

[20191216] <환경설정>

#### 1. idle을 사용하는 경우

1) 입력 없이 사용하는 경우: idle 실행 > New File > AA.py로 작성 후 저장 > 실행

2) File에 입력을 넣는 경우: AA.py가 있는 폴더에서 > input.txt로 입력 값을 넣고

> AA.py에

Import sys

sys.stdin=open("input.txt", "rt") 추가 > AA.py 실행

#### 2. Visual Studio 2019를 사용하는 경우

New File > AA.py로 작성 후 > 오른쪽 마우스 클릭 > Add > New Item > input.txt로 입력 작성

> AA.py에

Import sys

sys.stdin=open("input.txt", "rt") 추가 > AA.py > ctrl + F5 > 해당 폴더에 복사 붙여넣기 후 실행

### # Section 4

#### 1. 가장 큰 수(스택)

=> 왜 append가 먼저 오면 안되나

=> list를 int형으로 만들 때는 str로 먼저 변환

=> list를 다시 str로 만들 때고 str로 변환

선생님은 현수에게 숫자 하나를 주고, 해당 숫자의 자릿수들 중 m개의 숫자를 제거하여 가장 큰 수를 만들라고 했습니다. 여러분이 현수를 도와주세요.(단 숫자의 순서는유지해야 합니다)

만약 5276823 이 주어지고 3개의 자릿수를 제거한다면 7823이 가장 큰 숫자가 됩니다.

1) 앞에 있는 숫자가 자기보다 작으면, 앞에 있는 숫자를 지우고(Pop) 앞으로 전진(Append)

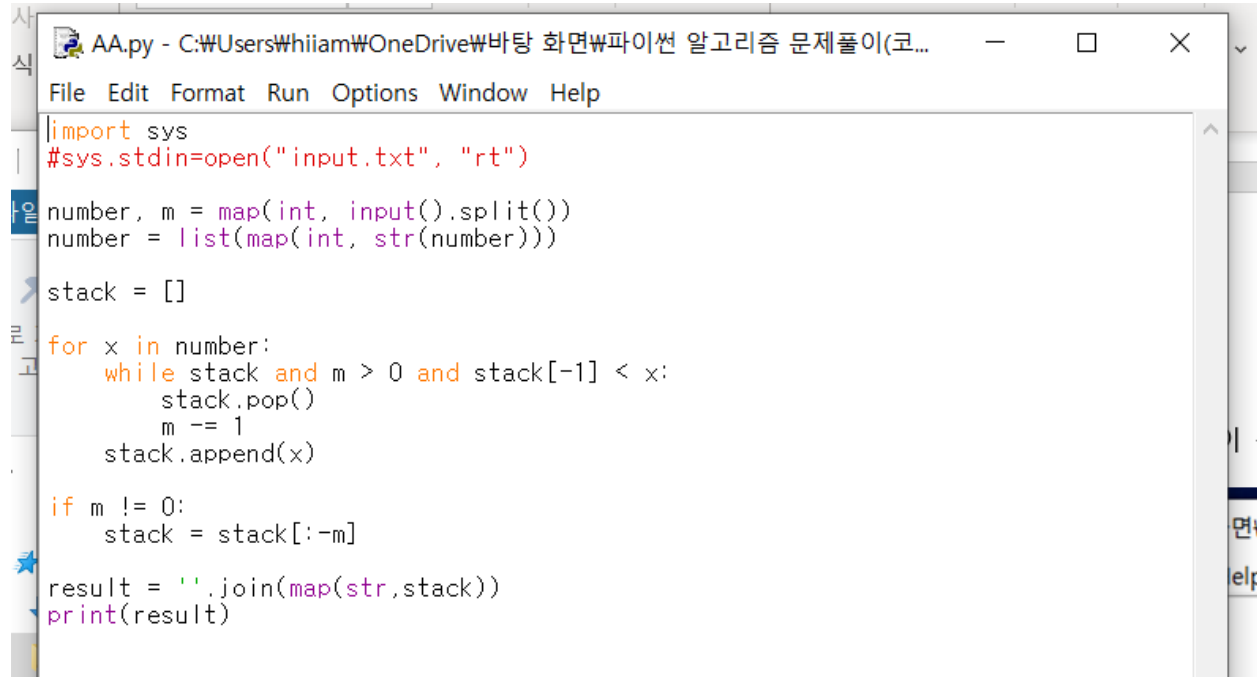
2) 지우는 숫자가 m개가 되면(m—로 count) 지울 수 없으므로 그대로 써줌

3) m에서 -하여 지우는 숫자를 count했는데 0이 안 되었을 때에는, stack[:m] 부분 까지만 출력

\*스택(Stack): Last In First Out, 가장 나중에 입력된 것이 가장 먼저 출력되는 구조

-list를 가지고 append해서 입력하고, pop해서 빼면 Stack 구조가 완성됨

-가장 먼저 들어간 것이 Stack의 Top 부분이 됨



```
import sys
#sys.stdin=open("input.txt", "rt")

number, m = map(int, input().split())
number = list(map(int, str(number)))

stack = []

for x in number:
    while stack and m > 0 and stack[-1] < x:
        stack.pop()
        m -= 1
    stack.append(x)

if m != 0:
    stack = stack[:m]

result = ''.join(map(str, stack))
print(result)
```

1) string으로 된 number를 list로 만듦

2) stack이 비어있지 않고, m이 양수이고, stack에 최근 들어간 숫자가 x보다 작으면 stack에서 pop 시 키고, m-1로 count해줌. 거짓이면 그렇지 않으면 stack에 append 시킴

3) for문이 끝난 후 만일 m이 0이 아니라면, stack[:m]까지만 stack으로 정의

4) join함수를 통해 다시 str로 바꾼 후 그 결과를 print

## 2. 쇠막대기(스택)

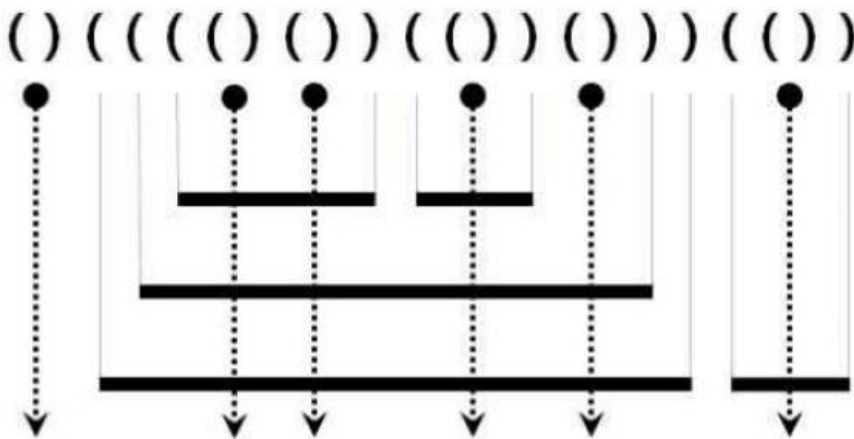
=> 선생님 풀이 다시 듣기

여러 개의 쇠막대기를 레이저로 절단하려고 한다. 효율적인 작업을 위해서 쇠막대기를 아래에 서 위로 겹쳐 놓고, 레이저를 위에서 수직으로 발사하여 쇠막대기들을 자른다. 쇠막대기와 레

이저의 배치는 다음 조건을 만족한다.

- 쇠막대기는 자신보다 긴 쇠막대기 위에만 놓일 수 있다. - 쇠막대기를 다른 쇠막대기 위에 놓는 경우 완전히 포함되도록 놓되, 끝점은 겹치지 않도록 놓는다.
- 각 쇠막대기를 자르는 레이저는 적어도 하나 존재한다.
- 레이저는 어떤 쇠막대기의 양 끝점과도 겹치지 않는다.

아래 그림은 위 조건을 만족하는 예를 보여준다. 수평으로 그려진 굵은 실선은 쇠막대기이고, 점은 레이저의 위치, 수직으로 그려진 점선 화살표는 레이저의 발사 방향이다.



이러한 레이저와 쇠막대기의 배치는 다음과 같이 괄호를 이용하여 왼쪽부터 순서대로 표현할 수 있다.

