코멘토 직무부트캠프 웹 개발 1주차 과제

멘티: 유혜림

(Git: https://github.com/HRim-dev/python_Web.git)

<Python 기본문법 익히기-프로그래머스 Level1(Python)문제 풀이>

문제1. 두 개 뽑아서 더하기

정수 배열 numbers가 주어집니다. numbers에서 서로 다른 인덱스에 있는 두 개의 수를 뽑아 더해서 만들 수 있는 모든 수를 배열에 오름차순으로 담아 return 하도록 solution 함수를 완성해주세요.

제한사항

- numbers의 길이는 2 이상 100 이하입니다.
  - numbers의 모든 수는 0 이상 100 이하입니다.

입출력 예

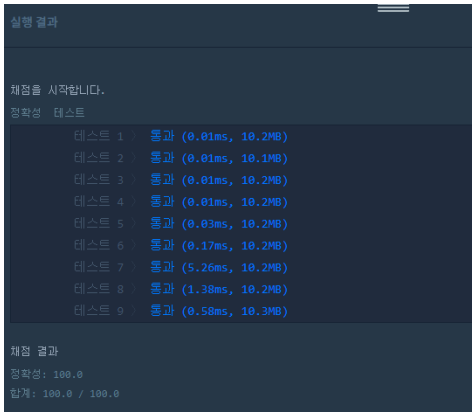
numbers	result
[2,1,3,4,1]	[2,3,4,5,6,7]
[5,0,2,7]	[2,5,7,9,12]

-나의 풀이

solution.py

```python
def solution(numbers):
    answer = []       #answer:결과값을 넣는 배열

    #numbers 배열내에 있는 서로 다른 인덱스에 있는 두개의 수를 뽑아서 만들수 있는 모든수를 구하기 위해
    #이중 반복문 사용
    for i in range(0,len(numbers)-1,1):         #i=0,1,2,3...(numbers배열 크기-1)
        for j in range(i+1,len(numbers),1):     #j=1,2,3,4...(numbers배열 크기)
            result=numbers[i]+numbers[j]        #result=두개의 수를 뽑아서 구한 합의 결과

            #answer에 result값이 중복되어있나 확인
            if result not in answer:        #answer배열에 result값이 들어있지 않으면
                answer.append(result)       #result를 answer배열에 추가

    #결과배열을 오름차순 정렬
    return sortec(answer)
```

-실행 결과



실행 결과

채점을 시작합니다.

정확성 테스트

테스트 1 통과 (0.01ms, 10.2MB)
테스트 2 통과 (0.01ms, 10.1MB)
테스트 3 통과 (0.01ms, 10.2MB)
테스트 4 통과 (0.01ms, 10.2MB)
테스트 5 통과 (0.03ms, 10.2MB)
테스트 6 통과 (0.17ms, 10.2MB)
테스트 7 통과 (5.26ms, 10.2MB)
테스트 8 통과 (1.38ms, 10.2MB)
테스트 9 통과 (0.58ms, 10.3MB)

채점 결과

정확성: 100.0

합계: 100.0 / 100.0

•보완하면 좋을 점(다른 사람의 풀이): Set(집합)을 사용하여 저장하면 중복저장이 되지 않는다.

(반복문을 사용해 중복 검사하지 않아도 됨. But, 순서가 없어서 정렬은 해야 함)

참조: https://velog.io/@daybreak/Python%EC%97%90%EC%84%9C-%EB%A6%AC%EC%8A%A4%ED%8A%B8-%EC%A4%91%EB%B3%B5%EC%A0%9C%EA%B1%B0

문제2. 완주하지 못한 선수

완주하지 못한 선수

마라톤에 참여한 선수들의 이름이 담긴 배열 participant와 완주한 선수들의 이름이 담긴 배열 completion이 주어질 때, 완주하지 못한 선수의 이름을 return 하도록 solution 함수를 작성해주세요.

제한사항

- 마라톤 경기에 참여한 선수의 수는 1명 이상 100,000명 이하입니다.
- completion의 길이는 participant의 길이보다 1 작습니다.
- 참가자의 이름은 1개 이상 20개 이하의 알파벳 소문자로 이루어져 있습니다.
- 참가자 중에는 동명이인이 있을 수 있습니다.

입출력 예

participant	completion	return
["leo", "kiki", "eden"]	["eden", "kiki"]	"leo"
["marina", "josipa", "nikola", "vinko", "filipa"]	["josipa", "filipa", "marina", "nikola"]	"vinko"
["mislav", "stanko", "mislav", "ana"]	["stanko", "ana", "mislav"]	"mislav"

- 나의 풀이

solution.py

```python
def solution(participant, completion):
    answer = ''  #answer:결과값
    flag=True    #flag: 참가자 명단(participant)과 완주한 선수 명단(completion)의 일치여부를 나타내주는 변수

    #참가자 명단 이름의 순서와 완주자 명단 이름의 순서를 일치시키기 위해 오름차순으로 정렬
    participant=sorted(participant)
    completion=sorted(completion)

    for c in range(len(completion)):
        if participant[c]!=completion[c]:   #참가자 명단 인덱스와 참여자 명단의 인덱스 값이 다르다면
            answer=participant[c]           #일치하지 않는 사람은 완주하지 못한 사람
            flag=False                      #완주못한 사람 존재
            break

    if flag:                                #완주자 명단을 모두 확인했을 때 참여자 명단과 같다면
        answer=participant[len(participant)-1]  #참여자 명단 마지막에 있는 사람이 완주하지 못한 사람

    return answer
```

