문제 1

문제:

> *7로 나누어지지만 5의 배수는 아닌 2000과 3200 사이의 모든 숫자를 찾는 프로그램을 작성하십시오 (두 숫자 포함). 얻은 숫자는 한 줄에 쉼표로 구분된 시퀀스로 인쇄해야 합니다.*

힌트:

> *range(#시작, #끝) 메소드 사용을 고려하십시오.*

주요 저자 솔루션: 파이썬 2

```
l=[]
for i in range(2000, 3201):
    if (i%7==0) and (i%5!=0):
        l.append(str(i))

print ','.join(l)
```

내 솔루션: 파이썬 3

- **for 루프 사용**

```
for i in range(2000,3201):
    if i%7 == 0 and i%5!=0:
        print(i,end=',')
print("\b")
```

- **제너레이터 및 리스트 컴프리헨션 사용**

```
print(*(i for i in range(2000, 3201) if i%7 == 0 and i%5 != 0),
sep=",")
```

문제 2

문제:

> *주어진 숫자의 계승을 계산할 수 있는 프로그램을 작성하십시오. 결과는 한 줄에 쉼표로 구분된 시퀀스로 인쇄해야 합니다. 다음과 같은 입력이 프로그램에 제공된다고 가정합니다: 8 그러면 출력은 다음과 같아야 합니다: 40320*

힌트:

> *질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.*

주요 저자 솔루션: 파이썬 2

```
def fact(x):
    if x == 0:
        return 1
    return x * fact(x - 1)

x = int(raw_input())
print fact(x)
```

내 솔루션: 파이썬 3

- **While 루프 사용**

```
n = int(input()) # input() 함수는 입력을 문자열 유형으로 받습니다.
                 # int()는 정수 유형으로 변환합니다.
fact = 1
i = 1
while i <= n:
    fact = fact * i;
    i = i + 1
print(fact)
```

- **For 루프 사용**

```
n = int(input()) # input() 함수는 입력을 문자열 유형으로 받습니다.
                # int()는 정수 유형으로 변환합니다.
fact = 1
for i in range(1,n+1):
    fact = fact * i
print(fact)
```

- **람다 함수 사용**

```
# 솔루션 작성자: harshraj22

n = int(input())
def shortFact(x): return 1 if x <= 1 else x*shortFact(x-1)
print(shortFact(n))
```

```
'''솔루션 작성자: minnielahoti
'''

while True:
try:
    num = int(input("숫자를 입력하세요: "))
    break
except ValueError as err:
    print(err)

org = num
fact = 1
while num:
    fact = num * fact
    num = num - 1
print(f'{org}의 계승은 {fact}입니다.')
```

```
'''솔루션 작성자: KruthikaSR
'''
from functools import reduce

def fun(acc, item):
    return acc*item

num = int(input())
print(reduce(fun,range(1, num+1), 1))
```

문제 3

문제:

> *주어진 정수 n을 사용하여 (i, i x i)를 포함하는 사전을 생성하는 프로그램을 작성하십시오. 여기서 i는 1과 n (두 숫자 포함) 사이의 정수입니다. 그런 다음 프로그램은 사전을 인쇄해야 합니다. 다음과 같은 입력이 프로그램에 제공된다고 가정합니다: 8*

> *그러면 출력은 다음과 같아야 합니다:*

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}
```

힌트:

> *질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다. **dict()** 사용을 고려하십시오.*

주요 저자 솔루션: 파이썬 2

```
n = int(raw_input())
d = dict()
for i in range(1,n+1):
    d[i] = i * i
print d
```

내 솔루션: 파이썬 3:

- **for 루프 사용**

```
n = int(input())
ans = {}
for i in range (1,n+1):
    ans[i] = i * i
print(ans)
```

- **사전 컴프리헨션 사용**

```
n = int(input())
ans={i : i*i for i in range(1,n+1)}
print(ans)
```

```
'''솔루션 작성자: minnielahoti
   수정자: TheNobleKnight
'''

try:
    num = int(input("숫자를 입력하세요: "))
except ValueError as err:
    print(err)

dictio = dict()
for item in range(num+1):
    if item == 0:
        continue
    else:
    dictio[item] = item * item
print(dictio)
```

```
'''솔루션 작성자: yurbika
   수정자: developer-47
'''

num = int(input("숫자: "))
print(dict(enumerate([i*i for i in range(1, num+1)], 1)))
```

결론

이것은 1 일차에 해결된 문제들입니다. 위의 문제들은 기본 구문 학습자에게 매우 쉽습니다. 제 솔루션에서 몇 가지 쉬운 코딩 방법을 보여주었습니다. 다음 날 새로운 문제에 어떻게 대처하고 공격하는지 봅시다.

다음 날로 가기

토론

문제 4

문제:

> *콘솔에서 쉼표로 구분된 숫자 시퀀스를 입력받아 모든 숫자를 포함하는 리스트와 튜플을 생성하는 프로그램을 작성하십시오. 다음과 같은 입력이 프로그램에 제공된다고 가정합니다:*

```
34,67,55,33,12,98
```

> *그러면 출력은 다음과 같아야 합니다:*

```
['34', '67', '55', '33', '12', '98']
('34', '67', '55', '33', '12', '98')
```

힌트:

> *질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다. **tuple()** 메소드는 리스트를 튜플로 변환할 수 있습니다.*

주요 저자 솔루션: 파이썬 2

```
values = raw_input()
l = values.split(",")
t = tuple(l)
print l
print t
```

내 솔루션: 파이썬 3

```
lst = input().split(',')  # 입력은 문자열로 받아지고 문자열이므로 내장 메소드 split을 가집니다.
                          # split 함수 안의 ','는 ','를 찾을 때마다 분할하고
                          # 입력을 lst 변수에 리스트로 저장합니다.

tpl = tuple(lst)          # tuple 메소드는 리스트를 튜플로 변환합니다.

print(lst)
print(tpl)
```

```
'''솔루션 작성자: minnielahoti
'''
print(tuple(input("쉼표로 구분된 숫자 시리즈를 입력하세요:").split(',')))
```

문제 5

문제:

최소 두 개의 메소드를 가진 클래스를 정의하십시오:

- ***getString**: 콘솔 입력에서 문자열을 가져옵니다.*
- ***printString**: 문자열을 대문자로 인쇄합니다.*

또한 클래스 메소드를 테스트하기 위한 간단한 테스트 함수를 포함하십시오.

힌트:

> *init 메소드를 사용하여 일부 매개변수를 생성하십시오.*

주요 저자 솔루션: 파이썬 2

```
class InputOutString(object):
    def __init__(self):
        self.s = ""

    def get_string(self):
        self.s = raw_input()

    def print_string(self):
        print self.s.upper()

str_obj = InputOutString()
str_obj.get_string()
str_obj.print_string()
```

내 솔루션: 파이썬 3

```
class IOstring():
    def get_string(self):
        self.s = input()

    def print_string(self):
        print(self.s.upper())

xx = IOstring()
xx.get_string()
xx.print_string()
```

문제 6

문제:

주어진 공식에 따라 값을 계산하고 인쇄하는 프로그램을 작성하십시오:

*Q = [(2 * C * D)/H]의 제곱근*

다음은 C와 H의 고정 값입니다:

C는 50입니다. H는 30입니다.

D는 쉼표로 구분된 시퀀스로 프로그램에 입력해야 하는 변수입니다. 예를 들어 다음과 같은 쉼표로 구분된 입력 시퀀스가 프로그램에 제공된다고 가정합니다:

```
100,150,180
```

프로그램의 출력은 다음과 같아야 합니다:

```
18,22,24
```

힌트:

> ***받은 출력이 십진수 형태인 경우 가장 가까운 값으로 반올림해야 합니다 (예: 받은 출력이 26.0이면 26으로 인쇄해야 함). 질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.***

주요 저자 솔루션: 파이썬 2

```
import math
c = 50
h = 30
value = []
items = [x for x in raw_input().split(',')]
for d in items:
    value.append(str(int(round(math.sqrt(2*c*float(d)/h)))))

print ','.join(value)
```

내 솔루션: 파이썬 3

```
from math import sqrt # *를 사용하여 모두 가져오는 것은 좋지 않으므로 특정 함수만 가져옵니다.
                      # 나쁜 습관입니다.

C,H = 50,30

def calc(D):
    return sqrt((2*C*D)/H)

D = [int(i) for i in input().split(',')] # 쉼표 위치에서 분할하고 리스트에 설정합니다.
D = [int(i) for i in D]   # 문자열을 정수로 변환합니다.
D = [calc(i) for i in D]  # D의 모든 항목에 대해 calc 메소드로 부동 소수점 값을 반환합니다.
D = [round(i) for i in D] # 모든 부동 소수점 값을 반올림합니다.
D = [str(i) for i in D]   # join 연산을 적용할 수 있도록 모든 정수를 문자열로 변환합니다.

print(",".join(D))
```

또는

```
from math import sqrt

C,H = 50,30

def calc(D):
    return sqrt((2*C*D)/H)

D = input().split(',')                          # 쉼표 위치에서 분할하고 리스트에 설정합니다.
D = [str(round(calc(int(i)))) for i in D]  # 컴프리헨션 메소드를 사용합니다. 이전 코드의 순서대로 작동합니다.
print(",".join(D))
```

또는

```
from math import sqrt
C,H = 50,30

def calc(D):
    return sqrt((2*C*D)/H)

print(",".join([str(int(calc(int(i)))) for i in input().split(',')]))
```

또는

```
from math import * # 모든 수학 함수 가져오기
C,H = 50,30

def calc(D):
    D = int(D)
    return str(int(sqrt((2*C*D)/H)))

D = input().split(',')
D = list(map(calc,D))    # D에 calc 함수를 적용하고 리스트로 저장합니다.
print(",".join(D))
```

```
'''솔루션 작성자: parian5
'''
from math import sqrt
C, H = 50, 30
mylist = input().split(',')
print(*(round(sqrt(2*C*int(D)/H)) for D in mylist), sep=",")
```

```
'''솔루션 작성자: saxenaharsh24
'''

my_list = [int(x) for x in input('').split(',')]
C, H, x = 50, 30, []

for D in my_list:
    Q = ((2*C*D)/H)**(1/2)
    x.append(round(Q))

print(','.join(map(str, x)))
```

문제 7

문제:

**_2개의 숫자 X, Y를 입력받아 2차원 배열을 생성하는 프로그램을 작성하십시오. 배열의 i번째 행과 j번째 열의 요소 값은 i * j여야 합니다.*

참고: i=0,1.., X-1; j=0,1,¡Y-1. 다음과 같은 입력이 프로그램에 제공된다고 가정합니다: 3,5

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]
```

힌트:

> *질문에 입력 데이터가 제공되는 경우 쉼표로 구분된 형태로 콘솔 입력으로 간주해야 합니다.*

주요 저자 솔루션: 파이썬 2

```
input_str = raw_input()
dimensions = [int(x) for x in input_str.split(',')]
row_num = dimensions[0]
col_num = dimensions[1]
multilist = [[0 for col in range(col_num)] for row in
range(row_num)]

for row in range(row_num):
    for col in range(col_num):
        multilist[row][col] = row * col

print multilist
```

내 솔루션: 파이썬 3

```
x,y = map(int,input().split(','))
lst = []

for i in range(x):
    tmp = []
    for j in range(y):
        tmp.append(i*j)
    lst.append(tmp)

print(lst)
```

또는

```
x,y = map(int,input().split(','))
lst = [[i*j for j in range(y)] for i in range(x)]
print(lst)
```

문제 8

문제:

쉼표로 구분된 단어 시퀀스를 입력받아 알파벳순으로 정렬한 후 쉼표로 구분된 시퀀스로 단어를 인쇄하는 프로그램을 작성하십시오.

다음과 같은 입력이 프로그램에 제공된다고 가정합니다:

```
without,hello,bag,world
```

그러면 출력은 다음과 같아야 합니다:

```
bag,hello,without,world
```

힌트:

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

주요 저자 솔루션: 파이썬 2

```
items = [x for x in raw_input().split(',')]
items.sort()
print ','.join(items)
```

내 솔루션: 파이썬 3

```
lst = input().split(',')
lst.sort()
print(",".join(lst))
```

```
'''솔루션 작성자: Poonam-glitch
'''
def my_func(e):
    return e[0]

my_list = input('쉼표로 구분된 문자열을 입력하세요: ').split(",")
my_list.sort(key=my_func)
print(",".join(my_list))
```

문제 9

문제:

> *여러 줄의 시퀀스를 입력받아 문장의 모든 문자를 대문자로 만든 후 줄을 인쇄하는 프로그램을 작성하십시오.*

> *다음과 같은 입력이 프로그램에 제공된다고 가정합니다:*

```
Hello world
Practice makes perfect
```

그러면 출력은 다음과 같아야 합니다:

```
HELLO WORLD
PRACTICE MAKES PERFECT
```

힌트:

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

주요 저자 솔루션: 파이썬 2

```
lines = []
while True:
    s = raw_input()
    if s:
        lines.append(s.upper())
    else:
        break

for sentence in lines:
    print sentence
```

내 솔루션: 파이썬 3

```
lst = []

while True:
    x = input()
    if len(x)==0:
        break
    lst.append(x.upper())

for line in lst:
    print(line)
```

또는

```
def user_input():
    while True:
        s = input()
        if not s:
            return
        yield s

for line in map(str.upper, user_input()):
    print(line)


'''솔루션 작성자: hajimalung baba
'''
def inputs():
    while True:
        string = input()
        if not string:
            return
        yield string

print(*(line.upper() for line in inputs()),sep='\n')
```

이전 날로 가기

다음 날로 가기

토론

문제 10

문제

공백으로 구분된 단어 시퀀스를 입력받아 중복된 단어를 모두 제거하고 영숫자 순으로 정렬한 후 단어를 인쇄하는 프로그램을 작성하십시오.

다음과 같은 입력이 프로그램에 제공된다고 가정합니다:

```
hello world and practice makes perfect and hello world again
```

그러면 출력은 다음과 같아야 합니다:

```
again and hello makes perfect practice world
```

힌트:

*질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다. 중복된 데이터를 자동으로 제거하기 위해 **set** 컨테이너를 사용한 다음 **sorted()**를 사용하여 데이터를 정렬합니다.*

주요 저자 솔루션: 파이썬 2

```
s = raw_input()
words = [word for word in s.split(" ")]
print " ".join(sorted(list(set(words))))
```

내 솔루션: 파이썬 3

```
word = input().split()

for i in word:
    if word.count(i) > 1:    # count 함수는 인수로 전달된 요소의 총 반복
횟수를 반환합니다.
        word.remove(i)     # 호출당 정확히 하나의 요소를 제거합니다.

word.sort()
print(" ".join(word))
```

또는

```
word = input().split()
[word.remove(i) for i in word if word.count(i) > 1 ]    # 컴프리헨션 메
소드를 사용한 제거 작업
word.sort()
print(" ".join(word))
```

또는

```
word = sorted(list(set(input().split())))                # 입력 문자열
분할 -> 고유 요소를 저장하기 위해 set()으로 변환
                                                         # -> 정렬을 적
용할 수 있도록 리스트로 변환
print(" ".join(word))
```

```
'''솔루션 작성자: Sukanya-Mahapatra
'''
inp_string = input("문자열 입력: ").split()
out_string = []
for words in inp_string:
    if words not in out_string:
        out_string.append(words)
print(" ".join(sorted(out_string)))
```

문제 11

문제

쉼표로 구분된 4자리 이진수 시퀀스를 입력받아 5로 나누어지는지 확인하는 프로그램을 작성하십시오. 5로 나누어지는 숫자는 쉼표로 구분된 시퀀스로 인쇄해야 합니다.

예시:

```
0100,0011,1010,1001
```

그러면 출력은 다음과 같아야 합니다:

```
1010
```

참고: 데이터는 콘솔을 통해 입력된다고 가정합니다.