1. 레이저는 여는 괄호와 닫는 괄호의 인접한 쌍 '()' 으로 표현된다. 또한, 모든 '()' 는 반드시 레이저를 표현한다.
2. 쇠막대기의 왼쪽 끝은 여는 괄호 '(' 로, 오른쪽 끝은 닫힌 괄호 ')' 로 표현된다.

위 예의 괄호 표현은 그림 위에 주어져 있다.

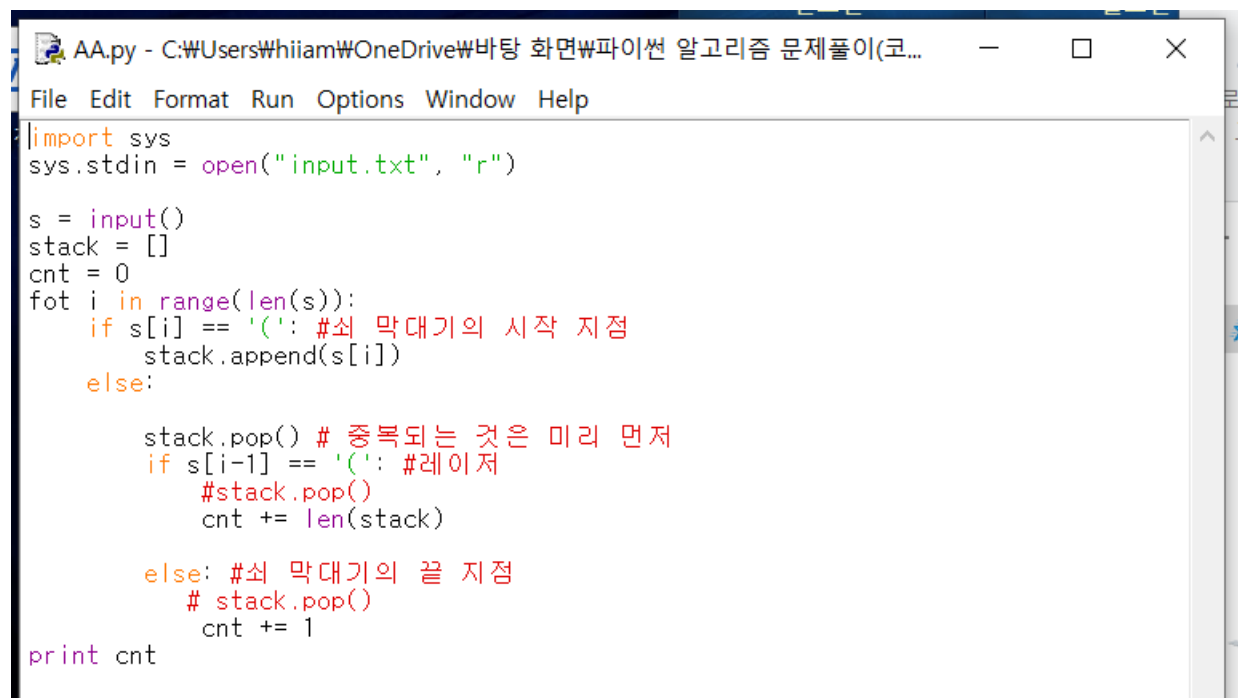
쇠막대기는 레이저에 의해 몇 개의 조각으로 잘려지는데, 위 예에서 가장 위에 있는 두 개의

쇠막대기는 각각 3개와 2개의 조각으로 잘려지고, 이와 같은 방식으로 주어진 쇠막대기들은 총 17개의 조각으로 잘려진다.

쇠막대기와 레이저의 배치를 나타내는 괄호 표현이 주어졌을 때, 잘려진 쇠막대기 조각의 총 개수를 구하는 프로그램을 작성하시오.

- 1) 여는 괄호는 stack에 append
- 2) 닫는 괄호를 만났을 때 앞에 것이 여는 괄호라면 레이저이므로, pop 하고, stack에 남아있는 여는 괄호의 개수를 sum에 누적시킨다.
- 3) 닫는 괄호를 만났을 때 앞의 것이 닫는 괄호라면 pop하고, sum에+1을 해준다.

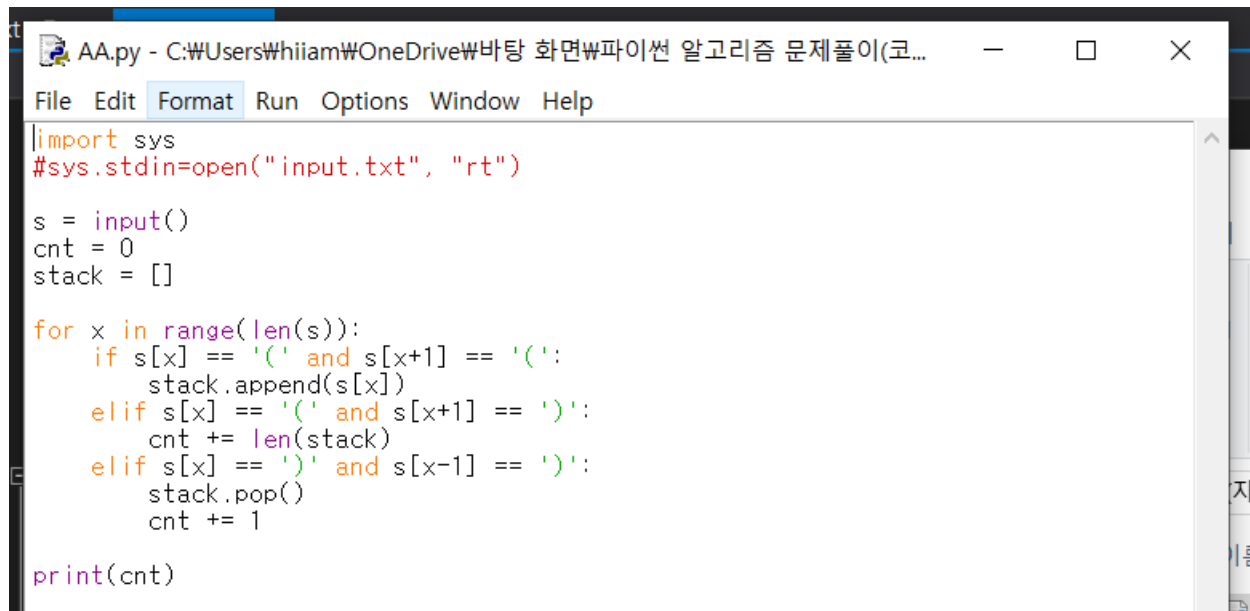
-선생님풀이



```
import sys
sys.stdin = open("input.txt", "r")

s = input()
stack = []
cnt = 0
for i in range(len(s)):
    if s[i] == '(': #쇠 막대기의 시작 지점
        stack.append(s[i])
    else:
        stack.pop() # 중복되는 것은 미리 먼저
        if s[i-1] == '(': #레이저
            #stack.pop()
            cnt += len(stack)
        else: #쇠 막대기의 끝 지점
            # stack.pop()
            cnt += 1
print cnt
```

-내풀이



```
AA.py - C:\Users\Whiam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코...
File Edit Format Run Options Window Help
|import sys
|#sys.stdin=open("input.txt", "rt")

s = input()
cnt = 0
stack = []

for x in range(len(s)):
    if s[x] == '(' and s[x+1] == '(':
        stack.append(s[x])
    elif s[x] == '(' and s[x+1] == ')':
        cnt += len(stack)
    elif s[x] == ')' and s[x-1] == ')':
        stack.pop()
        cnt += 1

print(cnt)
```

### 3. 후위표기식(스택)

\* 컴퓨터는 앞에서부터 순차처리 한다. 그런데, 중위표기식을 사용하면 컴퓨터가 연산 우선순위에 따라 처리하기 어려우므로 후위표기식으로 표현해준다.

\* 연산자 우선순위: \* / + -

\* 괄호 사이의 연산자는 우선순위를 만나면 pop하고 닫는 괄호를 만나면 pop 한다.