- 실행 결과

실행 결과

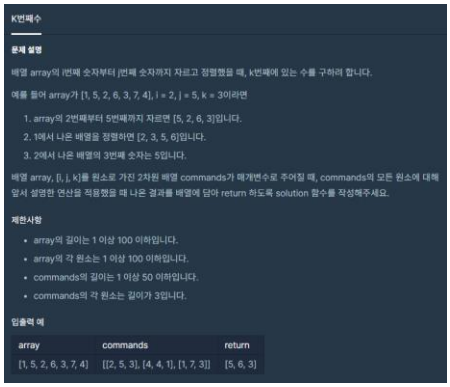
```
채점을 시작합니다.
정확성  테스트
        테스트 1 〉  통과 (0.01ms, 10.2MB)
        테스트 2 〉  통과 (0.01ms, 10.2MB)
        테스트 3 〉  통과 (0.29ms, 10.2MB)
        테스트 4 〉  통과 (0.68ms, 10.4MB)
        테스트 5 〉  통과 (0.59ms, 10.4MB)
효율성  테스트
        테스트 1 〉  통과 (35.66ms, 18MB)
        테스트 2 〉  통과 (67.49ms, 22.3MB)
        테스트 3 〉  통과 (83.37ms, 24.8MB)
        테스트 4 〉  통과 (96.95ms, 26.4MB)
        테스트 5 〉  통과 (94.05ms, 26.3MB)

채점 결과
정확성: 50.0
효율성: 50.0
합계: 100.0 / 100.0
```

- 보완하면 좋을 점(다른 사람의 풀이)

1) 해시를 이용하는 문제인데 해시를 사용하지 않았다.
해시와 딕셔너리에 대해 더 공부해야겠다.

2) collections.Counter 를 이용해 더 간단히 작성이 가능했다.
컬렉션에 대해서도 공부해야겠다.

문제3. K번째 수



K번째수

문제 설명

배열 array의 i번째 숫자부터 j번째 숫자까지 자르고 정렬했을 때, k번째에 있는 수를 구하려 합니다.

예를 들어 array가 [1, 5, 2, 6, 3, 7, 4], i = 2, j = 5, k = 3이라면

1. array의 2번째부터 5번째까지 자르면 [5, 2, 6, 3]입니다.
2. 1에서 나온 배열을 정렬하면 [2, 3, 5, 6]입니다.
3. 2에서 나온 배열의 3번째 숫자는 5입니다.

배열 array, [i, j, k]를 원소로 가진 2차원 배열 commands가 매개변수로 주어질 때, commands의 모든 원소에 대해 앞서 설명한 연산을 적용했을 때 나온 결과를 배열에 담아 return 하도록 solution 함수를 작성해주세요.

제한사항

- array의 길이는 1 이상 100 이하입니다.
- array의 각 원소는 1 이상 100 이하입니다.
- commands의 길이는 1 이상 50 이하입니다.
- commands의 각 원소는 길이가 3입니다.

입출력 예

array	commands	return
[1, 5, 2, 6, 3, 7, 4]	[[2, 5, 3], [4, 4, 1], [1, 7, 3]]	[5, 6, 3]

- 나의 풀이

solution.py

```python
def solution(array, commands):
    answer = []     #answer: 결과 값을 담은 배열

    #commands 배열의 한 행씩 접근
    for row in range(len(commands)):
        temp = []               #temp: 잘라진 원소들을 담는 배열
        i=commands[row][0]-1    #i=array배열에 자를 시작위치(인덱스는 0부터 시작하므로 -1해줌)
        j=commands[row][1]      #j: arrya배열에 자를 마지막 위치
        k=commands[row][2]-1    #k: 자른 배열(temp) 중 결과배열에 넣을 원소의 위치(인덱스는 0부터 시작하므로 -1해줌)

        for index in range(i,j):        #i에서 j까지 잘라서
            temp.append(array[index])   #temp배열에 넣어줌

        temp=sorted(temp)               #temp 배열(자른 배열) 오름차순 정렬

        answer.append(temp[k])          #temp배열(자른 배열)에 k번째 수 answer(결과 배열)에 넣어 줌

    return answer
```

- 실행 결과

실행 결과

채점을 시작합니다.