힌트:

> *질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.*

주요 저자 솔루션: 파이썬 2

```
value = []
items=[x for x in raw_input().split(',')]
for p in items:
    intp = int(p,2)
    if not intp % 5:
        value.append(p)

print ','.join(value)
```

내 솔루션: 파이썬 3

```
def check(x):                           # 이진수를 정수로 변환하고 5로 나누어
지면 0을 반환합니다.
    total,pw = 0,1
    reversed(x)

    for i in x:
        total+=pw * (ord(i) - 48)    # ord() 함수는 ASCII 값을 반환합니
다.
        pw*=2
    return total % 5

data = input().split(",")           # 여기서 입력을 받고 ',' 위치에서 분
할합니다.
lst = []

for i in data:
    if check(i) == 0:               # 0이 발견되면 0으로 나누어지고 리스
트에 추가됩니다.
        lst.append(i)

print(",".join(lst))
```

또는

```
def check(x):                       # check 함수는 5로 나누어지면 True를 반
환합니다.
    return int(x,2)%5 == 0          # int(x,b)는 x를 문자열로, b를 밑으로
사용하여
                                    # 십진수로 변환합니다.
data = input().split(',')

data = list(filter(check,data)) # filter(func,object) 함수에서
'check' 함수에 의해 True로 확인되면 'data'에서 요소를 선택합니다.
print(",".join(data))
```

또는

```
data = input().split(',')
data = list(filter(lambda i:int(i,2)%5==0,data))    # 람다는 한 줄짜리
함수를 작성하는 데 도움이 되는 연산자입니다.
print(",".join(data))
```

```
'''솔루션 작성자: nikitaMogilev
'''
data = input().split(',')
data = [num for num in data if int(num, 2) % 5 == 0]
print(','.join(data))


'''솔루션 작성자: hajimalung baba
'''
print(*(binary for binary in input().split(',') if
int(binary,base=2)%5==0))
```

문제 12

문제:

1000과 3000 사이(두 숫자 포함)의 모든 숫자 중에서 숫자의 각 자릿수가 짝수인 숫자를 찾는 프로그램을 작성하십시오. 얻은 숫자는 한 줄에 쉼표로 구분된

시퀀스로 인쇄해야 합니다.

힌트:

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

주요 저자 솔루션: 파이썬 2

```
values = []
for i in range(1000, 3001):
    s = str(i)
    if (int(s[0])%2 == 0) and (int(s[1])%2 == 0) and (int(s[2])%2 ==
0) and (int(s[3])%2 == 0):
        values.append(s)
print ",".join(values)
```

내 솔루션: 파이썬 3

```
lst = []

for i in range(1000,3001):
    flag = 1
    for j in str(i):          # 모든 정수 i는 문자열로 변환됩니다.
        if ord(j)%2 != 0:     # ord는 ASCII 값을 반환하고 j는 i의 모든
자릿수입니다.
            flag = 0          # 홀수 자릿수가 발견되면 플래그는 0이 됩니
다.
    if flag == 1:
        lst.append(str(i))    # i는 문자열로 리스트에 저장됩니다.

print(",".join(lst))
```

또는

```
def check(element):
    return all(ord(i)%2 == 0 for i in element)  # all은 요소의 모든 자릿수 i가 짝수이면 True를 반환합니다.

lst = [str(i) for i in range(1000,3001)]        # 주어진 모든 숫자의 리스트를 문자열 데이터 유형으로 만듭니다.
lst = list(filter(check,lst))                   # filter는 check 조건이 실패하면 리스트에서 요소를 제거합니다.
print(",".join(lst))
```

또는

```
lst = [str(i) for i in range(1000,3001)]
lst = list(filter(lambda i:all(ord(j)%2 == 0 for j in i), lst))   # 람다를 사용하여 filter 함수 내에 함수를 정의합니다.
print(",".join(lst))
```

```
'''솔루션 작성자: nikitaMogilev
'''
# 각 숫자의 자릿수를 람다 함수로 매핑하고 모든 자릿수가 짝수인지 확인합니다.
# str(num)은 map() 및 join()을 통해 숫자를 반복할 수 있는 기회를 제공합니다.
print(','.join([str(num) for num in range(1000, 3001) if
all(map(lambda num: int(num) % 2 == 0, str(num)))]))


'''솔루션 작성자: hajimalung
'''
from functools import reduce
# reduce를 사용하여 숫자에 짝수 자릿수만 있는지 확인합니다.
def is_even_and(bool_to_compare,num_as_char):
    return int(num_as_char)%2==0 and bool_to_compare

print(*(i for i in range(1000,3001) if
reduce(is_even_and,str(i),True)),sep=',')
```

문제 13

문제:

문장을 입력받아 글자 수와 숫자 수를 계산하는 프로그램을 작성하십시오.

다음과 같은 입력이 프로그램에 제공된다고 가정합니다:

```
hello world! 123
```

그러면 출력은 다음과 같아야 합니다:

```
LETTERS 10
DIGITS 3
```

힌트:

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

주요 저자 솔루션: 파이썬 2

```
s = raw_input()
d = {"DIGITS":0, "LETTERS":0}
for c in s:
    if c.isdigit():
        d["DIGITS"]+=1
    elif c.isalpha():
        d["LETTERS"]+=1
    else:
        pass
print "LETTERS", d["LETTERS"]
print "DIGITS", d["DIGITS"]
```

내 솔루션: 파이썬 3

```
word = input()
letter,digit = 0,0

for i in word:
    if ('a'<=i and i<='z') or ('A'<=i and i<='Z'):
        letter+=1
    if '0'<=i and i<='9':
        digit+=1

print("LETTERS {0}\nDIGITS {1}".format(letter,digit))
```

또는

```
word = input()
letter, digit = 0,0

for i in word:
    if i.isalpha(): # 알파벳이면 True를 반환합니다.
        letter += 1
    elif i.isnumeric(): # 숫자이면 True를 반환합니다.
        digit += 1
print(f"LETTERS {letter}\n{digits}") # 두 가지 다른 유형의 형식 지정 방법이 두 솔루션 모두에 표시됩니다.
```

```
''' 솔루션 작성자: popomaticbubble
'''
import re

input_string = input('> ')
print()
counter = {"LETTERS":len(re.findall("[a-zA-Z]", input_string)),
"NUMBERS":len(re.findall("[0-9]", input_string))}

print(counter)
```

```
'''솔루션 작성자: MarkisLandis
'''

sen = input("").split(" ")
alp, digit = 0, 0

for item in sen:
    lst = [char for char in item]
    for j in lst:
        if 64 < ord(j) < 123:
            alp += 1
        if j.isdigit():
            digit += 1
print(f"LETTERS : {alp} \n DIGITS : {digit}")


'''솔루션 작성자: hajimalung
'''
# reduce를 사용하여 계산합니다.
from functools import reduce

def count_letters_digits(counters,char_to_check):
    counters[0] += char_to_check.isalpha()
    counters[1] += char_to_check.isnumeric()
    return counters

print('LETTERS {0}\nDIGITS
{1}'.format(*reduce(count_letters_digits,input(),[0,0])))
```

결론

위의 모든 문제는 대부분 문자열 관련 문제입니다. 솔루션의 주요 부분에는 문자열 관련 함수와 코드를 더 짧은 형태로 작성하기 위한 컴프리헨션 메소드가 포함됩니다.

이전 날로 가기

다음 날로 가기

토론

문제 14

문제:

문장을 입력받아 대문자 수와 소문자 수를 계산하는 프로그램을 작성하십시오.

다음과 같은 입력이 프로그램에 제공된다고 가정합니다:

```
Hello world!
```

그러면 출력은 다음과 같아야 합니다:

```
UPPER CASE 1
LOWER CASE 9
```

힌트:

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

주요 저자 솔루션: 파이썬 2

```
s = raw_input()
d = {"UPPER CASE":0, "LOWER CASE":0}
for c in s:
    if c.isupper():
        d["UPPER CASE"]+=1
    elif c.islower():
        d["LOWER CASE"]+=1
    else:
        pass
print "UPPER CASE", d["UPPER CASE"]
print "LOWER CASE", d["LOWER CASE"]
```

내 솔루션: 파이썬 3

```
word = input()
upper,lower = 0,0

for i in word:
    if 'a'<=i and i<='z' :
        lower+=1
    if 'A'<=i and i<='Z':
        upper+=1

print("UPPER CASE {0}\nLOWER CASE {1}".format(upper,lower))
```

또는

```
word = input()
upper,lower = 0,0

for i in word:
        lower+=i.islower() # islower()는 소문자이면 1 (True)을, 아니면
0 (False)을 반환합니다. 파이썬에서 True는 1로, False는 0으로 처리됩니다.
        upper+=i.isupper() # isupper()는 대문자이면 1 (True)을, 아니면
0 (False)을 반환합니다.

print("UPPER CASE {0}\nLOWER CASE {1}".format(upper,lower))
```

또는

```
word = input()
upper = sum(1 for i in word if i.isupper()) # sum 함수는 조건이 True이
면 1을 누적하여 합산합니다.
lower = sum(1 for i in word if i.islower())

print("UPPER CASE {0}\nLOWER CASE {1}".format(upper,lower))
```

또는

```
# 솔루션 작성자: Amitewu

string = input("문장을 입력하세요")
upper = 0
lower = 0
for x in string:
    if x.isupper() == True:
        upper += 1
    if x.islower() == True:
        lower += 1

print("UPPER CASE: ", upper)
print("LOWER CASE: ", lower)
```

문제 15

문제:

> ***주어진 숫자를 a 값으로 사용하여 a+aa+aaa+aaaa 값을 계산하는 프로그램을 작성하십시오.***

> ***다음과 같은 입력이 프로그램에 제공된다고 가정합니다:***

그러면 출력은 다음과 같아야 합니다:

```
11106
```

힌트:

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

주요 저자 솔루션: 파이썬 2

```
a = raw_input()
n1 = int( "%s" % a )
n2 = int( "%s%s" % (a,a) )
n3 = int( "%s%s%s" % (a,a,a) )
n4 = int( "%s%s%s%s" % (a,a,a,a) )
print n1+n2+n3+n4
```

내 솔루션: 파이썬 3

```
a = input()
total,tmp = 0,str()          # 정수와 빈 문자열 초기화

for i in range(4):
    tmp+=a                 # 'a'를 'tmp'에 연결합니다.
    total+=int(tmp)        # 문자열 유형을 정수 유형으로 변환합니다.

print(total)
```

또는

```
a = input()
total = int(a) + int(2*a) + int(3*a) + int(4*a)  # N*a=Na, 예를 들어 a="23", 2*a="2323", 3*a="232323"
print(total)
```

```
'''솔루션 작성자: ChichiLovesDonkeys
'''
from functools import reduce
x = input('숫자를 입력하세요:')
reduce(lambda x, y: int(x) + int(y), [x*i for i in range(1,5)])
```

```
'''솔루션 작성자: lcastrooliveira
'''
def question_15(string_digit):
    return sum(int(string_digit * n) for n in range(1, 5))

inp = input()
print(question_15(inp))
```

```
'''솔루션 작성자: apenam7
'''
a = input()
print(sum(int(i*a) for i in range(1,5)))
```

이전 날로 가기

다음 날로 가기

토론

문제 16

문제:

리스트 컴프리헨션을 사용하여 리스트의 각 홀수를 제곱하십시오. 리스트는 쉼표로 구분된 숫자 시퀀스로 입력됩니다. >다음과 같은 입력이 프로그램에 제공된다고 가정합니다:

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

그러면 출력은 다음과 같아야 합니다:

```
1,9,25,49,81
```

힌트:

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

주요 저자 솔루션: 파이썬 2

```
## 저자의 솔루션이 정확하지 않아 여기에 포함하지 않았습니다.
```

내 솔루션: 파이썬 3

```
lst = [str(int(i)**2) for i in input().split(',') if int(i) % 2]
print(",".join(lst))
```

```
'''솔루션 작성자: shagun'''
# 홀수 제곱

lst = input().split(',')      # 쉼표 위치에서 분할하고 리스트에 설정합니다.

seq = []
lst = [int(i) for i in lst]  # 문자열을 정수로 변환합니다.
for i in lst:
        if i%2 != 0:
                i = i*i
                seq.append(i)


seq = [str(i) for i in seq]   # join 연산을 적용할 수 있도록 모든 정수를
문자열로 변환합니다.
print(",".join(seq))


'''솔루션 작성자: Jack'''
seq = input().split(',')
lst = [int(i) for i in seq]
def flt(i):                                                  # 필터 함수 정의
    return i % 2 != 0
result_l = [str(i * i) for i in filter(flt,lst)]
print(",".join(result_l))
```

테스트 케이스와 솔루션에 실수가 있었는데 @dwedigital의 도움으로 알림을 받고 수정했습니다. 그에게 진심으로 감사합니다.

문제 17

문제:

콘솔 입력에서 트랜잭션 로그를 기반으로 은행 계좌의 순 금액을 계산하는 프로그램을 작성하십시오. 트랜잭션 로그 형식은 다음과 같습니다:

```
D 100
W 200
```

- D는 입금을 의미하고 W는 출금을 의미합니다.

다음과 같은 입력이 프로그램에 제공된다고 가정합니다:

```
D 300
D 300
W 200
D 100
```

그러면 출력은 다음과 같아야 합니다:

```
500
```

힌트:

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

주요 저자 솔루션: 파이썬 2

```
import sys
netAmount = 0
while True:
    s = raw_input()
    if not s:
        break
    values = s.split(" ")
    operation = values[0]
    amount = int(values[1])
    if operation=="D":
        netAmount+=amount
    elif operation=="W":
        netAmount-=amount
    else:
        pass
print netAmount
```

내 솔루션: 파이썬 3

```
total = 0
while True:
    s = input().split()
    if not s:             # 문자열이 비어 있으면 중단합니다.
        break
    cm,num = map(str,s)    # 두 개의 입력이 문자열 데이터 유형으로 cm과
num에 분배됩니다.

    if cm=='D':
        total+=int(num)
    if cm=='W':
        total-=int(num)

print(total)
```

```
'''솔루션 작성자: leonedott'''

lst = []
while True:
  x = input()
  if len(x)==0:
    break
  lst.append(x)

balance = 0
for item in lst:
  if 'D' in item:
    balance += int(item.strip('D '))
  if 'W' in item:
    balance -= int(item.strip('W '))
print(balance)
```

```
'''솔루션 작성자: AlexanderSro'''

account = 0
while True:
    action = input("입금/출금/잔액/종료? D/W/B/Q: ").lower()
    if action == "d":
        deposit = input("얼마나 입금하시겠습니까? ")
        account = account + int(deposit)
    elif action == "w":
        withdrow = input("얼마나 출금하시겠습니까? ")
        account = account - int(withdrow)
    elif action == "b":
        print(account)
    else:
        quit()
```

```
'''솔루션 작성자: ShalomPrinz
'''
lines = []
while True:
    loopInput = input()
    if loopInput == "done": # "done" 입력 시 종료
        break
    else:
        lines.append(loopInput)

lst = list(int(i[2:]) if i[0] == 'D' else -int(i[2:]) for i in
lines)
print(sum(lst))
```

```
'''솔루션 작성자: popomaticbubble
'''
transactions = []

while True:
    text = input("> ")
    if text:
        text = text.strip('D ')
        text = text.replace('W ', '-')
        transactions.append(text)
    else:
        break

transactions = (int(i) for i in transactions)
balance = sum(transactions)
print(f"잔액은 {balance}입니다.")
```

```
'''솔루션 작성자: ChichiLovesDonkeys
'''

money = 0
while 1:
    trans = input().split(' ')
    if trans[0] == 'D':
        money = money + int(trans[1])
    elif trans[0] == 'W':
        money = money - int(trans[1])
    elif input() == '': # 빈 입력 시 종료
        break
    print(f'현재 잔액은: {money}입니다.')
```

이전 날로 가기

다음 날로 가기

토론

문제 18

문제:

한 웹사이트에서 사용자에게 등록을 위해 사용자 이름과 비밀번호를 입력하도록 요구합니다. 사용자가 입력한 비밀번호의 유효성을 확인하는 프로그램을 작성하십시오.

다음은 비밀번호 확인 기준입니다:

- *[a-z] 사이의 문자 최소 1개*
- *[0-9] 사이의 숫자 최소 1개*
- *[A-Z] 사이의 문자 최소 1개*
- *[$#@] 중 문자 최소 1개*
- *거래 비밀번호 최소 길이: 6*
- *거래 비밀번호 최대 길이: 12*

프로그램은 쉼표로 구분된 비밀번호 시퀀스를 입력받아 위의 기준에 따라 확인해야 합니다. 기준을 충족하는 비밀번호는 각각 쉼표로 구분하여 인쇄해야 합니다.