-선생님 풀이

```
AA.py - C:/Users/hiiam/OneDrive/바탕 화면/파이썬 알고리즘 문제풀이(코딩테...
File Edit Format Run Options Window Help

import sys
sys.stdin = open("input.txt", "r")

a = input()
stack = []
result = ''

for x in a:
    if x.isdecimal(): #피연산자 확인
        result += x
    else:
        if x == '(':
            stack.append(x)
        elif ( x == '*' ) or ( x == '/' ):
            while stack and (stack[-1] == '*' or stack[-1] == '/'):
                result += stack.pop()
            stack.append(x)
        elif ( x == '+' ) or ( x == '-' ): #괄호 안의 +와 -
            while stack and stack[-1] != '(':
                result += stack.pop()
            stack.append(x)
        elif ( x == ')' ):
            while stack and stack[-1] != '(': #여는 괄호 없앴
                result += stack.pop()
            stack.pop()

while stack: #stack에 남아있는 것들을 print
    result += stack.pop()
```

=> 왜 덧셈 뺄셈에서, stack[-1]이 +,-,/\*일 때를 고려 안 하는지 모르겠고 왜 +과 -에서 괄호를 조건으로 하는지 모르겠음

-내풀이

```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코딩테스트 대비)\섹션 4(자료구조 활용)\3. 후위표기식 만...
File Edit Format Run Options Window Help
import sys
#sys.stdin=open("input.txt", "rt")

#s = sys.stdin.readline()
s = input()
stack = []
result = ''

for x in s:
    if x.isdecimal():
        result += x
    else:
        if (x == '*' or x == '/'):
            while stack and (stack[-1] == '*' or stack[-1] == '/'):
                result += stack.pop()
            stack.append(x)

        elif (x == '+' or x == '-'):
            while stack and ((stack[-1] == '+' or stack[-1] == '-') or (stack[-1] == '*' or stack[-1] == '/')):
                result += stack.pop()
            stack.append(x)

        elif (x == '('):
            stack.append(x)

        elif (x == ')'):
            while stack and (stack[-1] != '('):
                result += stack.pop()
            stack.pop()

while stack:
    result += stack.pop()

print(result)
```

#### 4. 후위연산(스택)

후위연산식이 주어지면 연산한 결과를 출력하는 프로그램을 작성하세요.

만약  $3*(5+2)-9$  을 후위연산식으로 표현하면  $352+*9-$  로 표현되며 그 결과는 21입니다.

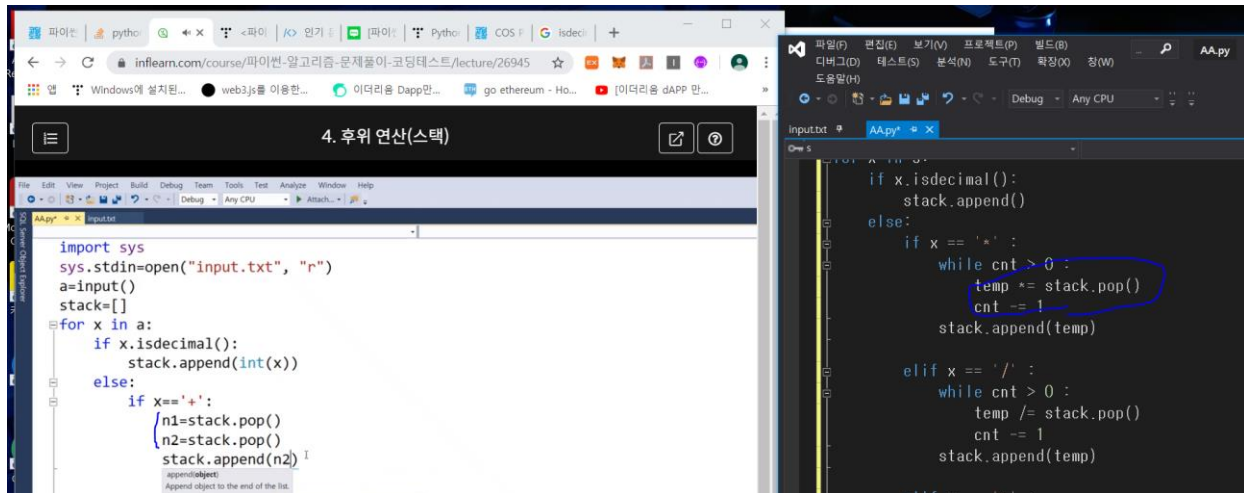
- 1) 연산자 앞 두 개의 피연산자를 계산하면서 결과를 냄
- 2) 두 개의 피연산자 중 무조건 앞의 피연산자가 연산을 당하는 것

=>스택으로 표시

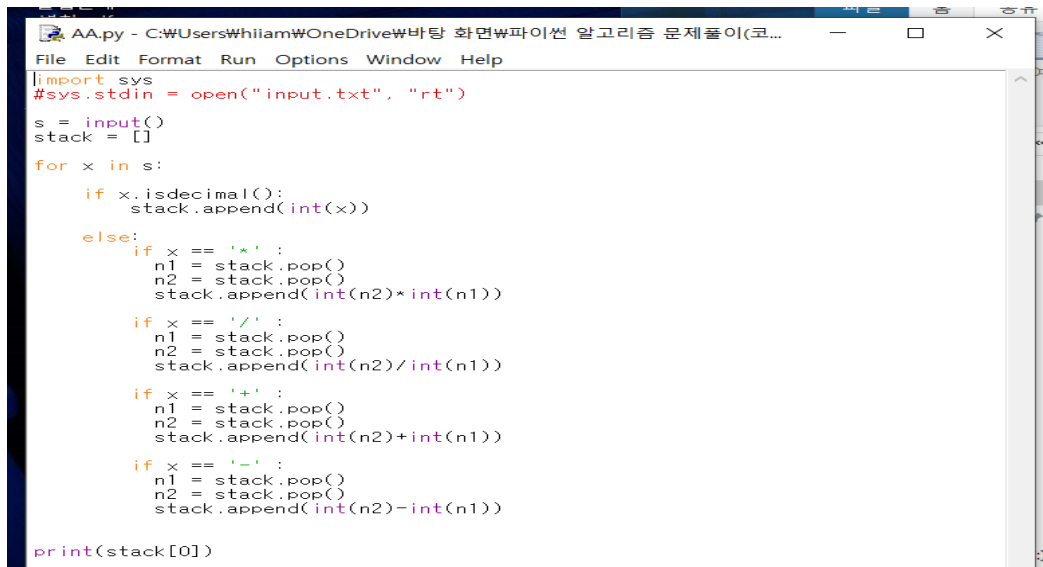
- 1) 숫자를 만나면 무조건 stack에 push
- 2) 그러다가 연산자를 만나면 pop하여 앞의 두 개를 연산 후 다시 push

\*isdecimal(), isalpha() 등은 해당 문자열이 10진수인지 알파벳인지 나타내 주는 것. 따라서 리스트에 넣으면 문자열로 넣어짐. 숫자로 넣고 싶으면 int형으로 바꿔서 넣어주어야 함

-선생님 풀이와 내 풀이



->연산 순서를 지켜야 해서 어떻게 넣어야 하나 생각했는데 pop을 두 번 하면 됨



- 1) 숫자가 나오면 stack에 int형으로 push시킴
- 2) 연산자가 나오면, 연산자 앞 두 개의 피연산자를 pop시켜서 두 번째로 나온 피연산자, 맨 처음 나온 피연산자 순으로 연산을 수행 후 다시 stack에 push시킴
- 3) 그 후 stack에 마지막 남은 숫자를 print



## 5. 공주구하기(큐)

정보 왕국의 이웃 나라 외동딸 공주가 숲속의 괴물에게 잡혀갔습니다. 정보 왕국에는 왕자가 N명이 있는데 서로 공주를 구하러 가겠다고 합니다. 정보왕국의 왕은 다음과 같은 방법으로 공주를 구하러 갈 왕자를 결정하기로 했습니다. 왕은 왕자들을 나이 순으로 1번부터 N번까지 차례로 번호를 매긴다. 그리고 1번 왕자부터 N번 왕자까지 순서대로 시계 방향으로 돌아가며 동그랗게 앉게 한다. 그리고 1번 왕자부터 시계방향으로 돌아가며 1부터 시작하여 번호를 외치게 한다. 한 왕자가 K(특정숫자)를 외치면 그 왕자는 공주를 구하러 가는데서 제외되고 원 밖으로 나오게 된다. 그리고 다음 왕자부터 다시 1부터 시작하여 번호를 외친다. 이렇게 해서 마지막까지 남은 왕자가 공주를 구하러 갈 수 있다.