정확성 테스트

테스트 1 〉 통과 (0.01ms, 10.1MB)
테스트 2 〉 통과 (0.01ms, 10.2MB)
테스트 3 〉 통과 (0.01ms, 10.2MB)
테스트 4 〉 통과 (0.01ms, 10.2MB)
테스트 5 〉 통과 (0.01ms, 10.1MB)
테스트 6 〉 통과 (0.01ms, 10.3MB)
테스트 7 〉 통과 (0.01ms, 10.2MB)

채점 결과

정확성: 100.0

합계: 100.0 / 100.0

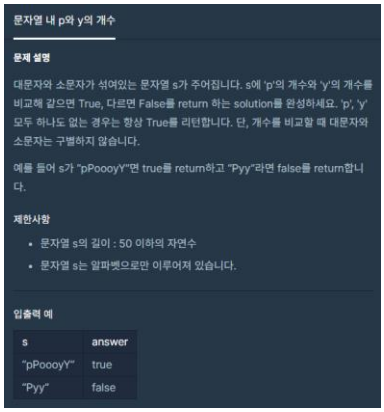
- 보완하면 좋을 점(다른 사람의 풀이)

1. 람다식으로도 표현이 가능하다는 것을 알았다.
람다식에 대해 이해하고 익힐 수 있도록 공부해야겠다.

2. 각 행마다 i,j,k 값을 한번에 입력하는 방법도 있었다.

```python
def solution(array, commands):
    answer = []
    for command in commands:
        i,j,k = command
        answer.append(list(sorted(array[i-1:j]))[k-1])
    return answer
```

문제4. 문자열 내 p와 y의 개수



문자열 내 p와 y의 개수

문제 설명

대문자와 소문자가 섞여있는 문자열 s가 주어집니다. s에 'p'의 개수와 'y'의 개수를 비교해 같으면 True, 다르면 False를 return 하는 solution를 완성하세요. 'p', 'y' 모두 하나도 없는 경우는 항상 True를 리턴합니다. 단, 개수를 비교할 때 대문자와 소문자는 구별하지 않습니다.

예를 들어 s가 "pPoooyY"면 true를 return하고 "Pyy"라면 false를 return합니다.

제한사항

- 문자열 s의 길이 : 50 이하의 자연수
- 문자열 s는 알파벳으로만 이루어져 있습니다.

입출력 예

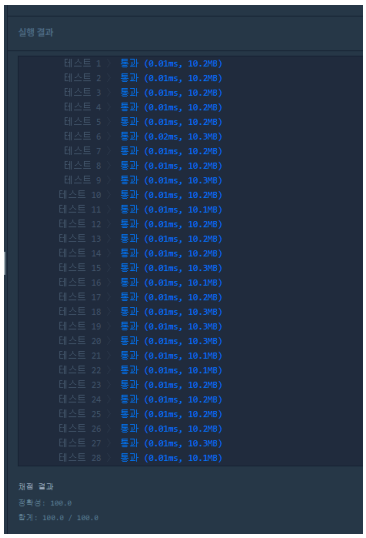
s	answer
"pPoooyY"	true
"Pyy"	false

- 나의 풀이

solution.py

```
def solution(s):
    answer = True               #answer: 결과값

    s=s.lower()                 #개수를 비교할때 대소문자 구별하지 않으므로 모든 문자를 소문자로 통일
    sList=list(s)               #문자열을 리스트(배열)로 변환(각 문자를 sList 배열원소로 저장)

    pCount=0                    #pCount: 문자 p의 개수
    yCount=0                    #yCount: 문자 y의 개수

    for i in range(len(sList)): #sList의 크기 만큼 문자 비교
        if sList[i]=='p':           #sList의 원소가 'p'이면
            pCount+=1                   #pCount 1 증가
        elif sList[i]=='y':         #sList의 원소가 'y'이면
            yCount+=1                   #yCount 1 증가

    if pCount==yCount:          #p의 개수와 y개수가 같으면
        answer=True                 #결과=True(참)
    else:                       #p의 개수와 y개수가 같지 않으면
        answer=False                #결과=False(거짓)

    return answer
```

- 실행 결과



실행 결과

테스트 1 〉 통과 (0.01ms, 10.2MB)
테스트 2 〉 통과 (0.01ms, 10.2MB)
테스트 3 〉 통과 (0.01ms, 10.2MB)
테스트 4 〉 통과 (0.01ms, 10.2MB)
테스트 5 〉 통과 (0.01ms, 10.2MB)
테스트 6 〉 통과 (0.02ms, 10.3MB)
테스트 7 〉 통과 (0.01ms, 10.2MB)
테스트 8 〉 통과 (0.01ms, 10.2MB)
테스트 9 〉 통과 (0.01ms, 10.3MB)
테스트 10 〉 통과 (0.01ms, 10.2MB)
테스트 11 〉 통과 (0.01ms, 10.1MB)
테스트 12 〉 통과 (0.01ms, 10.2MB)
테스트 13 〉 통과 (0.01ms, 10.2MB)
테스트 14 〉 통과 (0.01ms, 10.2MB)
테스트 15 〉 통과 (0.01ms, 10.3MB)
테스트 16 〉 통과 (0.01ms, 10.1MB)
테스트 17 〉 통과 (0.01ms, 10.2MB)
테스트 18 〉 통과 (0.01ms, 10.3MB)
테스트 19 〉 통과 (0.01ms, 10.3MB)
테스트 20 〉 통과 (0.01ms, 10.3MB)
테스트 21 〉 통과 (0.01ms, 10.1MB)
테스트 22 〉 통과 (0.01ms, 10.1MB)
테스트 23 〉 통과 (0.01ms, 10.2MB)
테스트 24 〉 통과 (0.01ms, 10.2MB)
테스트 25 〉 통과 (0.01ms, 10.2MB)
테스트 26 〉 통과 (0.01ms, 10.2MB)
테스트 27 〉 통과 (0.01ms, 10.3MB)
테스트 28 〉 통과 (0.01ms, 10.1MB)