예시

다음 비밀번호가 프로그램에 입력으로 주어지면:

```
ABd1234@1,a F1#,2w3E*,2We3345
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
ABd1234@1
```

힌트:

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

주요 저자 솔루션: 파이썬 2

```
import re
value = []
items = [x for x in raw_input().split(',')]
for p in items:
    if len(p) < 6 or len(p) > 12:
        continue
    else:
        pass
    if not re.search("[a-z]",p):
        continue
    elif not re.search("[0-9]",p):
        continue
    elif not re.search("[A-Z]",p):
        continue
    elif not re.search("[$#@]",p):
        continue
    elif re.search("\s",p):
        continue
    else:
        pass
    value.append(p)
print ",".join(value)
```

내 솔루션: 파이썬 3

```
def is_low(x):                  # 문자열에 소문자가 있으면 True를 반환합
니다.
    for i in x:
        if 'a'<=i and i<='z':
            return True
    return  False

def is_up(x):                   # 문자열에 대문자가 있으면 True를 반환합
니다.
    for i in x:
        if 'A'<= i and i<='Z':
            return True
    return  False

def is_num(x):                  # 문자열에 숫자가 있으면 True를 반환합니
다.
    for i in x:
        if '0'<=i and i<='9':
            return True
    return  False

def is_other(x):                # 문자열에 "$#@" 중 하나라도 있으면 True
를 반환합니다.
    for i in x:
        if i=='$' or i=='#' or i=='@':
            return True
    return False

s = input().split(',')
lst = []

for i in s:
    length = len(i)
    if 6 <= length and length <= 12 and is_low(i) and is_up(i) and
is_num(i) and is_other(i):   # 모든 요구 사항이 충족되었는지 확인합니다.
        lst.append(i)

print(",".join(lst))
```

또는

```
def check(x):
    cnt = (6<=len(x) and len(x)<=12) # cnt는 초기에 조건 (6<=len(x)
and len(x)<=12)이 참이면 1, 거짓이면 0이 됩니다.
    for i in x:
        if i.isupper():
            cnt+=1 # 대문자가 있으면 cnt를 1 증가시킵니다.
            break
    for i in x:
        if i.islower():
            cnt+=1 # 소문자가 있으면 cnt를 1 증가시킵니다.
            break
    for i in x:
        if i.isnumeric():
            cnt+=1 # 숫자가 있으면 cnt를 1 증가시킵니다.
            break
    for i in x:
        if i=='@' or i=='#'or i=='$':
            cnt+=1 # 특수문자가 있으면 cnt를 1 증가시킵니다.
            break
    return cnt == 5                # 총 5가지 조건이 모두 충족되면 True를
반환합니다. (길이 조건 + 4가지 문자 유형 조건)

s = input().split(',')
lst = filter(check,s)              # filter 함수는 check() 함수에 의해
True를 반환하는 s의 단어를 선택합니다.
print(",".join(lst))
```

또는

```
import  re

s = input().split(',')
lst = []

for i in s:
    cnt = 0
    cnt+=(6<=len(i) and len(i)<=12)
    cnt+=bool(re.search("[a-z]",i))      # 여기서 re 모듈에는 re.search() 함수가 포함되어 있으며, 이 함수는 패턴 문자열 i가 [a-z]/[A-Z]/[0-9]/[@#$] 문자 중 하나와 일치하는 위치의 객체 정보를 반환합니다.
    cnt+=bool(re.search("[A-Z]",i))      # 일치하는 항목이 하나도 없으면 NONE을 반환하며, 이는 부울 표현식에서 False로 변환됩니다. 일치하는 항목이 있으면 True입니다.
    cnt+=bool(re.search("[0-9]",i))
    cnt+=bool(re.search("[@#$]",i))
    if cnt == 5:
        lst.append(i)

print(",".join(lst))
```

```
'''솔루션 작성자: pratikb0501
'''
import re
a = input('비밀번호 입력: ').split(',')
pass_pattern = re.compile(r"^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z])(?=.*[$#@]).{6,12}$") # 정규 표현식: 숫자, 소문자, 대문자, 특수문자($#@)를 각각 하나 이상 포함하고 길이는 6~12자여야 함
for i in a:
    if pass_pattern.fullmatch(i): # fullmatch는 전체 문자열이 패턴과 일치하는지 확인
        print(i)
```

또는

```python
import re
def multiple (patterns, string):
  for i in patterns:
    if not re.search(i, string): # 각 패턴에 대해 문자열 검색
      return False

  if 6 <= len(string) <= 12: # 길이 확인
    return True
  else:
    return False
x = str(input("비밀번호 입력: "))
patterns = [r"[a-z]", r"[A-Z]", r"[0-9]", r"[$|#|@]"] # 패턴 리스트
print(multiple(patterns, x))
```

문제 19

문제:

(이름, 나이, 점수) 튜플을 오름차순으로 정렬하는 프로그램을 작성해야 합니다. 여기서 이름은 문자열이고 나이와 점수는 숫자입니다. 튜플은 콘솔을 통해 입력됩니다. 정렬 기준은 다음과 같습니다:

- *1: 이름 기준 정렬*
- *2: 그런 다음 나이 기준 정렬*
- *3: 그런 다음 점수 기준 정렬*

우선순위는 이름 > 나이 > 점수입니다.

다음 튜플이 프로그램에 입력으로 주어지면:

```
Tom,19,80
John,20,90
Jony,17,91
Jony,17,93
Json,21,85
```

> *그러면 프로그램의 출력은 다음과 같아야 합니다:*

```
[('John', '20', '90'), ('Jony', '17', '91'), ('Jony', '17', '93'),
('Json', '21', '85'), ('Tom', '19', '80')]
```

힌트:

> *질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다. 여러 정렬 키를 사용하려면 itemgetter를 사용합니다.*

주요 저자 솔루션: 파이썬 2

```
from operator import itemgetter, attrgetter

l = []
while True:
    s = raw_input()
    if not s:
        break
    l.append(tuple(s.split(",")))

print sorted(l, key=itemgetter(0,1,2))
```

내 솔루션: 파이썬 3

```
lst = []
while True:
    s = input().split(',')
    if not s[0]:                            # 빈 입력이면 중단합니다.
        break
    lst.append(tuple(s))

lst.sort(key= lambda x:(x[0],int(x[1]),int(x[2])))  # 여기서 키는 람다로 정의되고 데이터는 요소 우선순위 0>1>2에 따라 오름차순으로 정렬됩니다.
print(lst)
```

결론

위의 문제를 풀기 전에는 re(정규 표현식) 모듈과 그 사용법에 대해 전혀 몰랐습니다. 여러 키로 정렬하는 방법도 몰랐습니다. 이러한 문제를 다양한 방식으로 해결하기 위해 해당 구문을 탐색하고 배워야 했습니다. re 모듈에는 흥미로운 내용이 많지만 그중 많은 것을 이해하는 데 약간의 어려움을 겪었습니다.

이전 날로 가기

다음 날로 가기

토론

문제 20

문제:

주어진 범위 0과 n 사이에서 7로 나누어지는 숫자를 반복할 수 있는 제너레이터가 있는 클래스를 정의하십시오.

다음과 같은 입력이 프로그램에 제공된다고 가정합니다:

```
7
```

그러면 출력은 다음과 같아야 합니다:

```
0
7
14
```

힌트:

클래스, 함수 및 컴프리헨션 사용을 고려하십시오.

주요 저자 솔루션: 파이썬 2

이 문제에 대한 솔루션 코드는 언급된 문제와 관련이 없었고 함수를 호출하는 동안 입력 오류가 있었습니다.

솔루션: 파이썬 3

```
'''솔루션 작성자: ShalomPrinz
'''
class MyGen():
    def by_seven(self, n): # n까지 7의 배수를 생성하는 제너레이터
        for i in range(0, int(n/7) + 1): # 0부터 n/7까지 반복 (n/7은 n까지 7의 배수 개수)
            yield i * 7 # 현재 숫자에 7을 곱한 값을 반환

for i in MyGen().by_seven( int(input('숫자를 입력하세요... ')) ): # MyGen 클래스의 인스턴스를 만들고 by_seven 메소드 호출
    print(i)
```

```
'''솔루션 작성자: Seawolf159
'''
class Divisible:

    def by_seven(self, n): # n까지 7의 배수를 생성하는 제너레이터
        for number in range(1,n + 1): # 1부터 n까지 반복
            if number % 7 == 0: yield number # 7로 나누어지면 반환


divisible = Divisible() # Divisible 클래스의 인스턴스 생성
generator = divisible.by_seven(int(input("숫자를 입력하세요. --> ")))
# by_seven 메소드 호출하여 제너레이터 얻기
for number in generator: # 제너레이터 반복
    print(number)
```

문제 21

문제:

로봇이 원점 (0,0)에서 시작하여 평면에서 움직입니다. 로봇은 주어진 단계만큼 UP, DOWN, LEFT, RIGHT 방향으로 움직일 수 있습니다. 로봇 이동의 흔적은 다음과 같습니다:

```
UP 5
DOWN 3
LEFT 3
RIGHT 2
```

방향 뒤의 숫자는 단계입니다. 일련의 이동 후 현재 위치와 원점 사이의 거리를 계산하는 프로그램을 작성하십시오. 거리가 부동 소수점이면 가장 가까운 정수를 인쇄하십시오. 예시: 다음 튜플이 프로그램에 입력으로 주어지면:

```
UP 5
DOWN 3
LEFT 3
RIGHT 2
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
2
```

힌트:

> *질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다. 여기서 거리는 유클리드 거리를 나타냅니다. sqrt 함수를 사용하려면 math 모듈을 가져옵니다.*

주요 저자 솔루션: 파이썬 2

```
import math
pos = [0,0]
while True:
    s = raw_input()
    if not s:
        break
    movement = s.split(" ")
    direction = movement[0]
    steps = int(movement[1])
    if direction=="UP":
        pos[0]+=steps
    elif direction=="DOWN":
        pos[0]-=steps
    elif direction=="LEFT":
        pos[1]-=steps
    elif direction=="RIGHT":
        pos[1]+=steps
    else:
        pass

print int(round(math.sqrt(pos[1]**2+pos[0]**2)))
```

내 솔루션: 파이썬 3

```python
import  math

x,y = 0,0
while True:
    s = input().split()
    if not s: # 입력이 없으면 루프 종료
        break
    if s[0]=='UP':                  # s[0]은 명령을 나타냅니다.
        x-=int(s[1])                # s[1]은 이동 단위를 나타냅니다. (y
좌표 변경으로 수정해야 함)
    if s[0]=='DOWN':
        x+=int(s[1])                # (y 좌표 변경으로 수정해야 함)
    if s[0]=='LEFT':
        y-=int(s[1])                # (x 좌표 변경으로 수정해야 함)
    if s[0]=='RIGHT':
        y+=int(s[1])                # (x 좌표 변경으로 수정해야 함)
                                    # N**P는 N^P를 의미합니다.
# 원점에서 현재 위치 (y, x)까지의 유클리드 거리 계산 (일반적으로 (x,y)로 사용
되지만, 위 코드에서는 y가 x축, x가 y축 역할을 함)
# 문제의 의도에 맞게 수정: UP/DOWN은 y좌표, LEFT/RIGHT는 x좌표 변경
# x -> y_coord, y -> x_coord
# UP: y_coord += steps
# DOWN: y_coord -= steps
# LEFT: x_coord -= steps
# RIGHT: x_coord += steps

# 수정된 로직 (변수명은 그대로 사용하되, 좌표계의 의미를 바로잡음):
# x는 y축 이동량, y는 x축 이동량으로 간주

# 초기 위치 (0,0)
# UP 5: y_pos = 0 + 5 = 5.  (0, 5)
# DOWN 3: y_pos = 5 - 3 = 2. (0, 2)
# LEFT 3: x_pos = 0 - 3 = -3. (-3, 2)
# RIGHT 2: x_pos = -3 + 2 = -1. (-1, 2)
# 거리 = sqrt((-1)^2 + 2^2) = sqrt(1 + 4) = sqrt(5) approx 2.23
# 가장 가까운 정수는 2

# 위 코드의 변수 할당을 수정하여 표준 좌표계처럼 동작하게 하거나,
# 현재 변수 사용을 유지하되 주석을 명확히 하여 x가 수직, y가 수평 이동을 나타
냄을 밝힘.
# 여기서는 기존 코드의 로직을 최대한 따르되, 최종 거리 계산 시 변수의 의미를 명
확히 함.
# pos[0] -> y좌표, pos[1] -> x좌표 (주요 저자 솔루션 기준)
# 내 솔루션 기준: x -> y축 변화량 (부호 반대), y -> x축 변화량

# 내 솔루션의 좌표계 해석:
```

```
# (0,0)에서 시작
# UP 5 -> x = 0 - 5 = -5. 현재: (y=0, x=-5) -> (0, -5)로 해석 (y가 x
축, x가 y축 역할)
# DOWN 3 -> x = -5 + 3 = -2. 현재: (y=0, x=-2) -> (0, -2)
# LEFT 3 -> y = 0 - 3 = -3. 현재: (y=-3, x=-2) -> (-3, -2)
# RIGHT 2 -> y = -3 + 2 = -1. 현재: (y=-1, x=-2) -> (-1, -2)
# 거리 = sqrt((-1)^2 + (-2)^2) = sqrt(1+4) = sqrt(5) -> 2

# 만약 표준 좌표계를 따른다면:
# pos_x, pos_y = 0,0
# UP 5: pos_y += 5
# DOWN 3: pos_y -=3
# LEFT 3: pos_x -=3
# RIGHT 2: pos_x +=2
# 최종 위치: (-1, 2). 거리 = sqrt((-1)^2 + 2^2) = sqrt(5) -> 2

# 현재 내 솔루션의 변수 x, y는 최종적인 x,y 좌표가 아니라 누적된 이동량을 나타
내는 것으로 보임.
# UP 5 -> x = -5 (y축 방향으로 -5만큼 이동했다고 해석, 즉 아래로 5) - 이는
UP의 일반적인 의미와 반대.
# 코드를 문제 설명에 맞게 수정하는 것이 좋음.
# UP -> y 증가, DOWN -> y 감소, LEFT -> x 감소, RIGHT -> x 증가

# 수정된 내 솔루션 (표준 좌표계):
pos_x, pos_y = 0,0
while True:
    s = input().split()
    if not s:
        break
    direction = s[0]
    steps = int(s[1])
    if direction == 'UP':
        pos_y += steps
    elif direction == 'DOWN':
        pos_y -= steps
    elif direction == 'LEFT':
        pos_x -= steps
    elif direction == 'RIGHT':
        pos_x += steps

dist = round(math.sqrt(pos_x**2 + pos_y**2))
print(dist)
```

```
'''솔루션 작성자: pratikb0501
'''

from math import sqrt
lst = []
position = [0,0] # position[0]은 y좌표, position[1]은 x좌표로 사용
while True:
    a = input()
    if not a:
        break
    lst.append(a)
for i in lst:
    if 'UP' in i:
        position[0] += int(i.strip('UP ')) # UP이면 y좌표 증가
    if 'DOWN' in i:
        position[0] -= int(i.strip('DOWN ')) # DOWN이면 y좌표 감소
    if 'LEFT' in i:
        position[1] -= int(i.strip('LEFT ')) # LEFT면 x좌표 감소
    if 'RIGHT' in i:
        position[1] += int(i.strip('RIGHT ')) # RIGHT면 x좌표 증가
# 최종 위치는 (position[1], position[0])
print(round(sqrt(position[1] ** 2 + position[0] ** 2))) # 유클리드 거
리
```

이전 날로 가기

다음 날로 가기

토론

문제 22

문제:

입력으로부터 단어의 빈도를 계산하는 프로그램을 작성하십시오. 출력은 키를 영숫자순으로 정렬한 후 출력해야 합니다.

다음과 같은 입력이 프로그램에 제공된다고 가정합니다:

```
New to Python or choosing between Python 2 and Python 3? Read Python
2 or Python 3.
```

그러면 출력은 다음과 같아야 합니다:

```
2:2
3.:1
3?:1
New:1
Python:5
Read:1
and:1
between:1
choosing:1
or:2
to:1
```

힌트

> *질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.*

주요 저자 솔루션: 파이썬 2

```
freq = {}   # 텍스트 내 단어 빈도
line = raw_input()
for word in line.split():
    freq[word] = freq.get(word,0)+1

words = freq.keys()
words.sort()

for w in words:
    print "%s:%d" % (w,freq[w])
```

내 솔루션: 파이썬 3

```
ss = input().split()
word = sorted(set(ss))      # 분리된 단어는 집합으로 저장되고 정렬됩니다.

for i in word:
    print("{0}:{1}".format(i,ss.count(i))) # i: 단어, ss.count(i): 단
어의 빈도수
```

또는

```
ss = input().split()
dict = {}
for i in ss:
    i = dict.setdefault(i,ss.count(i))    # setdefault() 함수는 키와
값을 받아 사전으로 설정합니다.

dict = sorted(dict.items())               # items() 함수는 사전의 키와
값을 모두 리스트로 반환한 다음 정렬합니다.
                                          # 정렬은 기본적으로 첫 번째 ->
두 번째 키 순서로 수행됩니다.
for i in dict:
    print("%s:%d"%(i[0],i[1]))
```

또는

```
ss = input().split()
dict = {i:ss.count(i) for i in ss}     # i -> 분리된 단어,
ss.count(i) -> ss에서 i의 총 등장 횟수로 사전을 설정합니다.
dict = sorted(dict.items())            # items() 함수는 사전의 키와 값
을 모두 리스트로 반환한 다음 정렬합니다.
                                       # 정렬은 기본적으로 첫 번째 -> 두
번째 키 순서로 수행됩니다.
for i in dict:
    print("%s:%d"%(i[0],i[1]))
```

또는

```
from collections import Counter

ss = input().split()
ss = Counter(ss)          # 키와 빈도를 사전으로 반환합니다.
ss = sorted(ss.items())   # 튜플 리스트로 반환합니다.

for i in ss:
    print("%s:%d"%(i[0],i[1]))
```

솔루션 작성자: AnjanKumarG

```
from pprint import pprint # pprint는 "pretty print"의 약자로, 복잡한 자료구조를 사람이 읽기 쉽게 출력해줍니다.
p=input().split()
pprint({i:p.count(i) for i in p}) # 각 단어 i를 키로, p.count(i) (단어의 빈도)를 값으로 하는 딕셔너리를 생성하여 출력
```

문제 23

문제:

> ***숫자의 제곱 값을 계산할 수 있는 메소드를 작성하십시오.***

힌트:

`**` 연산자를 사용하며, `n**p`는 `n^p`를 의미합니다.