예를 들어 총 8명의 왕자가 있고, 3을 외친 왕자가 제외된다고 하자. 처음에는 3번 왕자가 3을 외쳐 제외된다. 이어 6, 1, 5, 2, 8, 4번 왕자가 차례대로 제외되고 마지막까지 남게 된 7번 왕자에게 공주를 구하러갑니다. N과 K가 주어질 때 공주를 구하러 갈 왕자의 번호를 출력하는 프로그램을 작성하시오.

\*큐: First In First Out(시계 방향), 입력이 시작하는 곳과 출력이 시작하는 곳이 다르다.

-Python에서는 deque 자료구조(<-)를 사용함(입력: append, appendleft 출력: popleft)

1) deque에서 특정 숫자인 k의 앞번호(k-1)까지는 번호를 부르고 다시 queue로 입력:

Popleft -> append

2) 특정 숫자(k)를 만나면: popleft

3) 특정 숫자 이후부터 다시 1번이 되어 Popleft -> append

4) 마지막 1개가 남으면 popleft

```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코딩테스트 대비)\섹션 4(자료구...
File Edit Format Run Options Window Help
import sys
from collections import deque
#sys.stdin = open("input.txt", "rt")

n, k = map(int, input().split())
dq=list(range(1,n+1)) #아직 리스트

dq = deque(dq) #dq를 queue형으로 변환

while dq: #dq가 빈 queue가 될 때까지

    for _ in range(k-1): #아무 변수 없이 반복문 실행, K제외 후 1번부터 다시 시작
        cur = dq.popleft()
        dq.append(cur)

    dq.popleft() # K는 제외시킴

    if len(dq) == 1: #마지막 1명이 남았을 때 출력
        print(dq[0])
        dq.popleft()
```

-queue 자료형 써주려면 from collections import deque 꼭 써줘야 함

- 1) list(range(1,n+1)) == dq = [ x for x in range(1,n+1)]로 공주를 구하는 왕자들 리스트 생성
- 2) 왕자들 리스트를 queue형태로 생성
- 3) dq가 빈 queue가 될 때까지, k-1번 동안 아무 변수없이 반복문을 실행하면서 번호 부르고 뒤쪽으로 이동
- 4) 특정 숫자가 나오면(k-1번의 for문이 끝나면) queue에서 제외(popleft)시킴
- 5) 1명의 왕자가 남을때까지 계속 for문을 실행
- 6) 1명의 왕자가 남으면, 그 왕자를 출력하고, popleft시켜줘야 while문이 종료됨

## 6. 응급실(큐)

메디컬 병원 응급실에는 의사가 한 명밖에 없습니다. 응급실은 환자가 도착한 순서대로 진료를 합니다. 하지만 위험도가 높은 환자는 빨리 응급조치를 의사가 해야 합니다. 이런 문제를 보완하기 위해 응급실은 다음과 같은 방법으로 환자의 진료순서를 정합니다.

- 환자가 접수한 순서대로의 목록에서 제일 앞에 있는 환자목록을 꺼냅니다.
- 나머지 대기 목록에서 꺼낸 환자 보다 위험도가 높은 환자가 존재하면 대기목록 제일 뒤로 다시

넣습니다. 그렇지 않으면 진료를 받습니다.

현재 N명의 환자가 대기목록에 있습니다. N명의 대기목록 순서의 환자 위험도가 주어지면, 대기목록상의 M번째 환자는 몇 번째로 진료를 받는지 출력하는 프로그램을 작성하세요.

대기목록상의 M번째는 대기목록의 제일 처음 환자를 0번째로 간주하여 표현한 것입니다.

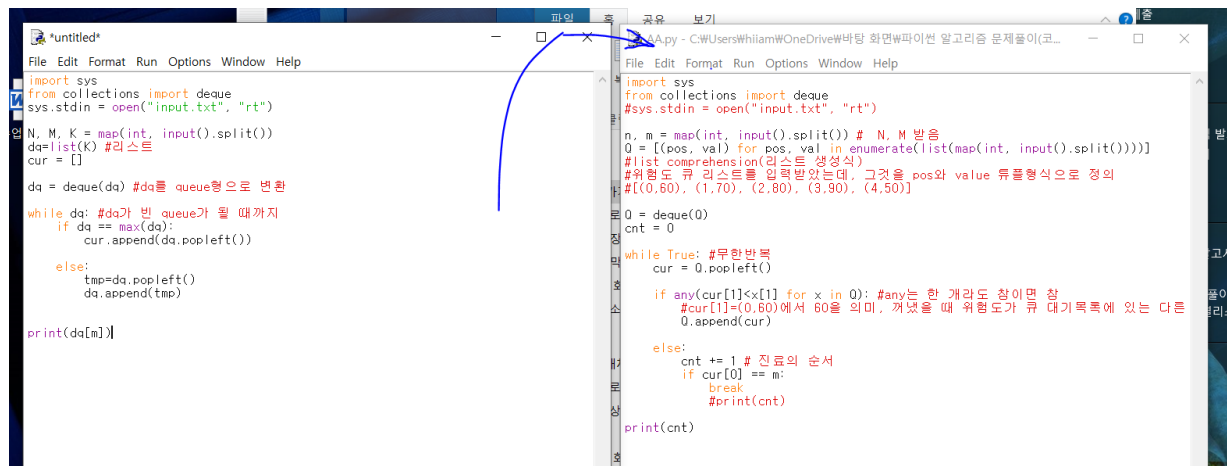
입력:

5 2

60 50 70 80 90

출력

3



1) index와 값이 두 개 다 필요할 때는 enumerate를 사용한다

2) 진료의 순서는 count를 따로 정의해서 사용

3) popleft는 반복되므로 한 번만 사용

4) any는 queue에서 해당 값을 제외한 다른 값들과 비교해줌

5) tuple이 x(0, 30)이면, x[0]=0, x[1]=30을 의미한다

6) input.split()를 list형으로 변환해줘야 list로 변환됨 안 그러면 그냥 값만 대입됨

따로 queue의 목록을 바꾸지 않아도 되므로 While True 사용

- 1) 대기목록 중 제일 앞의 환자를 popleft
- 2) 그 환자보다 위험도가 큰 다른 대기목록의 환자가 1명이라도 존재한다면 queue의 맨 뒤로 append
- 3) 그렇지 않다면, 진료의 순서에 1을 더함
- 4) 그 환자가 m번째라면 break후 while문을 빠져나오고 해당 진료 순서를 print

## 7. 교육과정 설계(큐)

현수는 1년 과정의 수업계획을 짜야 합니다. 수업중에는 필수과목이 있습니다. 이 필수과목은 반드시 이수해야 하며, 그 순서도 정해져 있습니다.

만약 총 과목이 A, B, C, D, E, F, G가 있고, 여기서 필수과목이 CBA로 주어진다면 필수과목은 C, B, A과목이며 이 순서대로 꼭 수업계획을 짜야 합니다. 현수가 C, B, D, A, G, E로 수업계획을 짜면 제대로 된 설계이지만 C, G, E, A, D, B 순서로 짰다면 잘 못 설계된 수업계획이 됩니다. 필수과목순서가 주어진다면 현수가 짠 N개의 수업설계가 잘된 것이면 "YES", 잘못된 것이면 "NO"를 출력하는 프로그램을 작성하세요.

### ■ 입력설명

첫 줄에 한 줄에 필수과목의 순서가 주어집니다. 모든 과목은 영문 대문자입니다.

두 번째 줄에  $N(1 \leq N \leq 10)$ 이 주어집니다.

세 번째 줄부터 현수가 짠 N개의 수업설계가 주어집니다.(수업설계의 길이는 15이하이다)

### ■ 출력설명

수업설계가 잘된 것이면 "YES", 잘못된 것이면 "NO"를 출력합니다.

