , 0 ≤ A_i ≤ 100)

[출력]

각 줄마다 " #T " (T는 테스트 케이스 번호)를 출력한 뒤, 구간합의 최대값을 출력한다.

[입력 예시]

```
3
5 3
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
5 3
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
5 5
5 3 2 2 1
5 3 2 2 1
5 3 2 2 1
5 3 2 2 1
5 3 2 2 100
```

[출력 예시]

```
#1 12
#2 3
#3 100
```

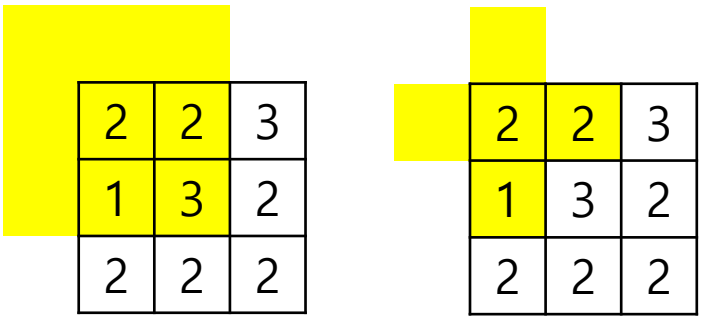
[Python 트랙] 3회차 과목평가 – 알고리즘

| 문제 2 : 영역의 합 (배점 30점)

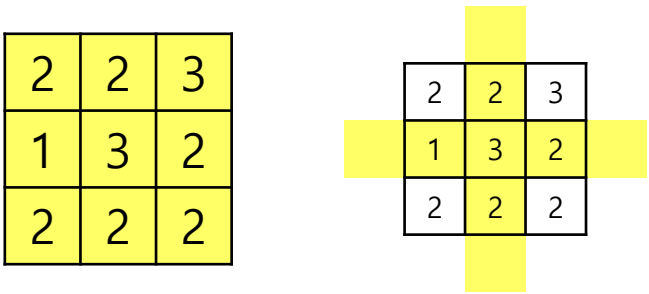
10이하 자연수가 들어있는 NxN 크기의 2차원 배열이 있다.
모든 원소를 중심으로 한 3x3 배열의 합과 4방향 원소에 대한 합 중 가장 큰 값을 찾는 프로그램을 만드시오. **단, 다음 조건에 따라 합을 구해야 한다.**

(1) 3x3 배열 일부가 배열을 벗어나는 경우는 3x3 배열의 합은 0이 된다.
(2) 배열 A의 원소 A[i][j]의 4방향 원소는, A[i][j]부터 한 방향에 A[i][j]개 씩의 원소이다. 원소의 합은 배열을 벗어나지 않는 원소에 대해서만 계산한다.

다음은 4x4 배열의 예이다.
A[0][0]의 경우, [0][0]을 중심으로 한 3x3 배열은 일부가 주어진 배열을 벗어나므로 합은 0이다. 4방향 원소는 A[0][0]를 포함해 오른쪽으로 2개, 아래로 2개, 왼쪽으로 2개, 위로 2개의 원소를 말한다. 이 경우 영역을 벗어나지 않는 2+2+1만 계산한다.



A[1][1]의 경우 3x3의 합은 16, 4방향 합은 10으로 계산된다.



모든 원소에 대해 같은 작업을 반복하여 이 중 최대값을 출력한다.

[Python 트랙] 3회차 과목평가 – 알고리즘

[제약사항]

내장함수 max()를 사용하지 않고 코드를 작성해야 한다.

[입력]

첫 줄에 테스트케이스의 개수 T가 주어진다. 다음 줄부터 케이스 별로, 첫 줄에 N, N개씩 N개의 줄에 걸쳐 10이하의 자연수가 주어진다.

1<=N<=20

[출력]

#과 1번 부터인 테스트케이스 번호, 빈 칸에 이어 답을 출력한다.

[입력 예시]

```
3
1
1
3
2 2 2
2 2 2
2 2 2
3
2 2 3
1 9 2
2 2 2
```

[출력 예시]

```
#1 1
#2 18
#3 25
```

[Python 트랙] 3회차 과목평가 – 알고리즘

| 문제 3 : 이진검색(배점 20점)

다음 코드의 빈칸에 들어갈 내용을 적고(1~5), 이 알고리즘의 명칭과 특징, 동작에 대해 설명 하시오.

(단, 다음 검색구간 설정 시 middle을 제외해야 한다.)

```
def f(a, N, key) # 배열, 배열의 크기, 키값
    start = 0
    end = N-1
    while   (1)   :
        middle =    (2)
        if   (3)   :          # 검색 성공
            return middle
        elif a[middle] > key :
          end =   (4)
        else :
          start =    (5)
    return -1                  # 검색 실패
```