주요 저자 솔루션: 파이썬 2

```
def square(num):
    return num ** 2

print square(2)
print square(3)
```

내 솔루션: 파이썬 3

```
n=int(input())
print(n**2)
```

문제 24

문제:

파이썬에는 많은 내장 함수가 있으며, 사용 방법을 모르는 경우 온라인 문서를 읽거나 책을 찾을 수 있습니다. 하지만 파이썬에는 모든 내장 함수에 대한 내장 문서 기능이 있습니다.

abs(), int(), raw_input()과 같은 일부 파이썬 내장 함수 문서를 인쇄하는 프로그램을 작성하십시오.

그리고 사용자 정의 함수에 대한 문서도 추가하십시오.

힌트:

```
내장 문서 메소드는 __doc__입니다.
```

주요 저자 솔루션: 파이썬 2

```
print abs.__doc__
print int.__doc__
print raw_input.__doc__

def square(num):
    '''입력 숫자의 제곱 값을 반환합니다.

    입력 숫자는 정수여야 합니다.
    '''
    return num ** 2

print square(2)
print square.__doc__
```

내 솔루션: 파이썬 3

```
print(str.__doc__)
print(sorted.__doc__)

def pow(n,p):
    '''
    매개변수 n: 임의의 정수입니다.
    매개변수 p: n에 대한 거듭제곱입니다.
    반환 값: n의 p 거듭제곱 = n^p
    '''

    return n**p

print(pow(3,4))
print(pow.__doc__)
```

문제 25

문제:

> ***클래스 매개변수를 가지고 동일한 인스턴스 매개변수를 갖는 클래스를 정의하십시오.***

힌트:

인스턴스 매개변수를 정의하려면 __init__ 메소드에 추가해야 합니다. 생성자 매개변수로 객체를 초기화하거나 나중에 값을 설정할 수 있습니다.

주요 저자 솔루션: 파이썬 2

```
class Person:
    # 클래스 매개변수 "name" 정의
    name = "Person"

    def __init__(self, name = None):
        # self.name은 인스턴스 매개변수입니다.
        self.name = name

jeffrey = Person("Jeffrey")
print "%s name is %s" % (Person.name, jeffrey.name) # Person.name은
클래스 변수, jeffrey.name은 인스턴스 변수

nico = Person()
nico.name = "Nico" # 인스턴스 변수 설정
print "%s name is %s" % (Person.name, nico.name)
```

내 솔루션: 파이썬 3

```
class Car:
    name = "Car" # 클래스 변수

    def __init__(self,name = None): # name의 기본값은 None
        self.name = name # 인스턴스 변수

honda=Car("Honda") # 인스턴스 생성 시 인스턴스 변수 name을 "Honda"로 설정
print("%s name is %s"%(Car.name,honda.name)) # Car.name은 "Car",
honda.name은 "Honda"

toyota=Car() # 인스턴스 생성 시 name을 제공하지 않으면 self.name은 None이
됨
toyota.name="Toyota" # 인스턴스 변수 name을 "Toyota"로 설정
print("%s name is %s"%(Car.name,toyota.name)) # Car.name은 "Car",
toyota.name은 "Toyota"
```

이전 날로 가기

다음 날로 가기

토론

문제 26

문제:

> *두 숫자의 합을 계산할 수 있는 함수를 정의하십시오.*

힌트:

> *두 숫자를 인수로 사용하는 함수를 정의하십시오. 함수 내에서 합계를 계산하고 값을 반환할 수 있습니다.*

주요 저자 솔루션: 파이썬 2

```
def SumFunction(number1, number2):
    return number1 + number2

print SumFunction(1,2)
```

내 솔루션: 파이썬 3

```
sum = lambda n1,n2 : n1 + n2      # 여기서 람다는 합계와 같은 작은 함수를 정의하는 데 사용됩니다.
print(sum(1,2))
```

문제 27

문제:

> *정수를 문자열로 변환하여 콘솔에 인쇄할 수 있는 함수를 정의하십시오.*

힌트:

> *str()을 사용하여 숫자를 문자열로 변환하십시오.*

주요 저자 솔루션: 파이썬 2

```
def printValue(n):
    print str(n)

printValue(3)
```

내 솔루션: 파이썬 3

```
conv = lambda x : str(x) # 람다 함수 conv는 입력 x를 문자열로 변환합니다.
n = conv(10)
print(n)
print(type(n))            # 변수의 유형을 확인합니다.
```

문제 28

문제:

> *문자열 형태로 두 개의 정수를 받아 합계를 계산한 다음 콘솔에 인쇄할 수 있는 함수를 정의하십시오.*

힌트:

> *int()를 사용하여 문자열을 정수로 변환하십시오.*

주요 저자 솔루션: 파이썬 2

```
def printValue(s1,s2):
    print int(s1) + int(s2)
printValue("3","4") #7
```

내 솔루션: 파이썬 3

```
sum = lambda s1,s2 : int(s1) + int(s2) # 람다 함수 sum은 문자열 s1, s2
를 정수로 변환하여 더합니다.
print(sum("10","45"))      # 55
```

문제 29

문제:

> *두 개의 문자열을 입력으로 받아 연결한 다음 콘솔에 인쇄할 수 있는 함수를 정의하십시오.*

힌트:

> *+ 기호를 사용하여 문자열을 연결하십시오.*

주요 저자 솔루션: 파이썬 2

```
def printValue(s1,s2):
    print s1 + s2

printValue("3","4") #34
```

내 솔루션: 파이썬 3

```
sum = lambda s1,s2 : s1 + s2 # 람다 함수 sum은 문자열 s1과 s2를 연결합니다.
print(sum("10","45"))         # 1045
```

문제 30

문제:

> *두 개의 문자열을 입력으로 받아 콘솔에 최대 길이의 문자열을 인쇄할 수 있는 함수를 정의하십시오. 두 문자열의 길이가 같으면 함수는 모든 문자열을 한 줄씩 인쇄해야 합니다.*

힌트:

> *len() 함수를 사용하여 문자열의 길이를 가져옵니다.*

주요 저자 솔루션: 파이썬 2

```
def printValue(s1,s2):
    len1 = len(s1)
    len2 = len(s2)
    if len1 > len2:
        print s1
    elif len2 > len1:
        print s2
    else:
        print s1
        print s2

printValue("one","three")
```

내 솔루션: 파이썬 3

```
def printVal(s1,s2):
    len1 = len(s1)
    len2 = len(s2)
    if len1 > len2:
        print(s1)
    elif len1 < len2:
        print(s2)
    else:
        print(s1)
        print(s2)

s1,s2=input().split() # 사용자로부터 공백으로 구분된 두 문자열을 입력받습니다.
printVal(s1,s2)
```

```
'''솔루션 작성자: yuan1z'''
func = lambda a,b: print(max((a,b),key=len)) if len(a)!=len(b) else print(a+'\n'+b)
# 람다 함수 func:
# len(a)와 len(b)가 다르면, key=len을 사용하여 길이가 더 긴 문자열을 찾아 출력합니다.
# 길이가 같으면 a와 b를 각각 다른 줄에 출력합니다.
```

이전 날로 가기

다음 날로 가기

토론

문제 31

문제:

> ***키가 1과 20 사이의 숫자(둘 다 포함)이고 값이 키의 제곱인 사전을 인쇄할 수 있는 함수를 정의하십시오.***

힌트:

`dict[key]=value` 패턴을 사용하여 사전에 항목을 넣습니다. 숫자의 거듭제곱을 얻으려면 `**` 연산자를 사용합니다. 루프에는 `range()`를 사용합니다.

주요 저자 솔루션: 파이썬 2

```
def printDict():
    d=dict()
    for i in range(1,21):
        d[i]=i**2
    print d

printDict()
```

내 솔루션: 파이썬 3

```
def printDict():
    dict={i:i**2 for i in range(1,21)}    # 컴프리헨션 메소드 사용
    print(dict)

printDict()
```

문제 32

문제:

> *키가 1과 20 사이의 숫자(둘 다 포함)이고 값이 키의 제곱인 사전을 생성할 수 있는 함수를 정의하십시오. 함수는 키만 인쇄해야 합니다.*

힌트:

dict[key]=value 패턴을 사용하여 사전에 항목을 넣습니다. 숫자의 거듭제곱을 얻으려면 ** 연산자를 사용합니다. 루프에는 range()를 사용합니다. 사전의 키를 반복하려면 keys()를 사용합니다. 또한 item()을 사용하여 키/값 쌍을 가져올 수 있습니다.

주요 저자 솔루션: 파이썬 2

```
def printDict():
    d=dict()
    for i in range(1,21):
        d[i]=i**2
    for k in d.keys(): # d.keys()는 딕셔너리 d의 모든 키를 리스트로 반환합니다.
        print k
printDict()
```

내 솔루션: 파이썬 3

```
def printDict():
    dict = {i: i**2 for i in range(1, 21)}
    print(dict.keys())      # 사전의 키를 인쇄합니다.
                            # Python 3에서 dict.keys()는 dict_keys 객
체를 반환합니다. 리스트로 보려면 list(dict.keys())를 사용합니다.

printDict()
```

문제 33

문제:

> ***값이 1과 20 사이의 숫자(둘 다 포함)의 제곱인 리스트를 생성하고 인쇄할 수 있는 함수를 정의하십시오.***

힌트:

숫자의 거듭제곱을 얻으려면 ** 연산자를 사용합니다. 루프에는 range()를 사용합니다. 리스트에 값을 추가하려면 list.append()를 사용합니다.

주요 저자 솔루션: 파이썬 2

```
def printList():
    li=list()
    for i in range(1,21):
        li.append(i**2)
    print li

printList()
```

내 솔루션: 파이썬 3

```
def printList():
    lst = [i ** 2 for i in range(1, 21)] # 리스트 컴프리헨션을 사용하여 1부터 20까지 각 숫자 i의 제곱을 포함하는 리스트 lst를 생성합니다.
    print(lst)

printList()
```

문제 34

문제:

> ***값이 1과 20 사이의 숫자(둘 다 포함)의 제곱인 리스트를 생성할 수 있는 함수를 정의하십시오. 그런 다음 함수는 리스트의 처음 5개 요소를 인쇄해야 합니다.***

힌트:

숫자의 거듭제곱을 얻으려면 ** 연산자를 사용합니다. 루프에는 range()를 사용합니다. 리스트에 값을 추가하려면 list.append()를 사용합니다. 리스트를 슬라이스하려면 [n1:n2]를 사용합니다.

주요 저자 솔루션: 파이썬 2

```
def printList():
    li=list()
    for i in range(1,21):
        li.append(i**2)
    print li[:5] # 리스트 li의 처음부터 인덱스 5 이전까지의 요소 (즉, 처음 5개 요소)를 출력합니다.

printList()
```

내 솔루션: 파이썬 3

```
def printList():
    lst = [i ** 2 for i in range(1, 21)] # 1부터 20까지 각 숫자의 제곱을 포함하는 리스트를 생성합니다.

    for i in range(5): # 0부터 4까지 반복합니다.
        print(lst[i]) # 리스트 lst의 i번째 요소를 출력합니다. 즉, 처음 5개 요소를 각각 다른 줄에 출력합니다.

printList()
```

```
'''솔루션 작성자: popomaticbubble
'''
def squares(n):
    squares_list = [i**2 for i in range(1,n+1)] # 1부터 n까지 각 숫자의 제곱을 포함하는 리스트를 생성합니다.
    print(squares_list[0:5]) # 리스트의 처음 5개 요소를 출력합니다.
squares(20)
```

```
'''솔루션 작성자: yuan1z'''
func = lambda :print([i**2 for i in range(1,21)][:5]) # 람다 함수 func: 1부터 20까지 각 숫자의 제곱을 포함하는 리스트를 생성하고, 그 리스트의 처음 5개 요소를 출력합니다.
```

문제 35

문제:

> ***값이 1과 20 사이의 숫자(둘 다 포함)의 제곱인 리스트를 생성할 수 있는 함수를 정의하십시오. 그런 다음 함수는 리스트의 마지막 5개 요소를 인쇄해야 합니다.***

힌트:

숫자의 거듭제곱을 얻으려면 ** 연산자를 사용합니다. 루프에는 range()를 사용합니다. 리스트에 값을 추가하려면 list.append()를 사용합니다. 리스트를 슬라이스하려면 [n1:n2]를 사용합니다.

주요 저자 솔루션: 파이썬 2

```
def printList():
    li=list()
    for i in range(1,21):
        li.append(i**2)
    print li[-5:] # 리스트 li의 뒤에서 5번째 요소부터 끝까지의 요소 (즉, 마지막 5개 요소)를 출력합니다.

printList()
```

내 솔루션: 파이썬 3

```
def printList():
    lst = [i ** 2 for i in range(1, 21)] # 1부터 20까지 각 숫자의 제곱을 포함하는 리스트를 생성합니다.
    for i in range(19,14,-1): # 19부터 15까지 역순으로 반복합니다 (인덱스 19, 18, 17, 16, 15).
        print(lst[i]) # 리스트 lst의 i번째 요소를 출력합니다. 즉, 마지막 5개 요소를 역순으로 각각 다른 줄에 출력합니다.
                      # 만약 순서대로 출력하려면 print(lst[-5:]) 또는 print(lst[15:]) 와 같이 슬라이싱을 사용하거나,
                      # for i in range(15, 20): print(lst[i]) 와 같이 반복합니다.

printList()
```

```
'''솔루션 작성자: popomaticbubble
'''
def squares(n):
    squares_list = [i**2 for i in range(1,n+1)] # 1부터 n까지 각 숫자의 제곱을 포함하는 리스트를 생성합니다.
    print(squares_list[-5:]) # 리스트의 마지막 5개 요소를 출력합니다.
squares(20)
```

문제 36

문제:

> ***값이 1과 20 사이의 숫자(둘 다 포함)의 제곱인 리스트를 생성할 수 있는 함수를 정의하십시오. 그런 다음 함수는 리스트에서 처음 5개 요소를 제외한 모든 값을 인쇄해야 합니다.***

힌트: 숫자의 거듭제곱을 얻으려면 ** 연산자를 사용합니다. 루프에는 range()를 사용합니다. 리스트에 값을 추가하려면 list.append()를 사용합니다. 리스트를 슬라이스하려면 [n1:n2]를 사용합니다.

주요 저자 솔루션: 파이썬 2

```
def printList():
    li=list()
    for i in range(1,21):
        li.append(i**2)
    print li[5:] # 리스트 li의 인덱스 5부터 끝까지의 요소 (즉, 처음 5개 요소를 제외한 나머지)를 출력합니다.

printList()
```

내 솔루션: 파이썬 3

```
def printList():
    lst = [i ** 2 for i in range(1, 21)] # 1부터 20까지 각 숫자의 제곱을 포함하는 리스트를 생성합니다.
    for i in range(5,20): # 5부터 19까지 반복합니다.
        print(lst[i]) # 리스트 lst의 i번째 요소를 출력합니다. 즉, 인덱스 5부터 19까지의 요소를 각각 다른 줄에 출력합니다.

printList()
```

문제 37

문제:

> ***값이 1과 20 사이의 숫자(둘 다 포함)의 제곱인 튜플을 생성하고 인쇄할 수 있는 함수를 정의하십시오.***

힌트:

숫자의 거듭제곱을 얻으려면 ** 연산자를 사용합니다. 루프에는 range()를 사용합니다. 리스트에 값을 추가하려면 list.append()를 사용합니다. 리스트에서 튜플을 얻으려면 tuple()을 사용합니다.