### ■ 입력예제 1

CBA

3

CBDAGE

FGCDAB

CTSBDEA

## ▣ 출력예제 1

#1 YES

#2 NO

#3 YES

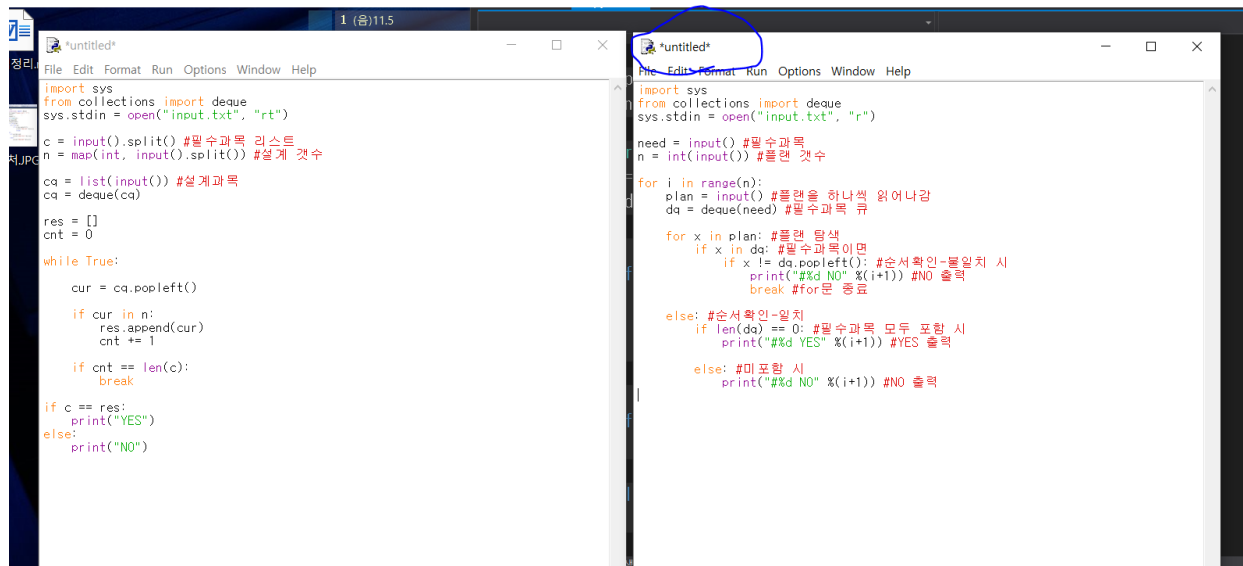
1) 필수과목을 deque로 만들

2) 설계한 과목을 입력받아 하나씩 탐색해 나간다.

-탐색 순서대로 해당 과목이 필수과목 큐에 있으면 필수과목 큐에서 과목을 꺼내고 일치/불일치 여부 판단한다

-탐색이 끝나면, 필수과목 큐가 empty queue가 되어야 함

3) 모든 탐색 후 불일치/일치 여부를 결정하는 것이 아니라 하나라도 순서가 맞지 않으면 NO인 것!!



```
import sys
from collections import deque
sys.stdin = open("input.txt", "rt")

c = input().split() #필수과목 리스트
n = map(int, input().split()) #설계 갯수

cq = list(input()) #설계과목
cq = deque(cq)

res = []
cnt = 0

while True:
    cur = cq.popleft()

    if cur in n:
        res.append(cur)
        cnt += 1

    if cnt == len(c):
        break

if c == res:
    print("YES")
else:
    print("NO")
```

```
import sys
from collections import deque
sys.stdin = open("input.txt", "r")

need = input() #필수과목
n = int(input()) #플랜 갯수

for i in range(n):
    plan = input()
    dq = deque(need)

    for x in plan:
        #플랜 탐색
        if x in dq:
            #필수과목이면
            if x != dq.popleft():
                #순서확인-불일치 시
                print("#%d NO" % (i+1)) #NO 출력
                break #for문 종료
        else:
            #순서확인-일치
            if len(dq) == 0:
                #필수과목 모두 포함 시
                print("#%d YES" % (i+1)) #YES 출력
            else:
                #미포함 시
                print("#%d NO" % (i+1)) #NO 출력
```

-왼쪽: 내풀이, 오른쪽: 선생님 풀이

```
import sys
from collections import deque
#sys.stdin = open("input.txt", "rt")
require = input()
n = int(input())
for x in range(1, n+1):
    plan = input()
    rq = deque(require) #이걸 미리 선언하면, empty queue가 계속 돌아감
                        #다른 이름으로 선언해주어야 함
    for y in plan:
        if y in rq:
            if y != rq.popleft():
                print("#%d NO" %x)
                break
        else:
            if len(rq) == 0:
                print("#%d YES" %x)
            else:
                print("#%d NO" %x)
```

```
import sys
from collections import deque
#sys.stdin = open("input.txt", "r")
need = input() #필수과목
n = int(input()) #총몇 개수
for i in range(n):
    plan = input() #플랜을 하나씩 읽어옴
    dq = deque(need) #필수과목 큐
    for x in plan: #플랜 탐색
        if x in dq: #필수과목이면
            if x != dq.popleft(): #순서 확인-불일치 시
                print("#%d NO" %(i+1)) #NO 출력
                break #for문 종료
        else: #순서확인-일치
            if len(dq) == 0: #필수과목 모두 포함 시
                print("#%d YES" %(i+1)) #YES 출력
            else: #미포함 시
                print("#%d NO" %(i+1)) #NO 출력
```

- 1) 순서가 필요한 것을 queue에 넣기
- 2) dq를 for문 안에 선언해주는 것은 dq에서 계속 pop을 하기 때문이다. 변수 밖에서 선언해주면, 한 번 들고 나서 계속 empty queue로 돌아서 YES가 나옴(dq라고 이름도 따로 선언해주어야 함)
- 3) python에는 if~else~문 뿐만 아니라 for~ else~ 문이 있다.

-for문이 정상적으로 돌면, else문이 실행안되지만, for문이 비정상 종료되면, else문이 동작하게 됨

- 1) 필수과목을 입력받아 need에 저장
- 2) 설계과목 개수를 입력받아 n에 저장
- 3) 설계과목 개수 입력받는 동안(n번 동안) 설계과목을 plan에 저장하고, 필수과목을 queue로 변환하여 dq에 저장
- 4) 설계과목에서 하나를 꺼냈는데 그것이 dq에 존재하면, popleft하여 순서를 확인하고, 순서가 일치하지 않으면(순서대로 꺼낸 것과 일치하지 않으면) NO 출력 후 **break**
- 5) 만일 순서가 다 맞으면 필수과목을 모두 다 포함했는지 확인하고 YES 출력, 또는 필수과목을 하나도 안 넣었거나 모두 포함하지 않았다면 NO 출력

## 8. 단어 찾기

현수는 영어로 시를 쓰는 것을 좋아합니다. 현수는 시를 쓰기 전에 시에 쓰일 단어를 미리 노트에 적어둡니다. 이번에는 N개의 단어를 노트에 적었는데 시에 쓰지 않은 단어가 하나 있다고 합니다. 여러

분이 찾아 주세요.

#### ■ 입력설명

첫 번째 줄에 자연수  $N(3 \leq N \leq 100)$ 이 주어진다. 두 번째 줄부터 노트에 미리 적어놓은  $N$ 개의 단어가 주어지고, 이어 바로 다음 줄부터 시에 쓰인  $N-1$ 개의 단어가 주어진다.

#### ■ 출력설명

첫 번째 줄에 시에 쓰지 않은 한 개의 단어를 출력한다.