채점 결과

정확성: 100.0

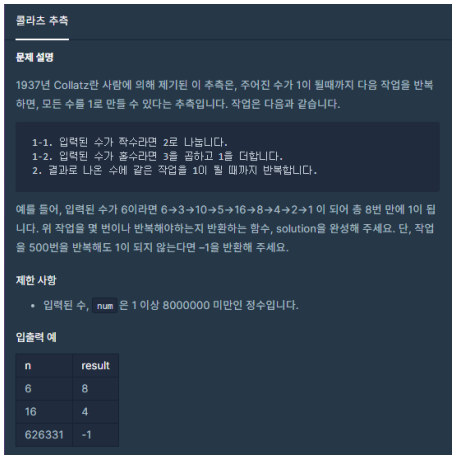
합계: 100.0 / 100.0

- 보완하면 좋을 점(다른 사람 풀이)

  list 자료형의 count()라는 메소드가 있다는 것을 알게 되었다.
  Count() 함수를 사용하면 더 간단히 작성이 가능하다.

```
def numPY(s):
    # 함수를 완성하세요
    return s.lower().count('p') == s.lower().count('y')
```

문제5. 콜라츠 추측



콜라츠 추측

문제 설명

1937년 Collatz란 사람에 의해 제기된 이 추측은, 주어진 수가 1이 될때까지 다음 작업을 반복하면, 모든 수를 1로 만들 수 있다는 추측입니다. 작업은 다음과 같습니다.

1-1. 입력된 수가 짝수라면 2로 나눕니다.
1-2. 입력된 수가 홀수라면 3을 곱하고 1을 더합니다.
2. 결과로 나온 수에 같은 작업을 1이 될 때까지 반복합니다.

예를 들어, 입력된 수가 6이라면 6→3→10→5→16→8→4→2→1 이 되어 총 8번 만에 1이 됩니다. 위 작업을 몇 번이나 반복해야하는지 반환하는 함수, solution을 완성해 주세요. 단, 작업을 500번을 반복해도 1이 되지 않는다면 –1을 반환해 주세요.

제한 사항

- 입력된 수, num 은 1 이상 8000000 미만인 정수입니다.

입출력 예

n	result
6	8
16	4
626331	-1

- 나의 풀이

solution.py

```python
def solution(num):
    answer = 0              #answer: 결과값(연산과정 횟수)

    while num>1:            #num이 1보다 클때까지 반복
        if answer==500:     #연산 횟수가 500회면
            return -1           #-1 반환
        elif num%2==0:      #num이 짝수이면
            num=num/2           #num을 2로 나눔
            answer+=1           #연산횟수 증가
        elif num%2==1:      #num이 홀수이면
            num=num*3+1         #num에 3을 곱하고 1을 더함
            answer+=1           #연산 횟수 증가

    return answer
```

- 실행 결과

실행 결과

테스트 1
입력값 : 6
기댓값 : 8
실행 결과 : 테스트를 통과하였습니다.

테스트 2
입력값 : 16
기댓값 : 4
실행 결과 : 테스트를 통과하였습니다.

테스트 3
입력값 : 626331
기댓값 : -1
실행 결과 : 테스트를 통과하였습니다.

테스트 결과 (~˘▾˘)~
3개 중 3개 성공

실행 테스트 케이스를 통과했다는 의미로, 작성한 코드가 문제의 정답은 아닐 수 있습니다.
(실행 테스트 케이스는 [테스트 케이스 추가하기] 버튼을 통해 확인하실 수 있습니다.)

문제6. 수박수박수박수박수?

수박수박수박수박수박수?

문제 설명

길이가 n이고, "수박수박수박수...."와 같은 패턴을 유지하는 문자열을 리턴하는 함수, solution을 완성하세요. 예를들어 n이 4이면 "수박수박"을 리턴하고 3이라면 "수박수"를 리턴하면 됩니다.

제한 조건

- n은 길이 10,000이하인 자연수입니다.

입출력 예

n	return
3	"수박수"
4	"수박수박"

- 나의 풀이

solution.py

```python
def solution(n):
    answer = ''             #answer: 결과값 저장

    for i in range(n):      #n만큼 패턴반복
        if i%2==0:              #i값(자릿수)이 짝수이면(문자열의 첫번째 숫자의 위치는 0부터 시작됨)
            answer+='수'            #answer(결과)에 '수'를 붙여준다.
        elif i%2==1:            #i값(자릿수)이 홀수이면
            answer+='박'            #answer(결과)에 '박'을 붙여준다.

    return answer
```

- 실행 결과

문제7. 소수 찾기

소수 찾기

문제 설명

1부터 입력받은 숫자 n 사이에 있는 소수의 개수를 반환하는 함수, solution을 만들어 보세요.