주요 저자 솔루션: 파이썬 2

```
def printTuple():
    li=list()
    for i in range(1,21):
        li.append(i**2)
    print tuple(li) # 리스트 li를 튜플로 변환하여 출력합니다.

printTuple()
```

내 솔루션: 파이썬 3

```
def printTupple(): # 함수 이름 오타 수정: printTuple
    lst = [i ** 2 for i in range(1, 21)] # 1부터 20까지 각 숫자의 제곱을 포함하는 리스트를 생성합니다.
    print(tuple(lst)) # 리스트 lst를 튜플로 변환하여 출력합니다.

printTupple() # 함수 이름 오타 수정: printTuple()
```

```
'''
솔루션 작성자: Seawolf159
'''
def square_of_numbers():
    return tuple(i ** 2 for i in range(1, 21)) # 제너레이터 표현식을 사용하여 1부터 20까지 각 숫자 i의 제곱을 포함하는 튜플을 직접 생성하여 반환합니다.

print(square_of_numbers())
```

코멘트

이 섹션의 문제들은 매우 쉽고 모두 동일한 유형의 문제를 약간 변형한 것입니다. 주로 리스트, 사전, 튜플과 함께 사용되는 일반적으로 사용되는 일부 함수 작업을 사용하는 데 중점을 두었습니다. 전체 솔루션에서 효율적인 방식으로 문제를 해결하려고 시도하지 않았습니다. 오히려 제가 할 수 있는 다른 방식으로 해결하려고 노력했습니다. 이것은 초보자가 가장 간단한 문제를 다양한 방식으로 해결할 수 있는 방법을 아는 데 도움이 될 것입니다.

이전 날로 가기

다음 날로 가기

토론

문제 38

문제:

> *주어진 튜플 (1,2,3,4,5,6,7,8,9,10)에 대해, 처음 절반의 값은 한 줄에, 마지막 절반의 값은 다른 한 줄에 인쇄하는 프로그램을 작성하십시오.*

힌트:

> *튜플에서 슬라이스를 가져오려면 [n1:n2] 표기법을 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
tp = (1,2,3,4,5,6,7,8,9,10)
tp1 = tp[:5] # 튜플 tp의 처음부터 인덱스 5 이전까지 (즉, 처음 5개 요소)
tp2 = tp[5:] # 튜플 tp의 인덱스 5부터 끝까지 (즉, 나머지 5개 요소)
print tp1
print tp2
```

내 솔루션: 파이썬 3

```
tpl = (1,2,3,4,5,6,7,8,9,10)

for i in range(0,5): # 0부터 4까지 반복
    print(tpl[i],end = ' ') # tpl[i]를 출력하고 공백으로 끝냅니다 (줄 바
꿈 안 함).
print() # 줄 바꿈
for i in range(5,10): # 5부터 9까지 반복
    print(tpl[i],end = ' ') # tpl[i]를 출력하고 공백으로 끝냅니다.
```

또는

```
tpl = (1,2,3,4,5,6,7,8,9,10)
lst1,lst2 = [],[] # 빈 리스트 두 개 생성

for i in range(0,5):
    lst1.append(tpl[i]) # 처음 5개 요소를 lst1에 추가

for i in range(5,10):
    lst2.append(tpl[i]) # 나머지 5개 요소를 lst2에 추가

print(lst1)
print(lst2)
```

```
'''
솔루션 작성자: CoffeeBrakeInc
'''

tup = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
lt = int(len(tup)/2) # 튜플 길이의 절반을 정수로 계산
print(tup[:lt], tup[lt:]) # 처음 절반과 나머지 절반을 각각 출력
```

```
'''
솔루션 작성자: AasaiAlangaram
'''

tp = (1,2,3,4,5,6,7,8,9,10)

print('원본 튜플:',tp)

# 리스트 컴프리헨션을 사용하여 5개 요소씩 분할하여 출력
[print('분할된 리스트 :{List}'.format(List = tp[x:x+5])) for x in
range(0,len(tp),5)]
# range(0, len(tp), 5)는 0, 5를 생성합니다.
# x가 0일 때 tp[0:5] 출력
# x가 5일 때 tp[5:10] 출력
```

```
'''
솔루션 작성자: saxenaharsh24
'''

tup = [i for i in range(1, 11)] # 1부터 10까지 숫자를 포함하는 리스트 생성
print(f'{tuple(tup[:5])} \n{tuple(tup[5:])}') # 리스트의 처음 5개 요소를 튜플로, 나머지 5개 요소를 튜플로 변환하여 각각 다른 줄에 출력
```

문제 39

문제:

> *주어진 튜플 (1,2,3,4,5,6,7,8,9,10)에서 짝수 값만 포함하는 다른 튜플을 생성하고 인쇄하는 프로그램을 작성하십시오.*

힌트:

> *튜플을 반복하려면 "for"를 사용하십시오. 리스트에서 튜플을 생성하려면 tuple()을 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
tp = (1,2,3,4,5,6,7,8,9,10)
li = list()
for i in tp: # tp의 각 요소 i에 대해 반복
    if tp[i]%2 == 0: # 이 부분은 tp[i] 대신 i를 사용해야 합니다. tp의 인덱스가 아니라 요소 자체를 확인해야 합니다.
                     # 예를 들어 i가 1이면 tp[1]은 2입니다. i가 2이면 tp[2]는 3입니다.
                     # 올바른 조건: if i % 2 == 0:
        li.append(tp[i]) # 여기도 tp[i] 대신 i를 사용해야 합니다.

tp2 = tuple(li)
print tp2
```

참고: 위 주 저자 솔루션에는 `tp[i]` 를 사용하는 부분에 오류가 있습니다. `i` 자체가 튜플의 요소이므로 `i % 2 == 0` 및 `li.append(i)` 로 수정해야 합니다.

내 솔루션: 파이썬 3

```
tpl = (1,2,3,4,5,6,7,8,9,10)
tpl1 = tuple(i for i in tpl if i%2 == 0) # 튜플 tpl의 각 요소 i에 대해 i가 짝수이면 해당 i를 포함하는 제너레이터를 튜플로 변환합니다.
print(tpl1)
```

또는

```
tpl = (1,2,3,4,5,6,7,8,9,10)
tpl1 = tuple(filter(lambda x : x%2==0,tpl))  # 람다 함수는 짝수 요소를 찾으면 True를 반환합니다.
                                             # filter는 함수가 False를 반환하는 데이터를 제거합니다.
print(tpl1)
```

문제 40

문제:

> *문자열을 입력으로 받아 문자열이 "yes" 또는 "YES" 또는 "Yes"이면 "Yes"를 인쇄하고 그렇지 않으면 "No"를 인쇄하는 프로그램을 작성하십시오.*

힌트:

> *조건을 판단하려면 if 문을 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
s= raw_input()
if s=="yes" or s=="YES" or s=="Yes":
    print "Yes"
else:
    print "No"
```

솔루션: 파이썬 3

```
'''
솔루션 작성자: Seawolf159
'''
text = input("무언가를 입력하세요. --> ")
if text == "yes" or text == "YES" or text == "Yes":
    print("Yes")
else:
    print("No")
```

```
'''
솔루션 작성자: AasaiAlangaram
'''
input_str = input('문자열 입력:') # 변수명을 input에서 input_str로 변경 (input은 내장 함수 이름과 충돌)
output = ''.join(['Yes' if input_str == 'yes' or input_str =='YES' or input_str =='Yes' else 'No' ]) # 조건부 표현식을 사용하여 'Yes' 또는 'No'를 리스트에 넣고 join으로 문자열화
print(str(output))
```

```
'''
솔루션 작성자: Prashanth
'''
x = str(input().lower()) # 입력을 받아 소문자로 변환
if x == 'yes': # 소문자로 변환된 입력이 'yes'인지 비교
    print('Yes')
else:
    print('No')
```

문제 41

문제:

> ***map()을 사용하여 [1,2,3,4,5,6,7,8,9,10]의 요소 제곱 값으로 구성된 리스트를 만드는 프로그램을 작성하십시오.***

힌트:

> ***map()을 사용하여 리스트를 생성하십시오. 익명 함수를 정의하려면 lambda를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
li = [1,2,3,4,5,6,7,8,9,10]
squaredNumbers = map(lambda x: x**2, li) # li의 각 요소 x에 대해 x**2
를 계산하여 map 객체 반환 (Python 2에서는 리스트 반환)
print squaredNumbers
```

내 솔루션: 파이썬 3

```
# 문제 설명에서 요구 사항이 구체적으로 언급되었으므로 다른 방식의 코드는 작성하
지 않았습니다.

li = [1,2,3,4,5,6,7,8,9,10]
squaredNumbers = map(lambda x: x**2, li)  # map 유형 객체 데이터를 반환
합니다.
print(list(squaredNumbers))               # 객체를 리스트로 변환합니다.
```

```
'''
솔루션 작성자: saxenaharsh24
'''
def sqrs(item): # 제곱을 계산하는 함수 정의
    return item ** 2


lst = [i for i in range(1, 11)] # 1부터 10까지의 리스트 생성
print(list(map(sqrs, lst))) # map을 사용하여 lst의 각 요소에 sqrs 함수
적용
```

문제 42

문제:

---

map()과 filter()를 사용하여 [1,2,3,4,5,6,7,8,9,10]에서 짝수의 제곱 값으로 구

성된 리스트를 만드는 프로그램을 작성하십시오.

힌트:

map()을 사용하여 리스트를 생성하십시오. filter()를 사용하여 리스트의 요소를 필터링하십시오. 익명 함수를 정의하려면 lambda를 사용하십시오.

주요 저자 솔루션: 파이썬 2

```
li = [1,2,3,4,5,6,7,8,9,10]
evenNumbers = map(lambda x: x**2, filter(lambda x: x%2==0, li)) # 먼저 filter로 짝수만 거르고, 그 결과에 map으로 제곱 적용
print evenNumbers
```

내 솔루션: 파이썬 3

```
def even(x): # 짝수인지 확인하는 함수
    return x%2==0

def squer(x): # 제곱하는 함수 (오타: squer -> square)
    return x*x

li = [1,2,3,4,5,6,7,8,9,10]
li = map(squer,filter(even,li))    # 먼저 even 함수로 숫자를 필터링한 다음 결과 요소에 map()을 적용합니다.
print(list(li))
```

```
"""
솔루션 작성자: saxenaharsh24
"""
def even(item): # 짝수이면 제곱을 반환하고, 홀수이면 아무것도 반환하지 않는 (None을 반환하는) 함수
    if item % 2 == 0:
        return item**2


lst = [i for i in range(1, 11)]
# map(even, lst)의 결과는 [None, 4, None, 16, None, 36, None, 64, None, 100]과 유사합니다.
# filter(lambda j: j is not None, ...)는 None이 아닌 요소만 필터링합니다.
print(list(filter(lambda j: j is not None, list(map(even, lst)))))
```

문제 43

문제:

> ***filter()를 사용하여 1과 20 사이(둘 다 포함)의 짝수로 구성된 리스트를 만드는 프로그램을 작성하십시오.***

힌트:

> ***filter()를 사용하여 리스트의 요소를 필터링하십시오. 익명 함수를 정의하려면 lambda를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
evenNumbers = filter(lambda x: x%2==0, range(1,21)) # range(1,21)의
각 요소 x에 대해 x가 짝수인지 확인하여 필터링
print evenNumbers
```

내 솔루션: 파이썬 3

```
def even(x): # 짝수인지 확인하는 함수
    return x%2==0

evenNumbers = filter(even, range(1,21)) # range(1,21)의 각 요소에 대해
even 함수를 적용하여 필터링
print(list(evenNumbers)) # filter 객체를 리스트로 변환하여 출력
```

이전 날로 가기

다음 날로 가기

토론

문제 44

문제:

> ***map()을 사용하여 1과 20 사이(둘 다 포함) 숫자의 제곱으로 구성된 리스트를 만드는 프로그램을 작성하십시오.***

힌트:

> ***map()을 사용하여 리스트를 생성하십시오. 익명 함수를 정의하려면 lambda를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
squaredNumbers = map(lambda x: x**2, range(1,21)) # range(1,21)의 각 x에 대해 x의 제곱을 계산합니다. (Python 2에서는 map이 리스트를 반환)
print squaredNumbers
```

내 솔루션: 파이썬 3

```
def sqr(x): # x의 제곱을 반환하는 함수
    return x*x

squaredNumbers = list(map(sqr, range(1,21))) # range(1,21)의 각 숫자에 sqr 함수를 적용하고 결과를 리스트로 변환합니다.
print (squaredNumbers)
```

문제 45

문제:

> ***printNationality라는 정적 메소드를 가진 American이라는 클래스를 정의하십시오.***

힌트:

> ***클래스 정적 메소드를 정의하려면 @staticmethod 데코레이터를 사용하십시오. 두 가지 다른 메소드도 있습니다. 자세한 내용은 이 링크를 참조하십시오.***

주요 저자 솔루션: 파이썬 2

```
class American(object): # object를 상속 (Python 2에서의 new-style 클래스)
    @staticmethod
    def printNationality():
        print "America"

anAmerican = American() # American 클래스의 인스턴스 생성
anAmerican.printNationality() # 인스턴스를 통해 정적 메소드 호출
American.printNationality() # 클래스 자체를 통해 정적 메소드 호출
```

내 솔루션: 파이썬 3

```python
class American():
    @staticmethod
    def printNationality():
        print("I am American")

american = American()
american.printNationality()    # @staticmethod가 함수를 데코레이트하지 않으면 이 코드는 실행되지 않습니다.
                               # 클래스에 인스턴스가 없기 때문입니다. (-> 이 주석은 약간 오해의 소지가 있습니다. @staticmethod가 없으면 인스턴스 메소드가 되어 self 인자가 필요하게 됩니다. 인스턴스가 없는 것이 아니라, self 없이 호출하려고 해서 문제가 됩니다.)
                               # 정확히는, @staticmethod가 없으면 printNationality는 인스턴스 메소드가 되고, american.printNationality() 호출 시 self 인자가 암묵적으로 전달됩니다.
                               # 하지만 printNationality() 정의에는 self가 없으므로 TypeError가 발생합니다.

American.printNationality()    # 이 코드는 @staticmethod가
                               # printNationality()를 데코레이트하지 않아도 실행됩니다. (-> 이 주석도 오해의 소지가 있습니다. @staticmethod가 없으면 American.printNationality() 호출 시 self 인자가 누락되어 TypeError가 발생합니다.)
                               # 정확히는, @staticmethod가 없으면 클래스를 통해 직접 호출 시에도 self 인자를 명시적으로 제공해야 합니다 (American.printNationality(american) 처럼).
                               # @staticmethod를 사용하면 self 없이 클래스나 인스턴스를 통해 호출 가능해집니다.
```

참고: `@staticmethod` 데코레이터는 메소드가 클래스 인스턴스(`self`)나 클래스(`cls`) 자체에 접근할 필요가 없을 때 사용됩니다. 이 데코레이터가 있으면 인스턴스나 클래스를 통해 호출할 때 특별한 첫 번째 인자(self 또는 cls)가 전달되지 않습니다.