#### ■ 입력예제 1

5

big

good

sky

blue

mouse

sky

good

mouse

big

#### ■ 출력예제 1

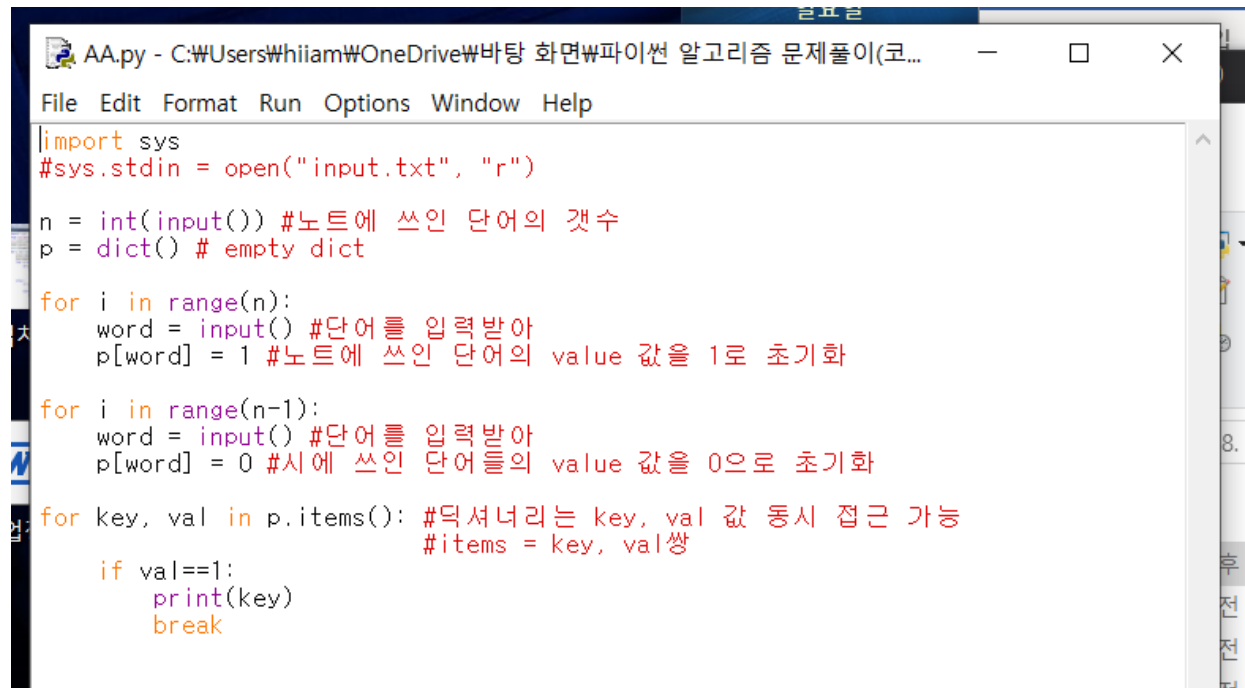
Blue

**\*딕셔너리로 hash를 구현한다:** list와 다르게 index(key값)로 정수 뿐만 아니라, 알파벳, 단어 등을 쓸 수 있다.

1) 입력받은 단어들을 key값으로, value 값을 1로 만들고

2) 그 후에 시에 쓰인 단어들의 value 값을 0으로 만든 후 유일하게 1인 value의 key 값을 출력함

-선생님풀이



```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코...
File Edit Format Run Options Window Help

import sys
#sys.stdin = open("input.txt", "r")

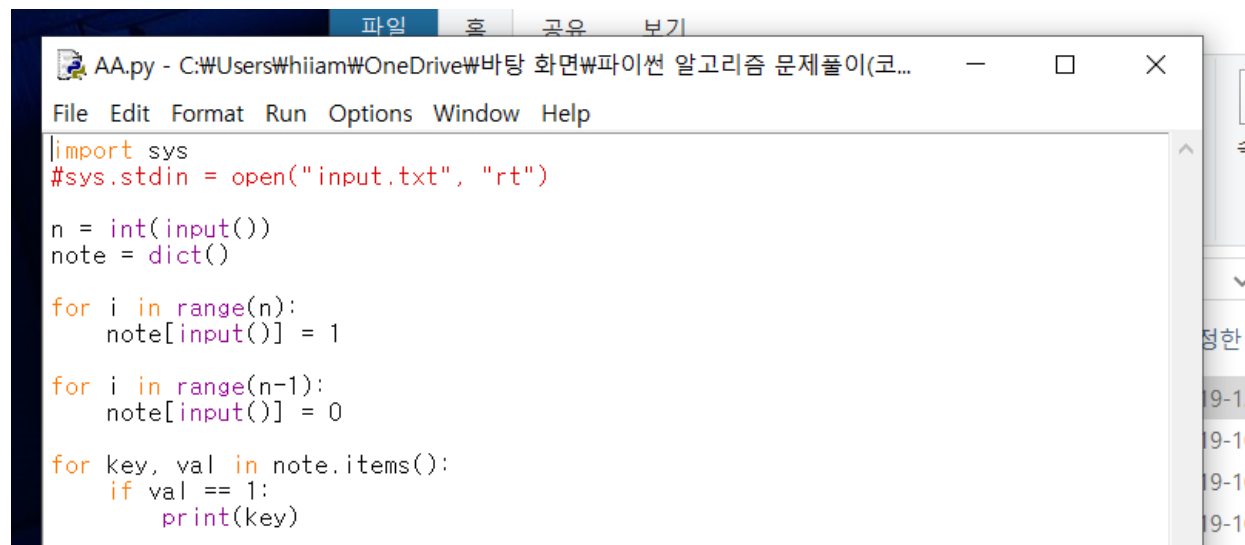
n = int(input()) #노트에 쓰인 단어의 갯수
p = dict() # empty dict

for i in range(n):
    word = input() #단어를 입력받아
    p[word] = 1 #노트에 쓰인 단어의 value 값을 1로 초기화

for i in range(n-1):
    word = input() #단어를 입력받아
    p[word] = 0 #시에 쓰인 단어들의 value 값을 0으로 초기화

for key, val in p.items(): #딕셔너리는 key, val 값 동시 접근 가능
    #items = key, val쌍
    if val==1:
        print(key)
        break
```

-내풀이



```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코...
File Edit Format Run Options Window Help

import sys
#sys.stdin = open("input.txt", "rt")

n = int(input())
note = dict()

for i in range(n):
    note[input()] = 1

for i in range(n-1):
    note[input()] = 0

for key, val in note.items():
    if val == 1:
        print(key)
```

### 9-1. Anagram(아나그램 : 구글 인터뷰 문제)(딕셔너리 해쉬)

Anagram이란 두 문자열이 알파벳의 나열 순서를 다르지만 그 구성이 일치하면 두 단어는 아나그램



이라고 합니다. 예를 들면 AbaAeCe 와 baeeACA 는 알파벳을 나열 순서는 다르지만 그 구성을 살펴 보면 A(2), a(1), b(1), C(1), e(2)로 알파벳과 그 개수가 모두 일치합니다. 즉 어느 한 단어를 재배열하면 상대편 단어가 될 수 있는 것을 아나그램이라 합니다. 길이가 같은 두 개의 단어가 주어지면 두 단어가 아나그램인지 판별하는 프로그램을 작성하세요. 아나그램 판별시 대소문자가 구분됩니다.

#### ■ 입력설명

첫 줄에 첫 번째 단어가 입력되고, 두 번째 줄에 두 번째 단어가 입력됩니다.

단어의 길이는 100을 넘지 않습니다.

#### ■ 출력설명

두 단어가 아나그램이면 "YES"를 출력하고, 아니면 "NO"를 출력합니다.