소수는 1과 자기 자신으로만 나누어지는 수를 의미합니다.
(1은 소수가 아닙니다.)

제한 조건

- n은 2이상 1000000이하의 자연수입니다.

입출력 예

n	result
10	4
5	3

- 나의 풀이

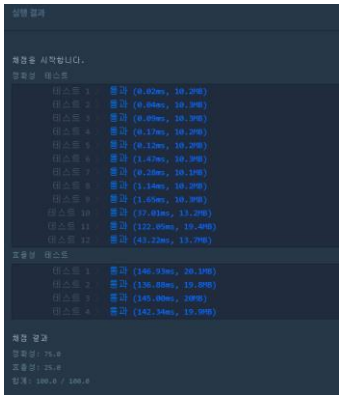
solution.py

```python
def solution(n):
    answer = 0              #answer: 결과값 저장 변수
                            #에라토스테네스의 체 초기화:모두 True로 초기화
    isPrime=[True]*(n+1)    #isPrime: 소수인지 아닌지 여부를 저장하는 배열
                            #(true:소수/false: 소수 아님)

    m=int(n**0.5)       #m: 반복 범위(n의 최대 약수가 sqrt(n)이하이므로 i=sqrt(n)까지 검사)

    for i in range(2, m+1):
        if isPrime[i]==True:                #i가 소수인 경우
            for j in range(i*i,n+1,i):      #i이후 i의 배수들을
                isPrime[j]=False            #False(소수아님)으로 바꿈

    answer=len([i for i in range(2,n+1) if isPrime[i]==True])
    #answer(결과)=소수 목록 산출 후 배열 크기

    return answer
```

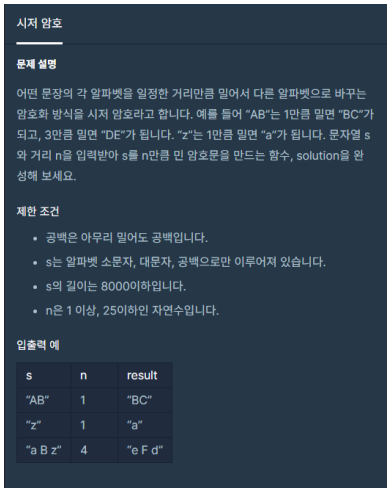
※에라토스테네스의 체에 대한 설명 참고:

(https://ko.wikipedia.org/wiki/%EC%97%90%EB%9D%BC%ED%86%A0%EC%8A%A4%ED%85%8C%EB%84%A4%EC%8A%A4%EC%9D%98_%EC%B2%B4)

- 실행 결과



문제8. 시저 암호



시저 암호

문제 설명

어떤 문장의 각 알파벳을 일정한 거리만큼 밀어서 다른 알파벳으로 바꾸는 암호화 방식을 시저 암호라고 합니다. 예를 들어 "AB"는 1만큼 밀면 "BC"가 되고, 3만큼 밀면 "DE"가 됩니다. "z"는 1만큼 밀면 "a"가 됩니다. 문자열 s와 거리 n을 입력받아 s를 n만큼 민 암호문을 만드는 함수, solution을 완성해 보세요.

제한 조건

- 공백은 아무리 밀어도 공백입니다.
- s는 알파벳 소문자, 대문자, 공백으로만 이루어져 있습니다.
- s의 길이는 8000이하입니다.
- n은 1 이상, 25이하인 자연수입니다.

입출력 예

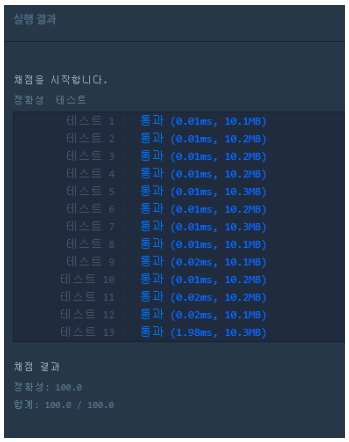
s	n	result
"AB"	1	"BC"
"z"	1	"a"
"a B z"	4	"e F d"

- 나의 풀이

solution.py

```python
def solution(s, n):
    answer = ''                         #answer: 결과값 저장되어있는 변수

    low = "abcdefghijklmnopqrstuvwxyz"  #low: 소문자 전체가 저장되었는 문자열
    up= low.upper()                     #up: 대문자 전체가 저장되어있는 문자열

    for char in s:                      #문자열 전체 검사
        if char in low:                     #만약 문자가 소문자라면
            index=low.find(char)+n              #전체 소문자 문자열을 찾고 index에 거기에 n만큼 더함
            answer+=low[index%26]               #26으로 나눈 나머지를 활용=>25를 초과하는 경우에도 활용가능
        elif char in up:                    #만약 문자가 대문자라면
            index=up.find(char)+n               #전체 소문자 문자열을 찾고 index에 거기에 n만큼 더함
            answer+=up[index%26]                #26으로 나눈 나머지를 활용=>25를 초과하는 경우에도 활용가

        else:                               #만약 문자가 공백일 경우
            answer+=" "                     #answer(결과값)에 공백 추가

    return answer
```

- 실행 결과



실행 결과

채점을 시작합니다.