문제 46

문제:

> ***American이라는 클래스와 그 하위 클래스 NewYorker를 정의하십시오.***

힌트:

> **하위 클래스를 정의하려면 `class Subclass(ParentClass):` 를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
class American(object): # object 상속 (Python 2)
    pass # 아무것도 하지 않는 클래스 정의

class NewYorker(American): # American 클래스를 상속하는 NewYorker 클래스 정의
    pass

anAmerican = American()
aNewYorker = NewYorker()
print anAmerican # 객체의 문자열 표현 출력 (예: <__main__.American object at 0x...>)
print aNewYorker # 객체의 문자열 표현 출력 (예: <__main__.NewYorker object at 0x...>)
```

내 솔루션: 파이썬 3

```
class American(): # Python 3에서는 기본적으로 object를 상속합니다.
    pass

class NewYorker(American): # American 클래스를 상속합니다.
    pass

american = American()
newyorker = NewYorker()

print(american) # 객체의 문자열 표현 출력
print(newyorker) # 객체의 문자열 표현 출력
```

이전 날로 가기

다음 날로 가기

토론

문제 47

문제

> *반지름으로 생성될 수 있는 Circle이라는 클래스를 정의하십시오. Circle 클래스에는 넓이를 계산할 수 있는 메소드가 있습니다.*

힌트

> *메소드를 정의하려면 def methodName(self)를 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
class Circle(object):
    def __init__(self, r): # 생성자, 반지름 r을 인스턴스 변수
self.radius에 저장
        self.radius = r

    def area(self): # 넓이를 계산하는 메소드
        return self.radius**2*3.14 # 반지름의 제곱 * 원주율 (3.14)

aCircle = Circle(2) # 반지름이 2인 Circle 객체 생성
print aCircle.area() # 넓이 출력
```

내 솔루션: 파이썬 3

```python
class Circle():
    def __init__(self,r):
        self.radius = r

    def area(self):
        return 3.1416*(self.radius**2) # 원주율로 3.1416 사용


circle = Circle(5)
print(circle.area())
```

문제 48

문제

> *길이와 너비로 생성될 수 있는 **Rectangle**이라는 클래스를 정의하십시오. **Rectangle** 클래스에는 넓이를 계산할 수 있는 메소드가 있습니다.*

힌트

> *메소드를 정의하려면 **def methodName(self)**를 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
class Rectangle(object):
    def __init__(self, l, w): # 생성자, 길이 l과 너비 w를 인스턴스 변수에
저장
        self.length = l
        self.width  = w

    def area(self): # 넓이를 계산하는 메소드
        return self.length*self.width # 길이 * 너비

aRectangle = Rectangle(2,10) # 길이 2, 너비 10인 Rectangle 객체 생성
print aRectangle.area() # 넓이 출력
```

내 솔루션: 파이썬 3

```
class Rectangle():
    def __init__(self,l,w):
        self.length = l
        self.width = w

    def area(self):
        return self.length*self.width


rect = Rectangle(2,4)
print(rect.area())
```

문제 49

문제

> ***Shape이라는 클래스와 그 하위 클래스 Square를 정의하십시오. Square 클래스에는 길이를 인수로 받는 init 함수가 있습니다. 두 클래스 모두 도형의 넓이를 인쇄할 수 있는 area 함수를 가지고 있으며, Shape의 넓이는 기본적으로 0입니다.***

힌트

> ***슈퍼 클래스의 메소드를 재정의하려면 슈퍼 클래스와 동일한 이름의 메소드를 정의할 수 있습니다.*** (주: 하위 클래스에서 동일한 이름의 메소드를 정의하여 재정의합니다.)

주요 저자 솔루션: 파이썬 2

```
class Shape(object):
    def __init__(self):
        pass

    def area(self): # Shape의 기본 넓이는 0
        return 0

class Square(Shape): # Shape 클래스를 상속
    def __init__(self, l):
        Shape.__init__(self) # 부모 클래스 Shape의 생성자 호출
        self.length = l # 길이 l을 인스턴스 변수에 저장

    def area(self): # Shape 클래스의 area 메소드를 재정의
        return self.length*self.length # 정사각형의 넓이: 한 변의 길이 *
한 변의 길이

aSquare= Square(3) # 한 변의 길이가 3인 Square 객체 생성
print aSquare.area() # 넓이 출력 (9)
```

내 솔루션: 파이썬 3

```
class Shape():
    def __init__(self):
        pass

    def area(self):
        return 0

class Square(Shape):
    def __init__(self,length = 0): # length의 기본값을 0으로 설정
        Shape.__init__(self) # Python 3에서는 super().__init__()도 사
용 가능
        self.length = length

    def area(self):
        return self.length*self.length

Asqr = Square(5) # 한 변의 길이가 5인 Square 객체 생성
print(Asqr.area())      # 주어진 인수로 25를 인쇄합니다.

print(Square().area())  # 기본 넓이로 0을 인쇄합니다 (length가 0으로 초기
화됨).
```

문제 50

문제

RuntimeError 예외를 발생시키십시오.

힌트

예외를 발생시키려면 raise()를 사용하십시오.

솔루션:

```
raise RuntimeError('문제가 발생했습니다') # 'something wrong'을 '문제가 발생했습니다'로 번역
```

결론

음, 위의 문제들은 객체 지향 프로그래밍의 기본 개념과 구현에 매우 중점을 둔 것 같습니다. 개념이 기능적 문제를 해결하는 것이 아니라 구조를 설계하는 것이므로 두 코드 모두 구현 부분에서 매우 유사합니다.

이전 날로 가기

다음 날로 가기

토론

문제 51

문제

> ***5/0을 계산하는 함수를 작성하고 try/except를 사용하여 예외를 처리하십시오.***

힌트

> ***try/except를 사용하여 예외를 처리하십시오.***

주요 저자 솔루션: 파이썬 2

```
def throws():
    return 5/0

try:
    throws()
except ZeroDivisionError: # 0으로 나누려고 할 때 발생하는 예외
    print "0으로 나누었습니다!"
except Exception, err: # 다른 모든 예외 처리 (Python 2 구문)
    print '예외가 발생했습니다'
finally: # 예외 발생 여부와 관계없이 항상 실행
    print '정리를 위한 finally 블록입니다'
```

내 솔루션: 파이썬 3

```
def divide():
    return 5/0

try:
    divide()
except ZeroDivisionError as ze: # ZeroDivisionError 예외를 잡아 변수
ze에 할당
    print("왜 숫자를 0으로 나누려고 하십니까!!")
except: # 다른 모든 예외 처리
    print("기타 예외")
```

문제 52

문제

> *문자열 메시지를 속성으로 사용하는 사용자 정의 예외 클래스를 정의하십시오.*

힌트

> *사용자 정의 예외를 정의하려면 **Exception**에서 상속받은 클래스를 정의해야 합니다.*

주요 저자 솔루션: 파이썬 2

```
class MyError(Exception): # Exception 클래스를 상속
    """나만의 예외 클래스

    속성:
        msg  -- 오류 설명
    """

    def __init__(self, msg):
        self.msg = msg # 오류 메시지를 인스턴스 변수 msg에 저장

error = MyError("문제가 발생했습니다") # MyError 예외 객체 생성
```

내 솔루션: 파이썬 3

```
class CustomException(Exception): # Exception 클래스를 상속
    """사용자 정의 목적을 위해 발생하는 예외

    속성:
        message -- 오류 설명
    """

    def __init__(self, message):
        self.message = message # 오류 메시지를 인스턴스 변수 message에 저장


num = int(input()) # 사용자로부터 정수 입력

try:
    if num < 10:
        raise CustomException("입력값이 10보다 작습니다") # num이 10보다 작으면 CustomException 발생
    elif num > 10:
        raise CustomException("입력값이 10보다 큽니다") # num이 10보다 크면 CustomException 발생
except CustomException as ce: # CustomException 예외를 잡아 변수 ce에 할당
    print("발생한 오류: " + ce.message) # 예외 메시지 출력
```

문제 53

문제

> ***"username@companyname.com" 형식의 이메일 주소가 있다고 가정하고, 주어진 이메일 주소의 사용자 이름을 인쇄하는 프로그램을 작성하십시오. 사용자 이름과 회사 이름은 모두 문자로만 구성됩니다.***

> ***예시: 다음 이메일 주소가 프로그램에 입력으로 주어지면:***

```
john@google.com
```

> ***그러면 프로그램의 출력은 다음과 같아야 합니다:***

```
john
```

> ***질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.***

힌트

> ***\w를 사용하여 문자와 일치시킵니다.*** (정규 표현식에서 \w는 영숫자 문자 + 밑줄을 의미합니다.)

주요 저자 솔루션: 파이썬 2

```
import re # 정규 표현식 모듈 가져오기
emailAddress = raw_input() # 사용자로부터 이메일 주소 입력
pat2 = "(\w+)@((\w+\.)+(com))" # 정규 표현식 패턴: (\w+)는 사용자 이름, ((\w+\.)+(com))은 회사 도메인 부분
r2 = re.match(pat2,emailAddress) # 패턴이 문자열 시작 부분과 일치하는지 확인
print r2.group(1) # 첫 번째 괄호 그룹 (사용자 이름) 출력
```

내 솔루션: 파이썬 3

```
email = "john@google.com"
email = email.split('@') # '@' 문자를 기준으로 문자열을 분리하여 리스트 생성
print(email[0]) # 리스트의 첫 번째 요소 (사용자 이름) 출력
```

또는

```
import re

email = "john@google.com elise@python.com" # 여러 이메일 주소가 있는 문자열
pattern = "(\w+)@\w+\.com" # 정규 표현식 패턴: (\w+)는 사용자 이름, \w+\.com은 @ 뒤의 회사 이름과 .com (닷을 이스케이프 처리)
ans = re.findall(pattern,email) # 패턴과 일치하는 모든 부분을 찾아 리스트로 반환 (사용자 이름만)
print(ans) # 결과 리스트 출력 (예: ['john', 'elise'])
```

이전 날로 가기

다음 날로 가기

토론

문제 54

문제

> *"username@companyname.com" 형식의 이메일 주소가 있다고 가정하고, 주어진 이메일 주소의 회사 이름을 인쇄하는 프로그램을 작성하십시오. 사용자 이름과 회사 이름은 모두 문자로만 구성됩니다.*

> *예시: 다음 이메일 주소가 프로그램에 입력으로 주어지면:*

```
john@google.com
```

> *그러면 프로그램의 출력은 다음과 같아야 합니다:*

```
google
```

> *질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.*

힌트

> *\w를 사용하여 문자와 일치시킵니다.*

주요 저자 솔루션: 파이썬 2

```
import re
emailAddress = raw_input()
pat2 = "(\w+)@(\w+)\.(com)" # 패턴: 사용자이름@회사이름.com. 그룹 2가 회사이름 (\w+)
r2 = re.match(pat2,emailAddress)
print r2.group(2) # 두 번째 그룹 (회사 이름) 출력
```

내 솔루션: 파이썬 3

```
import re

email = "john@google.com elise@python.com"
pattern = "\w+@(\w+)\.com" # 패턴: 사용자이름@회사이름.com. 회사이름을 그룹으로 지정. \.은 실제 점(.) 문자와 일치
ans = re.findall(pattern,email) # 패턴에 맞는 모든 회사 이름을 찾아 리스트로 반환
print(ans) # 예: ['google', 'python']
```

문제 55

문제

> ***공백으로 구분된 단어 시퀀스를 입력으로 받아 숫자만으로 구성된 단어를 인쇄하는 프로그램을 작성하십시오.***

> ***예시: 다음 단어가 프로그램에 입력으로 주어지면:***

```
2 cats and 3 dogs.
```

> ***그러면 프로그램의 출력은 다음과 같아야 합니다:***

```
['2', '3']
```

> *질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.*

힌트

> *정규식을 사용하여 모든 하위 문자열을 찾으려면 **re.findall()**을 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
import re
s = raw_input()
print re.findall("\d+",s) # \d+는 하나 이상의 숫자로 이루어진 시퀀스와 일치
```

내 솔루션: 파이썬 3

```
import re

text_input = input() # 변수명을 email에서 text_input으로 변경하여 명확성 증대
pattern = "\d+"
ans = re.findall(pattern,text_input)
print(ans)
```

또는

```
text_input = input().split() # 공백으로 단어 분리
ans = []
for word in text_input:
    if word.isdigit():      # isdigit()는 문자열이 모두 숫자로만 이루어져 있고 하나 이상의 문자가 있을 때 True 반환
        ans.append(word)    # isnumeric() / isdecimal() 함수도 유사하게 사용 가능
print(ans)
```

또는

```
text_input = input().split()
ans = [word for word in text_input if word.isdigit()]  # 리스트 컴프리헨션 사용
print(ans)
```

문제 56

문제

> ***유니코드 문자열 "hello world"를 인쇄하십시오.***

힌트

> ***유니코드 문자열을 정의하려면 u'문자열' 형식을 사용하십시오.*** (주: 파이썬 3에서는 모든 문자열이 기본적으로 유니코드입니다.)

주요 저자 솔루션: 파이썬 2

```
unicodeString = u"hello world!" # 파이썬 2에서 유니코드 문자열 명시
print unicodeString
```

파이썬 3 참고: 파이썬 3에서는 문자열 리터럴 앞에 `u` 를 붙일 필요가 없습니다. `my_string = "hello world!"` 와 같이 작성하면 `my_string` 은 이미 유니코드 문자열입니다.

문제 57

문제

> ***ASCII 문자열을 읽어 utf-8로 인코딩된 유니코드 문자열로 변환하는 프로그램을 작성하십시오.***

힌트

> ***변환하려면 unicode()/encode() 함수를 사용하십시오.*** (주: 파이썬 2에서는 `unicode()` , 파이썬 3에서는 문자열의 `.encode()` 메소드)

주요 저자 솔루션: 파이썬 2

```
s = raw_input() # ASCII 문자열 입력
u = unicode( s ,"utf-8") # 입력 문자열 s를 "utf-8" 인코딩을 사용하여 유니코드 문자열로 디코딩
print u
```

내 솔루션: 파이썬 3

```
s = input() # 파이썬 3에서 input()은 유니코드 문자열을 반환합니다.
# 만약 s가 이미 유니코드 문자열이고 이를 UTF-8 바이트 시퀀스로 "인코딩"하려면
.encode()를 사용합니다.
# 문제의 의도가 "ASCII 문자열을 읽어" -> "UTF-8 바이트로 변환"이라면 아래가
맞습니다.
u = s.encode('utf-8') # 문자열 s를 UTF-8 바이트 시퀀스로 인코딩
print(u) # 바이트 시퀀스 출력 (예: b'hello')

# 만약 문제의 의도가 "바이트 시퀀스(ASCII로 가정)를 읽어" -> "유니코드 문자열
로 변환(디코딩)"이라면,
# 예를 들어 바이트 문자열을 입력받는 상황을 가정해야 합니다.
# s_bytes = b"some ascii string"
# u_string = s_bytes.decode('ascii').encode('utf-8') # ASCII 바이트 -
> 유니코드 문자열 -> UTF-8 바이트
# print(u_string)
# 하지만 일반적인 input()은 이미 유니코드 문자열을 다룹니다.
```

문제 58

문제

> ***파이썬 소스 코드 파일이 유니코드임을 나타내는 특수 주석을 작성하십시오.***

힌트

> ***변환하려면 unicode() 함수를 사용하십시오.*** (주: 이 힌트는 문제와 직접 관련이 없어 보입니다. 인코딩 선언 주석에 대한 문제입니다.)

솔루션:

```
# -*- coding: utf-8 -*-
```

문제 59

문제

> ***콘솔에서 주어진 n(n>0)에 대해 1/2+2/3+3/4+...+n/(n+1)을 계산하는 프로그램을 작성하십시오.***

> ***예시: 다음 n이 프로그램에 입력으로 주어지면:***

```
5
```

> ***그러면 프로그램의 출력은 다음과 같아야 합니다:***

```
3.55
```

> ***질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.***

힌트

> ***정수를 부동 소수점으로 변환하려면 float()를 사용하십시오. 변환하지 않아도 파이썬은 기본적으로 값의 데이터 유형을 이해하므로 문제가 발생하지 않습니다.*** (주: 파이썬 2에서는 정수 나누기 정수는 정수 결과를 반환하므로 float() 변

환이 중요합니다. 파이썬 3에서는 정수 나누기 정수가 부동 소수점 결과를 반환합니다.)

주요 저자 솔루션: 파이썬 2

```
n=int(raw_input())
sum=0.0
for i in range(1,n+1):
    sum += float(float(i)/(i+1)) # i 또는 (i+1) 중 하나 이상을 float으로 변환하여 부동 소수점 나누기 수행
print sum
```

내 솔루션: 파이썬 3

```
n = int(input())
sum_val = 0 # 변수명을 sum에서 sum_val로 변경 (sum은 내장 함수)
for i in range(1, n+1):
    sum_val += i/(i+1) # 파이썬 3에서는 i/(i+1)이 자동으로 부동 소수점 나누기 수행
print(round(sum_val, 2))  # 소수점 이하 2자리로 반올림
```

```
'''솔루션 작성자: lcastrooliveira
'''
def question_59(n):
    # map과 람다를 사용하여 각 i에 대해 i/(i+1) 계산 후 합계, 결과를 소수점 2자리로 반올림
    print(round(sum(map(lambda x: x/(x+1), range(1, n+1))), 2))

question_59(5)
```

이전 날로 가기

다음 날로 가기

토론

문제 60

문제

다음을 계산하는 프로그램을 작성하십시오:

```
f(n)=f(n-1)+100 (n>0 일 때)
f(0)=0
```

콘솔에서 주어진 n(n>0)을 사용합니다.

예시: 다음 n이 프로그램에 입력으로 주어지면:

```
5
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
500
```

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

힌트

> *파이썬에서 재귀 함수를 정의할 수 있습니다.*

주요 저자 솔루션: 파이썬 2

```
def f(n):
    if n==0:
        return 0
    else:
        return f(n-1)+100 # n이 0이 아니면 f(n-1) + 100을 반환 (재귀 호출)

n=int(raw_input()) # 사용자로부터 n 입력
print f(n)
```

내 솔루션: 파이썬 3

```
def f(n):
    if n == 0:
        return 0
    return f(n-1) + 100

n = int(input())
print(f(n))
```

```
'''솔루션 작성자: NikolayEm
'''

n = int(input())
f = lambda x: f(x-1)+100 if x > 0 else 0 # 람다를 사용한 재귀 함수 정의
print(f(n))
```

문제 61

문제

피보나치 수열은 다음 공식에 따라 계산됩니다:

```
f(n)=0 (n=0 일 때)
f(n)=1 (n=1 일 때)
f(n)=f(n-1)+f(n-2) (n>1 일 때)
```

콘솔에서 주어진 n에 대해 f(n) 값을 계산하는 프로그램을 작성하십시오.