#### ■ 입력예제 1

AbaAeCe

baeeACA

#### ■ 출력예제 1

YES

1) 빈 딕셔너리의 값이나 키를 출력하거나 이용한다고 하면 error

2) dict의 멤버 함수로 dict.get함수를 사용

-dict.get(A, 0): 해당 dict에 A라는 key 값이 있으면 A의 value값을 return 하고, 없으면 0을 return함

3) 해당 문제는 이 dict.get을 사용해 counting 하는 함수

-내풀이

```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코...
File Edit Format Run Options Window Help

import sys
#sys.stdin = open("input.txt", "r")

an = list(input()) + list(input())
p = dict()

for i in an:
    p[i] = an.count(i)

for key, val in p.items():
    if (val % 2) != 0:
        print("NO")
        break
    else:
        print("YES")
        break
```

-선생님 풀이

```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코...
File Edit Format Run Options Window Help

import sys
#sys.stdin = open("input.txt", "r")

a1 = input()
a2 = input()

str1 = dict() #빈 dict1
str2 = dict() #빈 dict2

for x in a1:
    str1[x] = str1.get(x, 0) + 1 #빈 딕셔너리 사용 시 error 발생
for x in a2:
    str2[x] = str2.get(x, 0) + 1

if str1 == str2: # 두 딕셔너리 비교 (key, value 정확히 일치(대소문자)하면 true)
    print("YES")
else:
    print("NO")
```

=> 기업에서 dict로 구현하면 싫어할 수 있음 면접 볼 때(너무 간단)

=> 리스트로 직접 풀어보라고 지시할수도

## 9-2. Anagram(아나그램 : 구글 인터뷰 문제)(리스트 해쉬)

-내풀이

```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코...
File Edit Format Run Options Window Help
import sys
#sys.stdin = open("input.txt", "r")

an1 = list(input())
an2 = list(input())

for x in an1:
    if an1.count(x) != an2.count(x):
        print("NO")
        break
else:
    print("YES")
```

-선생님풀이 : ASCII number를 활용한 직접 hashing

```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코딩테스트 대비)\섹션 4(자료구조 활용)\W9-1. 아나그램(구글)\WAA.py (3.7.4)
File Edit Format Run Options Window Help
import sys
#sys.stdin = open("input.txt", "r")

an1 = list(input())
an2 = list(input())

#대문자 26+소문자 26 = 52개, 0으로 리스트 먼저 초기화
str1 = [0]*52 #an1 hashing
str2 = [0]*52 #an2 hashing

#an1의 알파벳들을 ASCII code로 변환 후 str1 인덱스에 해싱시킴
for x in an1:
    if x.isupper():
        #대문자 A=65(90)이므로, ord(x)-65값으로 indexing하고, 1을 더함(ex) A가 있다는 뜻
        #str1의 0~25(26개)까지를 indexing
        #ord(x) = x의 ASCII code 값 반환
        str1[ord(x)-65] += 1
    else:
        #소문자 a=97(122)이므로, ord(x)-71값으로 indexing하고, 1을 더함(ex) a가 있다는 뜻
        #str2의 26~52(26개)까지를 indexing
        #97-71 = 26
        str1[ord(x)-71] += 1

#an2의 알파벳들을 ASCII code로 변환 후 str2 인덱스에 해싱시킴
for x in an2:
    if x.isupper():
        #대문자 A=65이므로, ord(x)-65값으로 indexing하고, 1을 더함(ex) A가 있다는 뜻
        #ord(x) = x의 ASCII code 값 반환
        str2[ord(x)-65] += 1
    else:
        #소문자 a=97이므로, ord(x)-71값으로 indexing하고, 1을 더함(ex) a가 있다는 뜻
        str2[ord(x)-71] += 1

for i in range(52):
    if str1[i] != str2[i]:
        print("NO")
        break
else:
    print("YES")
```

1) 입력1을 an1, 입력2를 an2에 저장하고 이 입력값들을 ASCII CODE로 indexing하여 직접 hashing할 리스트 str1, str2를 생성해줌

2) str1~2는 대문자+소문자가 모두 52개이므로, 처음에 0으로 초기화하여 만들어줌

3) an1~2의 원소 x가 대문자(isupper)이면, ord(x)로 ASCII CODE로 변환한 다음 65를 빼주어 str1~2에 indexing 시키고, 해당 원소가 있다는 뜻으로 1을 더해줌

(여기서 65를 빼주는 이유는 A의 아스키 코드 값이 65이므로, str1[0]~str[25]=A~Z로 indexing되게 해주려고)

4) an1~2의 원소 x가 소문자(islower)라면, 역시 ord(x)로 ASCII CODE로 변환한 다음 71을 빼주어 str1~2에 indexing 시키고 해당 원소가 있다는 뜻으로 1을 더해준다.

(여기서 71인 이유는, 소문자 a의 아스키 코드 값은 97이고, 25까지는 indexing 되어 있기 때문에, 71을 빼주어야 str1[26]~str[52]=a~z로 indexing 되기 때문이다)

5) 그 후 hashing list 길이만큼 다시 for문을 돌면서 str1과 str2의 알파벳 개수가 일치하지 않으면 NO 반환 후 break해주고 그게 아니라면 YES 반환

(여기서 NO 이후 break 하는 이유는 YES 먼저 하고 break하면 그 다음에 NO가 있는데 무시됨)

## 10. 최소힙(힙)

최소힙은 완전이진트리로 구현된 자료구조입니다. 그 구성은 부모 노드값이 왼쪽자식과 오른쪽 자식 노드의 값보다 작게 트리를 구성하는 것입니다. 그렇게 하면 트리의 루트(root)노드는 입력된 값들 중 가장 작은 값이 저장되어 있습니다. 예를 들어 5 3 2 1 4 6 7순으로 입력되면 최소힙 트리는 아래와 같이 구성됩니다.

1

2 3

5 4 6 7

최소힙 자료를 이용하여 다음과 같은 연산을 하는 프로그램을 작성하세요.

1) 자연수가 입력되면 최소힙에 입력한다.

2) 숫자 0 이 입력되면 최소힙에서 최솟값을 꺼내어 출력한다.

(출력할 자료가 없으면 -1를 출력한다.)

3) -1이 입력되면 프로그램 종료한다.

#### ■ 입력설명

첫 번째 줄부터 숫자가 입력된다. 입력되는 숫자는 100,000개 이하이며 각 숫자의 크기는 정수형 범위에 있다.

#### ■ 출력설명

2) 연산을 한 결과를 보여준다.

#### ■ 입력예제 1

5

3

6

0

5

0

2

4

0

-1

#### ■ 출력예제 1

3

5

2

\*힙(Heap): 이진 트리를 이용한 자료구조

1(1: 부모)

2 3 (2: 왼쪽 자식, 3: 오른쪽 자식) => (트리의 기본 구성 단위: 1, 2, 3)

5 4 6 7(왼쪽 서브트리: 2,5,4, 오른쪽 서브트리: 2,6,7)

-0레벨: 1, 1레벨: 2,3, 2레벨: 5,4,6,7

-같은 레벨에서는 왼쪽부터 입력시킨다.

<면접에서 물어볼 수 있음>

\*최소힙: 부모 노드의 값이 자식의 노드 값보다 무조건 작게 구성된 완전이진트리(root노드가 가장 작은 값의 노드)

1) 처음에 값을 부모노드에 입력시킨다

2) 그 다음 레벨에 입력시킨 값이 부모노드와 비교하여 부모노드보다 작으면 부모노드와 해당노드를 swapping시킴(uphip)

3) 그 다음 레벨에 입력시켰는데 그 위 레벨의 부모노드 값과 비교하여 부모노드보다 작으면 부모노드값과 해당노드 값을 swapping시킴(uphip)

4) 그렇게 만들어진 최소힙에서 하나를 pop하면, 가장 작은 값의 root 노드가 pop됨

5) 그 후 **마지막 레벨의 오른쪽 노드값을 root 노드로 올리고**, downhip을 통해 최소힙을 다시 만들어 주는 과정을 거침

->자식노드 값이 부모노드 값보다 작으면 swaping시키는 것(부모노드가 내려가는 것)

\*파이썬에서는 heapq를 import시켜서 사용

```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코...
File Edit Format Run Options Window Help

import sys
import heapq as hq
#sys.stdin = open("input.txt", "rt")

#파이썬에서 heapq를 이용하려면, 리스트 필요
#해당 리스트에서 heappush, heappop으로 heap구현

heap = []

while True:
    n = int(input())
    if n == -1:
        break
    if n == 0: #0이면 root node값을 pop
        if len(heap) == 0: #힙이 비어있을 때
            print(-1)
        else:
            print(hq.heappop(heap))
            #heap리스트에서 자료를 하나 pop해 주는데,
            #heap구조에서는 root node값이 pop됨
    else:
        hq.heappush(heap, n)
```

1) 파이썬에서 heapq로 heap구현하는데, 리스트를 만들어서 거기에 자료를 넣은 다음에 그 리스트를 heapq에 push하고 pop하는 방식으로 사용

2) 입력 값에서 -1을 만나면 종료

3) 0을 만나면 root node를 pop시킴

-힙이 비어있으면 -1을 출력하고, 그게 아니라면 pop

4) 일반 자연수가 들어가면 리스트에 값을 넣어서 힙에 push시킴

=>면접에서 힙에 pop, push시키는 과정을 물어볼 수 있음

## 11. 최대힙(힙)

최대힙은 완전이진트리로 구현된 자료구조입니다. 그 구성은 부모 노드값이 왼쪽자식과 오른쪽 자식 노드의 값보다 크게 트리를 구성하는 것입니다. 그렇게 하면 트리의 루트(root)노드는 입력된 값들 중

가장 큰 값이 저장되어 있습니다. 예를 들어 5 3 2 1 4 6 7순으로 입력되면 최대힙 트리는 아래와 같이 구성됩니다.