정확성 테스트

테스트 1 통과 (0.01ms, 10.1MB)
테스트 2 통과 (0.01ms, 10.2MB)
테스트 3 통과 (0.01ms, 10.2MB)
테스트 4 통과 (0.01ms, 10.2MB)
테스트 5 통과 (0.01ms, 10.3MB)
테스트 6 통과 (0.01ms, 10.2MB)
테스트 7 통과 (0.01ms, 10.3MB)
테스트 8 통과 (0.01ms, 10.1MB)
테스트 9 통과 (0.02ms, 10.1MB)
테스트 10 통과 (0.01ms, 10.2MB)
테스트 11 통과 (0.02ms, 10.2MB)
테스트 12 통과 (0.02ms, 10.1MB)
테스트 13 통과 (1.98ms, 10.3MB)

채점 결과

정확성: 100.0

합계: 100.0 / 100.0

- 보완하면 좋을 점(다른 사람의 풀이)

-chr(i): 유니코드 코드 포인트가 정수 i 인 문자를 나타내는 문자열을 반환

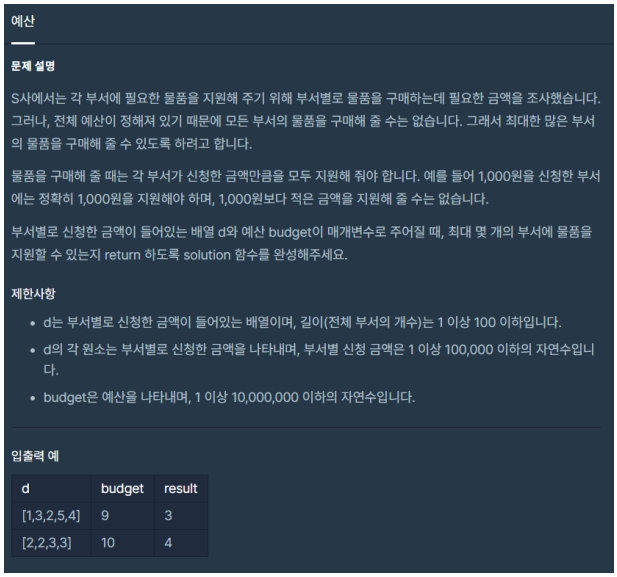
-ord(c): 하나의 유니코드 문자를 나타내는 문자열이 주어지면 해당 문자의 유니코드 코드 포인트를 나타내는 정수 반환

-join(리스트): 리스트의 문자열들을 합치는 역할

```python
def caesar(s, n):
    s = list(s)
    for i in range(len(s)):
        if s[i].isupper():
            s[i]=chr((ord(s[i])-ord('A')+ n)%26+ord('A'))
        elif s[i].islower():
            s[i]=chr((ord(s[i])-ord('a')+ n)%26+ord('a'))

    return "".join(s)
```

※출처:https://velog.io/@joygoround/test-%EC%8B%9C%EC%A0%80-%EC%95%94%ED%98%B8-python

문제9. 예산



예산

문제 설명

S사에서는 각 부서에 필요한 물품을 지원해 주기 위해 부서별로 물품을 구매하는데 필요한 금액을 조사했습니다. 그러나, 전체 예산이 정해져 있기 때문에 모든 부서의 물품을 구매해 줄 수는 없습니다. 그래서 최대한 많은 부서의 물품을 구매해 줄 수 있도록 하려고 합니다.

물품을 구매해 줄 때는 각 부서가 신청한 금액만큼을 모두 지원해 줘야 합니다. 예를 들어 1,000원을 신청한 부서에는 정확히 1,000원을 지원해야 하며, 1,000원보다 적은 금액을 지원해 줄 수는 없습니다.

부서별로 신청한 금액이 들어있는 배열 d와 예산 budget이 매개변수로 주어질 때, 최대 몇 개의 부서에 물품을 지원할 수 있는지 return 하도록 solution 함수를 완성해주세요.

제한사항

- d는 부서별로 신청한 금액이 들어있는 배열이며, 길이(전체 부서의 개수)는 1 이상 100 이하입니다.
- d의 각 원소는 부서별로 신청한 금액을 나타내며, 부서별 신청 금액은 1 이상 100,000 이하의 자연수입니다.
- budget은 예산을 나타내며, 1 이상 10,000,000 이하의 자연수입니다.