예시: 다음 n이 프로그램에 입력으로 주어지면:

```
7
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
13
```

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

힌트

> *파이썬에서 재귀 함수를 정의할 수 있습니다.*

주요 저자 솔루션: 파이썬 2

```
def f(n):
    if n == 0: return 0
    elif n == 1: return 1
    else: return f(n-1)+f(n-2) # n이 0 또는 1이 아니면 f(n-1) + f(n-2) 반환 (재귀 호출)

n=int(raw_input())
print f(n)
```

내 솔루션: 파이썬 3

```
def f(n):
    if n < 2: # n이 0 또는 1이면 n을 반환
        return n
    return f(n-1) + f(n-2) # 그렇지 않으면 f(n-1) + f(n-2) 반환

n = int(input())
print(f(n))
```

```
'''솔루션 작성자: NikolayEm
'''

n = int(input())
# 람다를 사용하여 피보나치 함수 정의
f = lambda x: 0 if x == 0 else 1 if x == 1 else f(x-1)+f(x-2)
# 0부터 n까지의 피보나치 수열을 쉼표로 구분하여 출력 (문제에서는 f(n)만 요구했지만, 이 솔루션은 전체 수열을 출력)
print(','.join([str(f(x)) for x in range(0, n+1)]))
```

문제 62

문제

피보나치 수열은 다음 공식에 따라 계산됩니다:

```
f(n)=0 (n=0 일 때)
f(n)=1 (n=1 일 때)
f(n)=f(n-1)+f(n-2) (n>1 일 때)
```

콘솔에서 주어진 n에 대해 f(n) 값을 계산하는 프로그램을 작성하십시오. (주: 문제 설명은 f(n)을 요구하지만, 예제 출력은 0부터 n까지의 피보나치 수열입니다. 예제 출력에 맞추겠습니다.)

예시: 다음 n이 프로그램에 입력으로 주어지면:

```
7
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
0,1,1,2,3,5,8,13
```

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

힌트

> ***파이썬에서 재귀 함수를 정의할 수 있습니다. 기존 리스트에서 리스트를 생성하려면 리스트 컴프리헨션을 사용하십시오. 문자열 리스트를 결합하려면 string.join()을 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
def f(n):
    if n == 0: return 0
    elif n == 1: return 1
    else: return f(n-1)+f(n-2)

n=int(raw_input())
values = [str(f(x)) for x in range(0, n+1)] # 0부터 n까지 각 x에 대해 f(x)를 계산하고 문자열로 변환하여 리스트 생성
print ",".join(values) # 리스트의 요소들을 쉼표로 구분하여 출력
```

내 솔루션: 파이썬 3 (메모이제이션 사용)

```
def f(n):
    if n < 2:
        fibo[n] = n
        return fibo[n]
    # fibo[n]이 이미 계산되었다면 해당 값을 사용 (메모이제이션)
    if fibo[n] == 0 and n != 0 : # 0번째 값은 0이므로, n이 0이 아닐 때만 fibo[n]이 0인지 확인
        fibo[n] = f(n-1) + f(n-2)
    elif n==0 : # fibo[0]은 0으로 초기화 되어 있으므로 그대로 반환
        return fibo[n]
    # 이미 계산된 값이 있다면 바로 반환 (이 부분은 위의 if 조건과 중복될 수 있으므로 로직 수정 필요)
    # 아래와 같이 수정하는 것이 더 명확합니다.
    # if fibo[n] != -1: # -1 등으로 초기화했다고 가정
    #    return fibo[n]
    # if n < 2: etc.

    # 현재 로직은 fibo 리스트를 외부에서 접근하고 수정합니다.
    # 더 나은 방법은 fibo 리스트를 f 함수의 인자로 전달하거나 클래스 멤버로 만드는 것입니다.
    # 아래는 주어진 코드 구조를 최대한 유지한 수정입니다.
    if fibo[n] == 0 and n!=0 : # 아직 계산되지 않았고, n이 0이 아니면 계산
        fibo[n] = f(n-1) + f(n-2)
    # n이 0이거나 n=1인 경우는 이미 위에서 처리됨 (fibo[0]=0, fibo[1]=1)
    # 또는 이미 계산된 경우 (fibo[n]이 0이 아닌 값을 가짐)
    return fibo[n]


n = int(input())
fibo = [0]*(n+1)  # (n+1) 크기의 리스트를 0으로 초기화
# 초기 조건 설정 (f(0)=0, f(1)=1인 경우)
if n>=0:
    fibo[0] = 0
if n>=1:
    fibo[1] = 1 # 이 부분을 f 함수 내부에서 처리하거나, f 함수 호출 전에 명시적으로 설정해야 함.
                # 현재 f 함수는 fibo[1]을 설정하지만, f(n) 호출 시 n<2 조건에서 fibo[n]=n으로 설정.

# f(n)을 호출하여 fibo 리스트를 채웁니다.
# 가장 큰 값부터 계산하도록 f(n)을 호출하면 작은 값들이 재귀적으로 계산됩니다.
# 하지만, 현재 f 함수는 fibo 리스트를 직접 수정하므로,
# 단순히 f(n)을 호출하는 것만으로는 fibo 리스트 전체가 올바르게 채워지지 않을 수 있습니다.
```

```
# 예: f(7) 호출 시 f(0)부터 f(7)까지 채워지도록 의도된 것으로 보입니다.

# 보다 일반적인 메모이제이션 접근 방식:
# fibo = [-1]*(n+1) # -1로 초기화하여 계산 여부 확인
# def f_memo(k):
#   if k < 2: return k
#   if fibo[k] != -1: return fibo[k]
#   fibo[k] = f_memo(k-1) + f_memo(k-2)
#   return fibo[k]
# result_list = [str(f_memo(i)) for i in range(n + 1)]
# print(",".join(result_list))

# 주어진 솔루션의 의도를 따라가겠습니다.
# f(n)을 호출하면 fibo[0]부터 fibo[n]까지 채워진다고 가정.
# (실제로는 f 함수 내부에서 n<2일 때 fibo[n]=n으로 할당하고, 그 외에는
fibo[n] = f(n-1)+f(n-2)로 할당)
# 이 방식은 fibo 리스트가 f 함수 외부 스코프에 정의되어 있고, f가 이를 직접
수정하는 방식입니다.

# fibo 리스트를 f 함수가 올바르게 채우도록 하려면, f 함수가 호출될 때마다
# 해당 인덱스의 값이 이미 채워져 있는지 확인하고, 아니라면 계산해야 합니다.
# 현재 코드는 f(n)을 한 번 호출합니다.

# 명확성을 위해, 각 f(i)를 호출하여 fibo 리스트를 채우는 방식으로 변경하거나,
# f 함수가 정확히 메모이제이션을 수행하도록 수정합니다.
# 여기서는 문제의 원래 의도(f(n) 호출로 전체 리스트 생성)를 따르기 어렵지만,
# 각 요소를 순서대로 계산하는 방식으로 수정하여 제시합니다.
for i in range(n + 1):
    if i < 2:
        fibo[i] = i
    else:
        fibo[i] = fibo[i-1] + fibo[i-2]

fibo_str = [str(i) for i in fibo]   # 정수 데이터를 문자열 유형으로 변환
ans = ",".join(fibo_str)    # fibo의 모든 문자열 요소를 ',' 문자로 결합
print(ans)
```

```
'''솔루션 작성자: popomaticbubble
'''
def fibo(n): # 단순 재귀 피보나치 함수
    if n < 2: return n
    return fibo(n-1)+fibo(n-2)

def print_fiblist(n):
    # 0부터 n까지 각 i에 대해 fibo(i)를 호출하고 문자열로 변환하여 리스트 생
성
    fib_list = [(str(fibo(i))) for i in range(0, n+1)]
    return print(",".join(fib_list)) # 리스트를 쉼표로 구분하여 출력
n = int(input())
print_fiblist(n)
```

```
'''솔루션 작성자: lcastrooliveira
'''
# 반복적 피보나치 수열 생성 함수
def question_62(n):
    if n == 0:
        return [0]
    if n == 1:
        return [0, 1]
    sequence = [0, 1] # 초기값 설정
    a, b = 0, 1 # 이전 두 값
    for x in range(2, n+1): # 2부터 n까지 반복
        c = a + b # 다음 피보나치 수 계산
        sequence.append(c) # 수열에 추가
        a = b # 값 업데이트
        b = c
    return sequence

print(question_62(10)) # 예시로 10까지의 피보나치 수열 출력
```

문제 63

문제

제너레이터를 사용하여 0과 n 사이의 짝수를 쉼표로 구분된 형태로 인쇄하는 프로그램을 작성하십시오. n은 콘솔을 통해 입력됩니다.

예시: 다음 n이 프로그램에 입력으로 주어지면:

```
10
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
0,2,4,6,8,10
```

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

힌트

제너레이터에서 다음 값을 생성하려면 yield를 사용하십시오.

솔루션: (Python 2 기준)

```
def EvenGenerator(n): # 0부터 n까지 짝수를 생성하는 제너레이터
    i=0
    while i<=n:
        if i%2==0:
            yield i # i가 짝수이면 반환
        i+=1


n=int(raw_input()) # Python 2 입력 방식
values = []
for i in EvenGenerator(n):
    values.append(str(i)) # 생성된 짝수를 문자열로 변환하여 리스트에 추가

print ",".join(values) # 쉼표로 구분하여 출력
```

또는 (Python 3 기준)

```
# 솔루션 작성자: StartZer0
n = int(input()) # Python 3 입력 방식

for i in range(0, n+1, 2): # 0부터 n까지 2씩 증가하며 반복 (즉, 짝수만)
  if i < n -1 and i != n : # 현재 짝수가 n-1보다 작고 n이 아니면 (즉, 마지막 짝수가 아니면)
    print(i, end = ',' )   # 쉼표와 함께 출력
  else: # 마지막 짝수이면
    print(i)               # 숫자만 출력 (쉼표 없음)
# 위 StartZer0 솔루션의 조건 i < n - 1은 마지막 두 짝수에 대해 다르게 동작할 수 있습니다.
# 예를 들어 n=10일 때, i=8이면 8 < 9 참, 8 출력. i=10일 때 10 < 9 거짓, 10 출력.
# n=9일 때, i=8이면 8 < 8 거짓, 8 출력.
# 좀 더 간단한 방법:
# values = []
# for i in EvenGenerator(n):
#   values.append(str(i))
# print(",".join(values))
# 또는 제너레이터 직접 사용:
# print(",".join(str(i) for i in EvenGenerator(n)))
```

문제 64

문제

제너레이터를 사용하여 0과 n 사이에서 5와 7로 모두 나누어지는 숫자를 쉼표로 구분된 형태로 인쇄하는 프로그램을 작성하십시오. n은 콘솔을 통해 입력됩니다.

예시: 다음 n이 프로그램에 입력으로 주어지면:

```
100
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
0,35,70
```

질문에 입력 데이터가 제공되는 경우 콘솔 입력으로 간주해야 합니다.

힌트

제너레이터에서 다음 값을 생성하려면 yield를 사용하십시오.

주요 저자 솔루션: 파이썬 2

```
def NumGenerator(n): # 0부터 n까지 5와 7로 모두 나누어지는 숫자를 생성하는 제너레이터
    for i in range(n+1):
        if i%5==0 and i%7==0: # 5와 7로 모두 나누어지면
            yield i # 반환

n=int(raw_input())
values = []
for i in NumGenerator(n):
    values.append(str(i))

print ",".join(values)
```

내 솔루션: 파이썬 3

```
def generate(n):
    for i in range(n+1):
        if i % 35 == 0:    # 5*7 = 35, 숫자가 a와 b로 나누어지면 a*b로도 나누어집니다 (a, b가 서로소일 때).
            yield i

n = int(input())
resp = [str(i) for i in generate(n)] # 제너레이터에서 생성된 각 숫자를 문자열로 변환하여 리스트 생성
print(",".join(resp)) # 쉼표로 구분하여 출력
```

이전 날로 가기

다음 날로 가기

토론

문제 65

문제

> *리스트 [2,4,6,8]의 모든 숫자가 짝수인지 확인하는 assert 문을 작성하십시오.*

힌트

> *단언문을 만들려면 "assert 표현식"을 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
li = [2,4,6,8]
for i in li:
    assert i%2==0 # i를 2로 나눈 나머지가 0인지 확인합니다. 0이 아니면
AssertionError 발생
```

내 솔루션: 파이썬 3

```
data = [2,4,5,6] # 리스트에 홀수 5를 포함하여 assert가 실패하도록 함
for i in data:
    assert i%2 == 0, "{}는 짝수가 아닙니다".format(i) # 조건이 거짓이면
메시지와 함께 AssertionError 발생
```

문제 66

문제

> ***콘솔에서 기본 수학 표현식을 입력받아 평가 결과를 인쇄하는 프로그램을 작성하십시오.***

> ***예시: 다음 n이 프로그램에 입력으로 주어지면:*** (주: n이 아니라 표현식)

```
35 + 3
```

> ***그러면 프로그램의 출력은 다음과 같아야 합니다:***

```
38
```

힌트

> ***표현식을 평가하려면 eval()을 사용하십시오.*** (주의: eval()은 안전하지 않은 입력을 평가할 수 있으므로 신뢰할 수 없는 소스의 입력에는 사용하지 않는 것이 좋습니다.)

주요 저자 솔루션: 파이썬 2

```
expression = raw_input() # 사용자로부터 표현식 입력
print eval(expression) # 표현식 평가 및 결과 출력
```

내 솔루션: 파이썬 3

```
expression = input()
ans = eval(expression)
print(ans)
```

문제 67

문제

> *정렬된 리스트에서 항목을 검색하는 이진 검색 함수를 작성하십시오. 함수는 리스트에서 검색할 요소의 인덱스를 반환해야 합니다.*

힌트

> *조건을 처리하려면 if/elif를 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
import math
def bin_search(li, element):
    bottom = 0
    top = len(li)-1
    index = -1 # 요소를 찾지 못한 경우 -1 반환
    while top>=bottom and index==-1: # 검색 범위가 유효하고 아직 찾지 못했다면 반복
        mid = int(math.floor((top+bottom)/2.0)) # 중간 인덱스 계산 (내림)
        if li[mid]==element: # 중간 요소가 찾는 요소와 같으면
            index = mid # 인덱스 저장
        elif li[mid]>element: # 중간 요소가 더 크면
            top = mid-1 # 검색 범위를 왼쪽 절반으로 줄임
        else: # 중간 요소가 더 작으면
            bottom = mid+1 # 검색 범위를 오른쪽 절반으로 줄임

    return index

li=[2,5,7,9,11,17,222]
print bin_search(li,11) # 11 검색 (결과: 4)
print bin_search(li,12) # 12 검색 (결과: -1)
```

내 솔루션: 파이썬 3

```
# 작성 예정 (아래 다른 사람들의 솔루션 참고)
```

ulmasovjafarbek 솔루션: 파이썬 3

```
def binary_search(lst, item):
    low = 0
    high = len(lst) - 1

    while low <= high: # low가 high보다 작거나 같을 동안 반복
        mid = round((low + high) / 2) # 중간 인덱스 (반올림, Python 3에서는 // (정수 나누기)가 더 일반적)
                                      # mid = (low + high) // 2 로 수정하는 것이 좋음

        if lst[mid] == item: # 찾았으면 인덱스 반환
            return mid
        elif lst[mid] > item: # 찾는 값이 더 작으면 high를 줄임
            high = mid - 1
        else: # 찾는 값이 더 크면 low를 늘림
            low = mid + 1
    return None # 찾지 못하면 None 반환

lst = [1,3,5,7,]
print(binary_search(lst, 9)) # 9는 리스트에 없으므로 None 출력
```

AasaiAlangaram 솔루션: 파이썬 3 (이진 검색 변형)

```
def binary_search_Ascending(array, target): # 오름차순 정렬된 배열 대상
    lower = 0
    upper = len(array) # upper를 len(array) - 1이 아닌 len(array)로 설
정
    print('배열 길이:',upper)
    while lower < upper: # lower가 upper보다 작을 동안 반복 (일반적으로는
lower <= upper)
        x = (lower + upper) // 2 # 중간 인덱스
        print('중간 값 인덱스:',x) # 'Middle Value'를 '중간 값 인덱스'로
번역
        value = array[x]
        if target == value:
            return x
        elif target > value:
            lower = x + 1 # target이 더 크면 lower를 x + 1로
(AasaiAlangaram 솔루션에서는 lower = x)
                          # 일반적인 이진 검색에서는 검색 범위를 확실히 줄
이기 위해 x + 1 또는 x - 1 사용
        elif target < value:
            upper = x # target이 더 작으면 upper를 x로 (AasaiAlangaram
솔루션에서는 upper = x)
                      # 일반적인 이진 검색에서는 upper = x - 1
                      # 현재 로직은 특정 상황에서 무한 루프나 잘못된 결과를
초래할 수 있음.
                      # 예를 들어 target이 value보다 클 때 lower = x로
설정하면, x가 계속 같은 값일 경우 무한 루프 가능성.
                      # 표준 이진 검색 로직을 따르는 것이 좋음.