7

4 6

1 3 2 5

최대힙 자료를 이용하여 다음과 같은 연산을 하는 프로그램을 작성하세요.

- 1) 자연수가 입력되면 최대힙에 입력한다.
- 2) 숫자 0 이 입력되면 최대힙에서 최댓값을 꺼내어 출력한다.  
(출력할 자료가 없으면 -1를 출력한다.)
- 3) -1이 입력되면 프로그램 종료한다.

#### ■ 입력설명

첫 번째 줄부터 숫자가 입력된다. 입력되는 숫자는 100,000개 이하이며 각 숫자의 크기는 정수형 범위에 있다.

#### ■ 출력설명

- 2) 연산을 한 결과를 보여준다.

#### ■ 입력예제 1

5

3

6

0

5



0

2

4

0

-1

#### ▣ 출력예제 1

6

5

5

\*최대힙: 부모 노드의 값이 두 자식의 노드 값보다 무조건 크데 구성된 완전이진트리(root노드가 가장 큰 최대 값의 노드)

1) 처음에 값을 부모노드에 입력시킨다

2) 그 다음 레벨에 입력시킨 값이 부모노드와 비교하여 부모노드보다 크면 부모노드와 해당노드를 swapping시킴(uphip)

3) 그 다음 레벨에 입력시켰는데 그 위 레벨의 부모노드 값과 비교하여 부모노드보다 크면 부모노드 값과 해당노드 값을 swapping시킴(uphip)

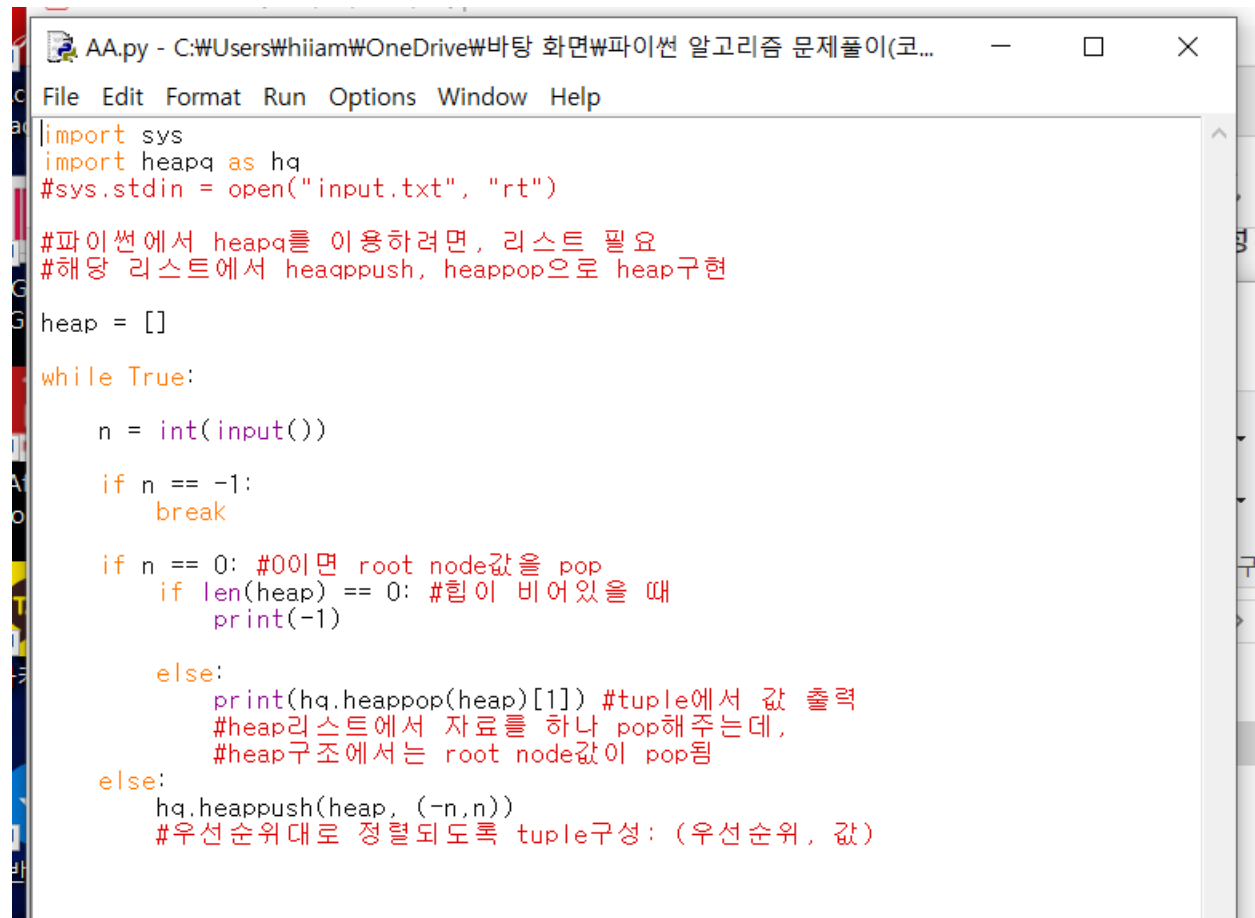
4) 그렇게 만들어진 최대힙에서 하나를 pop하면, 가장 큰 값의 root 노드가 pop됨

5) 그 후 **마지막 레벨의 오른쪽 노드값을 root 노드로 올리고**, downhip을 통해 최대힙을 다시 만들어 주는 과정을 거침

->자식노드 값이 부모노드 값보다 크면 swaping시키는 것(부모노드가 내려가는 것)

\* heapq module은 기본적으로 최소힙으로 작동한다. 여기서 최대힙의 효과를 내려면 원소의 부호를 바꾸어서 넣는 방법이 있다. 그 후, 최대힙의 root 노드를 pop하려면 부호를 바꿔서 다시 출력하면 된다.

-내풀이



```
AA.py - C:\Users\Whilam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코...
File Edit Format Run Options Window Help
import sys
import heapq as hq
#sys.stdin = open("input.txt", "rt")

#파이썬에서 heapq를 이용하려면, 리스트 필요
#해당 리스트에서 heappush, heappop으로 heap구현

heap = []

while True:
    n = int(input())
    if n == -1:
        break
    if n == 0: #0이면 root node값을 pop
        if len(heap) == 0: #힘이 비어있을 때
            print(-1)
        else:
            print(hq.heappop(heap)[1]) #tuple에서 값 출력
            #heap리스트에서 자료를 하나 pop해 주는데,
            #heap구조에서는 root node값이 pop됨
    else:
        hq.heappush(heap, (-n,n))
        #우선순위대로 정렬되도록 tuple구성: (우선순위, 값)
```

-선생님 풀이

```
File Edit Format Run Options Window Help
import sys
import heapq as hq
sys.stdin = open("input.txt", "rt")

#파이썬에서 heapq를 이용하려면, 리스트 필요
#해당 리스트에서 heappush, heappop으로 heap구현

heap = []

while True:
    n = int(input())

    if n == -1:
        break

    if n == 0: #0이면 root node값을 pop
        if len(heap) == 0: #힙이 비어있을 때
            print(-1)
        else:
            print(-hq.heappop(heap)) #다시 -붙여서 최대값 출력
            #heap리스트에서 자료를 하나 pop해 주는데,
            #heap구조에서는 root node값이 pop됨
    else:
        hq.heappush(heap, -n)
        # -붙여서 최대힙 만들고
```