입출력 예

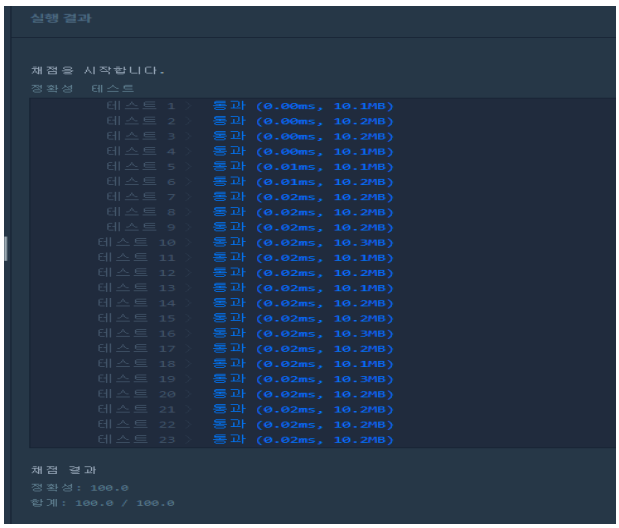
d	budget	result
[1,3,2,5,4]	9	3
[2,2,3,3]	10	4

-나의 풀이

solution.py

```python
def solution(d, budget):
    answer = 0      #answer: 결과값을 저장하는 변수(지원가능한 부서 개수)
    sum=0           #sum: 예산 중간 합

    d.sort()        #d를 오름차순으로 정렬하여 작은 액수부터 계산

    for i in d:
        if sum+i<=budget:   #만약 데이터의 합이 budget(예산)보다 작으면
            answer+=1       # answer(부서개수) 1 증가
            sum+=i          # 중간 합에 데이터 더함
        else:               #그렇지 않으면 (예산초과)
            break           #반복문을 빠져나감
    return answer
```

-실행 결과



```
실행 결과

채점을 시작합니다.
정확성  테스트
        테스트 1    통과 (0.00ms, 10.1MB)
        테스트 2    통과 (0.00ms, 10.2MB)
        테스트 3    통과 (0.00ms, 10.2MB)
        테스트 4    통과 (0.00ms, 10.1MB)
        테스트 5    통과 (0.01ms, 10.1MB)
        테스트 6    통과 (0.01ms, 10.2MB)
        테스트 7    통과 (0.02ms, 10.2MB)
        테스트 8    통과 (0.02ms, 10.2MB)
        테스트 9    통과 (0.02ms, 10.2MB)
        테스트 10   통과 (0.02ms, 10.3MB)
        테스트 11   통과 (0.02ms, 10.1MB)
        테스트 12   통과 (0.02ms, 10.2MB)
        테스트 13   통과 (0.02ms, 10.1MB)
        테스트 14   통과 (0.02ms, 10.2MB)
        테스트 15   통과 (0.02ms, 10.2MB)
        테스트 16   통과 (0.02ms, 10.3MB)
        테스트 17   통과 (0.02ms, 10.2MB)
        테스트 18   통과 (0.02ms, 10.1MB)
        테스트 19   통과 (0.02ms, 10.3MB)
        테스트 20   통과 (0.02ms, 10.2MB)
        테스트 21   통과 (0.02ms, 10.2MB)
        테스트 22   통과 (0.02ms, 10.2MB)
        테스트 23   통과 (0.02ms, 10.2MB)

채점 결과
정확성: 100.0
합계: 100.0 / 100.0
```

문제10. 같은 숫자는 싫어

같은 숫자는 싫어

문제 설명

배열 arr가 주어집니다. 배열 arr의 각 원소는 숫자 0부터 9까지로 이루어져 있습니다. 이때, 배열 arr에서 연속적으로 나타나는 숫자는 하나만 남기고 전부 제거하려고 합니다. 단, 제거된 후 남은 수들을 반환할 때는 배열 arr의 원소들의 순서를 유지해야 합니다. 예를 들면,

- arr = [1, 1, 3, 3, 0, 1, 1] 이면 [1, 3, 0, 1] 을 return 합니다.
- arr = [4, 4, 4, 3, 3] 이면 [4, 3] 을 return 합니다.

배열 arr에서 연속적으로 나타나는 숫자는 제거하고 남은 수들을 return 하는 solution 함수를 완성해 주세요.

제한사항

- 배열 arr의 크기 : 1,000,000 이하의 자연수
- 배열 arr의 원소의 크기 : 0보다 크거나 같고 9보다 작거나 같은 정수

입출력 예

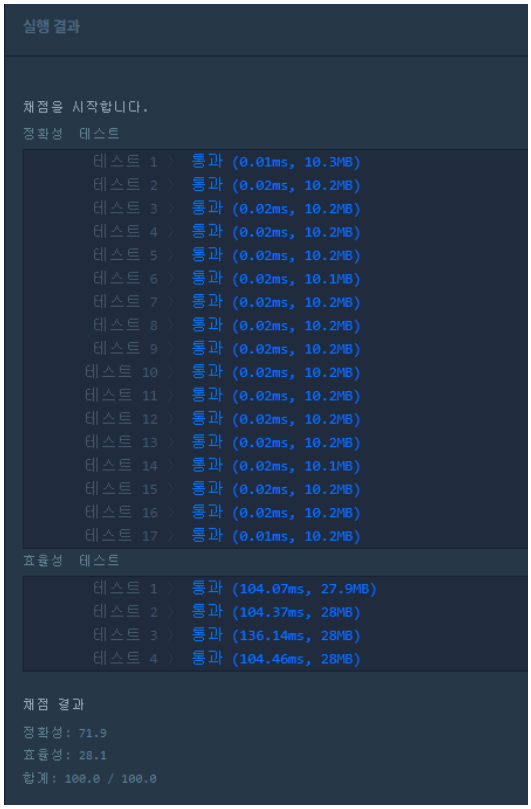
arr	answer
[1,1,3,3,0,1,1]	[1,3,0,1]
[4,4,4,3,3]	[4,3]

- 나의 풀이

solution.py

```python
def solution(arr):
    answer = []                         #answer: 연속적으로 나타나는 숫자를 제거하고 남은 수들을 저장한 배열
    answer.append(arr[0])               #answer(결과 배열)의 첫번째 원소는 항상 arr 배열의 첫번째 원소를 가짐

    for i in range(1,len(arr)):     #arr의 두번째 인덱스 부터  arr배열의 마지막인덱스까지 반복
        if arr[i]!=arr[i-1]:        #arr 원소의 앞, 뒤 값을 비교해서 둘이 다르다면
            answer.append(arr[i])   #arr의 이전 값을 answer배열에 append함

    return answer
```

- 실행 결과



실행 결과

```
채점을 시작합니다.
정확성  테스트
        테스트 1 〉  통과 (0.01ms, 10.3MB)
        테스트 2 〉  통과 (0.02ms, 10.2MB)
        테스트 3 〉  통과 (0.02ms, 10.2MB)
        테스트 4 〉  통과 (0.02ms, 10.2MB)
        테스트 5 〉  통과 (0.02ms, 10.2MB)
        테스트 6 〉  통과 (0.02ms, 10.1MB)
        테스트 7 〉  통과 (0.02ms, 10.2MB)
        테스트 8 〉  통과 (0.02ms, 10.2MB)
        테스트 9 〉  통과 (0.02ms, 10.2MB)
       테스트 10 〉  통과 (0.02ms, 10.2MB)
       테스트 11 〉  통과 (0.02ms, 10.2MB)
       테스트 12 〉  통과 (0.02ms, 10.2MB)
       테스트 13 〉  통과 (0.02ms, 10.2MB)
       테스트 14 〉  통과 (0.02ms, 10.1MB)
       테스트 15 〉  통과 (0.02ms, 10.2MB)
       테스트 16 〉  통과 (0.02ms, 10.2MB)
       테스트 17 〉  통과 (0.01ms, 10.2MB)
효율성  테스트
        테스트 1 〉  통과 (104.07ms, 27.9MB)
        테스트 2 〉  통과 (104.37ms, 28MB)
        테스트 3 〉  통과 (136.14ms, 28MB)
        테스트 4 〉  통과 (104.46ms, 28MB)

채점 결과
정확성: 71.9
효율성: 28.1
합계: 100.0 / 100.0
```

• **과제하면서 자주 발생했던 오류**

오류1: 들여쓰기 오류

-해결: 들여쓰기를 할 때 **처음에 탭 간격을 이용했으면 들여쓰기를 하는 부분 끝까지 탭을 이용해야 하고, 처음에 스페이스 간격을 이용했으면 끝까지 스페이스 간격만 이용해야 한다.**

[출처] [Python] 들여쓰기 오류 (Indentation Error)|**작성자** 산들아이

오류 2: 인덱스 오류

오류 3: TypeError: 'int' object is not callable

메시지	의미
SyntaxError: invalid syntax	구문 오류: 문법에 맞지 않다
IndentationError: expected an indented block	들여쓰기 오류: 들여쓰기를 해야 한다
NameError: name 'a' is not defined	이름 오류: 이름 'a'가 정의되지 않았다
TypeError: unsupported operand type(s) for +: 'a' and 'b'	유형 오류: 연산자 +가 피연자 유형 'a'와 'b'를 지원하지 않는다
TypeError: must be 'a', not 'b'	유형 오류: 유형이 'a'여야 하는데 'b'다
TypeError: 'int' object is not callable	유형 오류: 'int' 유형 객체는 호출할 수 없다
TypeError: f() takes 1 positional arguments but 2 were given	유형 오류: f() 함수는 인자 1개를 전달받는데 2개가 전달되었다
TypeError: 'str' object does not support item assignment	유형 오류: 'str' 유형 객체의 항목에 다른 값을 대입할 수 없다
ValueError: invalid literal for int() with base 10: 'a'	값 오류: 'a'는 int() 함수의 10진법으로 해석할 수 없다
AttributeError: type object 'int' has no attribute 'a'	속성 오류: 'int' 유형 객체에는 'a' 속성이 없다
ZeroDivisionError: division by zero	영 나눗셈 오류: 영(0)으로 나눌 수 없다
KeyError: 'a'	키 오류: (매핑 컬렉션에) 'a' 키가 없다

표 9-2 초보자가 접하기 쉬운 오류 메시지

출처: https://python.bakyeono.net/chapter-9-2.html