Array = [1,5,8,10,12,13,55,66,73,78,82,85,88,99]
print('값을 찾은 인덱스:',binary_search_Ascending(Array, 82)) # 'The
Value Found at Index:'를 번역
```

참고: AasaiAlangaram의 솔루션은 표준 이진 검색과 약간 다른 업데이트 로직(lower = x, upper = x)을 사용하고 있어, 특정 케이스에서 문제가 발생할 수 있습니다. 표준적인 방식은 `lower = mid + 1` 과 `upper = mid - 1` 을 사용하는 것입니다.

---

yuan1z 솔루션: 파이썬 3 (재귀적 이진 검색)

```
idx = 0 # 전역 변수 idx를 사용하여 최종 인덱스 추적 (재귀 호출 시 인덱스 보정용)
def bs(num,num_list):
    global idx # 전역 변수 idx 사용 선언
    if (len(num_list) == 1): # 리스트에 요소가 하나만 남았을 때
        if num_list[0] == num: # 그 요소가 찾는 숫자이면
            return idx # 현재까지 누적된 idx 반환
        else: # 아니면
            return "리스트에 없음" # "No exit in the list"를 번역
    elif num_list[len(num_list)//2 -1] >= num : # 찾는 숫자가 왼쪽 절반의 마지막 요소보다 작거나 같으면 (num in num_list[:len(num_list)//2] 와 유사한 로직)
                                                # (수정 제안: num <= num_list[len(num_list)//2 -1] 또는 num < num_list[len(num_list)//2] )
        return bs(num,num_list[:len(num_list)//2]) # 왼쪽 절반으로 재귀 호출
    else: # 찾는 숫자가 오른쪽 절반에 있다면
        idx += len(num_list)//2 # 오른쪽 절반으로 이동했으므로 idx에 왼쪽 절반의 길이를 더함
    return bs(num,num_list[len(num_list)//2:]) # 오른쪽 절반으로 재귀 호출

print(bs(66,[1,5,8,10,12,13,55,66,73,78,82,85,88,99,100]))
```

참고: yuan1z의 솔루션은 전역 변수를 사용하여 인덱스를 관리하며, 재귀적으로 리스트를 슬라이싱합니다. 리스트 슬라이싱은 새 리스트를 생성하므로 매우 큰 리스트에서는 비효율적일 수 있습니다. 인덱스를 직접 전달하는 방식이 더 효율적입니다. 조건 `num_list[len(num_list)//2 -1] >= num` 부분도 주의 깊게 검토 필요.

문제 68

문제

> ***파이썬 모듈을 사용하여 10과 100 사이의 임의의 부동 소수점 숫자를 생성하십시오.***

힌트

> ***random.random()을 사용하여 [0,1] 범위의 임의의 부동 소수점 숫자를 생성하십시오.***

주요 저자 솔루션: 파이썬 2

```
import random
print random.random()*100 # random.random()은 0.0 <= x < 1.0 범위의
부동 소수점 반환. 여기에 100을 곱하면 0.0 <= x < 100.0.
                          # 10과 100 사이를 정확히 맞추려면 조정 필요:
random.random() * 90 + 10
```

내 솔루션: 파이썬 3

```
import random
rand_num = random.uniform(10,100) # uniform(a, b)는 a <= x <= b (또는
a <= x < b, 문서 확인 필요) 범위의 부동 소수점 반환
                                  # random.uniform(a,b)는 a <= N <=
b 를 만족하는 임의의 부동소수점 숫자를 반환합니다.
print(rand_num)
```

문제 69

문제

> ***파이썬 모듈을 사용하여 5와 95 사이의 임의의 부동 소수점 숫자를 생성하십시오.***

힌트

> ***random.random()을 사용하여 [0,1] 범위의 임의의 부동 소수점 숫자를 생성하십시오.***

주요 저자 솔루션: 파이썬 2

```
import random
print random.random()*100-5 # 0.0 <= x < 1.0  =>  0.0 <= x*100 <
100.0  => -5.0 <= x*100-5 < 95.0
                            # 이 방식은 5와 95 사이를 정확히 보장하지 않
음. (최솟값 -5 가능)
                            # 올바른 방식: random.random() * 90 + 5
```

내 솔루션: 파이썬 3

```
import random
rand_num = random.uniform(5,95) # 5 <= N <= 95 범위의 부동 소수점 반환
print(rand_num)
```

이전 날로 가기

다음 날로 가기

토론

문제 70

문제

> ***random 모듈과 리스트 컴프리헨션을 사용하여 0과 10 사이(경계값 포함)의 임의의 짝수를 출력하는 프로그램을 작성하십시오.***

힌트

> ***리스트에서 임의의 요소를 선택하려면 random.choice()를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
li = [2,4,6,8] # 이 리스트는 사용되지 않습니다.
import random
# 0부터 10까지 숫자 중 짝수인 i로 리스트를 만들고, 그 중에서 무작위로 하나 선택
print random.choice([i for i in range(11) if i%2==0])
```

내 솔루션: 파이썬 3

```
import random
resp = [i for i in range(0,11,2)] # 0부터 10까지 2씩 증가하는 숫자(짝수)로 리스트 생성
print(random.choice(resp)) # 리스트에서 무작위로 하나 선택하여 출력
```

문제 71

문제

> ***random 모듈과 리스트 컴프리헨션을 사용하여 10과 150 사이(경계값 포함)에서 5와 7로 모두 나누어지는 임의의 숫자를 출력하는 프로그램을 작성하십시오.***

힌트

> ***리스트에서 임의의 요소를 선택하려면 random.choice()를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
import random
# 10부터 150까지 숫자 중 5와 7로 모두 나누어지는 i로 리스트를 만들고, 그 중에서 무작위로 하나 선택
print random.choice([i for i in range(10,151) if i%5==0 and i%7==0])
```

내 솔루션: 파이썬 3

```
import random
# 10부터 150까지 숫자 중 35 (5*7)로 나누어지는 i로 리스트 생성
resp = [i for i in range(10,151) if i % 35 == 0 ]
print(random.choice(resp)) # 리스트에서 무작위로 하나 선택하여 출력
```

문제 72

문제

> ***100과 200 사이(경계값 포함)의 임의의 숫자 5개로 구성된 리스트를 생성하는 프로그램을 작성하십시오.***

힌트

> ***임의의 값으로 구성된 리스트를 생성하려면 random.sample()을 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
import random
# range(100,201) (100부터 200까지)에서 중복 없이 5개의 숫자를 무작위로 선택하여 리스트 생성
print random.sample(range(100,201), 5)
```

내 솔루션: 파이썬 3

```
import random
resp = random.sample(range(100,201),5) # range(100,201)에서 5개의 샘플 추출
print(resp)
```

문제 73

문제

> ***100과 200 사이(경계값 포함)의 임의의 짝수 5개로 구성된 리스트를 무작위로 생성하는 프로그램을 작성하십시오.***

힌트

> ***임의의 값으로 구성된 리스트를 생성하려면 random.sample()을 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
import random
# 100부터 200까지 숫자 중 짝수인 i로 리스트를 만들고, 그 중에서 중복 없이 5개의 숫자를 무작위로 선택
print random.sample([i for i in range(100,201) if i%2==0], 5)
```

내 솔루션: 파이썬 3

```
import random
# range(100,201,2)는 100부터 200까지 짝수만 생성
resp = random.sample(range(100,201,2),5)
print(resp)
```

문제 74

문제

> *1과 1000 사이(경계값 포함)에서 5와 7로 모두 나누어지는 임의의 숫자 5개로 구성된 리스트를 무작위로 생성하는 프로그램을 작성하십시오.*

힌트

> *임의의 값으로 구성된 리스트를 생성하려면 random.sample()을 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
import random
# 1부터 1000까지 숫자 중 5와 7로 모두 나누어지는 i로 리스트를 만들고, 그 중에서 중복 없이 5개의 숫자를 무작위로 선택
print random.sample([i for i in range(1,1001) if i%5==0 and i%7==0], 5)
```

내 솔루션: 파이썬 3

```
import random
# 1부터 1000까지 숫자 중 35로 나누어지는 i로 리스트 생성
lst = [i for i in range(1,1001) if i%35 == 0]
resp = random.sample(lst,5) # 위 리스트에서 5개의 샘플 추출
print(resp)
```

이전 날로 가기

다음 날로 가기

토론

문제 75

문제

> *7과 15 사이(경계값 포함)의 정수를 무작위로 인쇄하는 프로그램을 작성하십시오.*

힌트

> *주어진 범위 내에서 임의의 정수를 얻으려면 random.randrange()를 사용하십시오.*

솔루션:

```
import random
# random.randrange(start, stop)는 start <= x < stop 범위의 정수를 반환합니다.
# 따라서 7과 15를 포함하려면 stop을 16으로 설정해야 합니다.
print random.randrange(7,16)
```

문제 76

문제

> ***문자열 "hello world!hello world!hello world!hello world!"를 압축하고 압축 해제하는 프로그램을 작성하십시오.***

힌트

> ***문자열을 압축하고 압축 해제하려면 zlib.compress()와 zlib.decompress()를 사용하십시오.***

솔루션: (Python 2 기준)

```
import zlib
s = 'hello world!hello world!hello world!hello world!'
t = zlib.compress(s) # 문자열 s를 압축합니다. (Python 2에서는 문자열을 직접 압축 가능)
print t # 압축된 바이트 시퀀스 출력
print zlib.decompress(t) # 압축 해제하여 원본 문자열 출력
```

```
'''anas1434 솔루션: (Python 3 기준)
'''
s = 'hello world!hello world!hello world!hello world!'
# 파이썬 3에서 zlib.compress()는 <bytes> 데이터 유형만 허용합니다.
y = bytes(s, 'utf-8') # 문자열 s를 UTF-8 인코딩을 사용하여 바이트로 변환
x = zlib.compress(y) # 바이트 시퀀스 y를 압축
print(x) # 압축된 바이트 시퀀스 출력
print(zlib.decompress(x)) # 압축 해제하여 원본 바이트 시퀀스 출력 (출력 시 자동으로 문자열로 변환될 수 있음)
```

문제 77

문제

> ***"1+1"을 100번 실행하는 데 걸리는 시간을 인쇄하는 프로그램을 작성하십시오.***

힌트

> ***실행 시간을 측정하려면 timeit() 함수를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
from timeit import Timer
# Timer 객체 생성. 첫 번째 인자는 실행할 코드, 두 번째 인자는 설정 코드 (여기서는 없음)
t = Timer("for i in range(100):1+1")
# timeit() 메소드는 코드를 여러 번 실행하여 평균 실행 시간을 계산합니다. (기본값: 1,000,000번)
# 문제에서는 "1+1"을 100번 실행하는 것 자체의 시간을 측정하라는 의미로 해석될 수 있으나,
# timeit은 해당 코드 블록(여기서는 "for i in range(100):1+1")의 실행 시간을 측정합니다.
print t.timeit()
```

내 솔루션: 파이썬 3 (datetime 사용)

```
import datetime

before = datetime.datetime.now() # 실행 전 현재 시간
for i in range(100):
    x = 1 + 1
after = datetime.datetime.now() # 실행 후 현재 시간
execution_time = after - before # 실행 시간 계산 (timedelta 객체)
print(execution_time.microseconds) # 마이크로초 단위로 출력
```

또는 (time 모듈 사용)

```
import time

before = time.time() # 실행 전 현재 시간 (초 단위 타임스탬프)
for i in range(100):
    x = 1 + 1
after = time.time() # 실행 후 현재 시간
execution_time = after - before # 실행 시간 계산 (초 단위)
print(execution_time)
```

문제 78

문제

> ***리스트 [3,6,7,8]을 섞어서(shuffle) 인쇄하는 프로그램을 작성하십시오.***

힌트

> ***리스트를 섞으려면 shuffle() 함수를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
from random import shuffle # random 모듈에서 shuffle 함수 가져오기
li = [3,6,7,8]
shuffle(li) # 리스트 li를 직접 섞습니다 (in-place). 반환 값은 None입니다.
print li # 섞인 리스트 출력
```

내 솔루션: 파이썬 3

```
import random

lst = [3,6,7,8]
random.shuffle(lst) # 리스트 lst를 직접 섞습니다.
print(lst)
```

또는 (특정 시드 사용)

```
import random

# 특정 시드로 섞기 (결과가 항상 동일하게 나옴)
lst = [3,6,7,8]
seed = 7
random.Random(seed).shuffle(lst) # 시드 값을 사용하여 Random 객체를 만들고 shuffle 호출
print(lst)
```

문제 79

문제

> ***주어가 ["I", "You"] 중 하나이고, 동사가 ["Play", "Love"] 중 하나이며, 목적어가 ["Hockey","Football"] 중 하나인 모든 문장을 생성하는 프로그램을 작성하십시오.***

힌트

> *리스트에서 요소를 가져오려면 list[index] 표기법을 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
subjects=["I", "You"]
verbs=["Play", "Love"]
objects=["Hockey","Football"]
for i in range(len(subjects)):
    for j in range(len(verbs)):
        for k in range(len(objects)):
            sentence = "%s %s %s." % (subjects[i], verbs[j],
objects[k]) # 문자열 포매팅
            print sentence
```

내 솔루션: 파이썬 3

```
subjects=["I", "You"]
verbs=["Play", "Love"]
objects=["Hockey","Football"]

for sub in subjects:
    for verb in verbs:
        for obj in objects:
            print("{} {} {}.".format(sub,verb,obj)) # 마침표 추가
```

```
'''popomaticbubble 솔루션:
'''
import itertools # itertools 모듈 가져오기
subject = ["I", "You"]
verb = ["Play", "Love"]
objects = ["Hockey","Football"]

sentence_parts = [subject, verb, objects] # 리스트들을 포함하는 리스트
# itertools.product는 여러 반복 가능한 객체들의 데카르트 곱을 반환합니다.
# *sentence_parts는 sentence_parts 리스트의 각 요소를 product 함수의 인자로 전달합니다.
n = list(itertools.product(*sentence_parts)) # 가능한 모든 조합을 튜플 형태로 생성
for i in n:
    print(i) # 각 조합 (튜플) 출력 (예: ('I', 'Play', 'Hockey'))
    # 문장 형태로 출력하려면: print("{} {} {}.".format(i[0], i[1], i[2]))
```

```
'''lcastrooliveira 솔루션:
'''
from itertools import product # itertools에서 product 함수 가져오기

def question_79():
    subject = ["I", "You"]
    verb = ["Play", "Love"]
    object_list = ["Hockey", "Football"] # 변수명을 object에서 object_list로 변경 (object는 파이썬 내장 타입과 충돌)
    # product(range(2), repeat=3)는 (0,0,0)부터 (1,1,1)까지의 모든 조합을 생성 (각 리스트의 인덱스로 사용)
    prod = [p for p in product(range(2), repeat=3)] # 각 리스트의 크기가 2이므로 range(2) 사용
    for combination in prod:
        # f-string을 사용하여 문장 구성
        print(f'{subject[combination[0]]} {verb[combination[1]]} {object_list[combination[2]]}.') # 마침표 추가

question_79()
```

이전 날로 가기

다음 날로 가기

토론

문제 80

문제

> *[5,6,77,45,22,12,24] 리스트에서 짝수를 제거한 후 리스트를 인쇄하는 프로그램을 작성하십시오.*

힌트

> *리스트에서 여러 요소를 삭제하려면 리스트 컴프리헨션을 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
li = [5,6,77,45,22,12,24]
li = [x for x in li if x%2!=0] # x가 홀수이면 리스트에 포함
print li
```

내 솔루션: 파이썬 3

```
def isOdd(n): # 홀수인지 확인하는 함수 (원래 isEven이었으나, 홀수를 남기므로 isOdd로 변경)
    return n%2!=0

li = [5,6,77,45,22,12,24]
lst = list(filter(isOdd,li)) # isOdd가 True를 반환하는 요소만 필터링
print(lst)
```

또는

```
li = [5,6,77,45,22,12,24]
lst = list(filter(lambda n:n%2!=0,li)) # 람다 함수를 사용하여 홀수만 필터링
print(lst)
```

문제 81

문제

리스트 컴프리헨션을 사용하여 [12,24,35,70,88,120,155]에서 5와 7로 나누어지는 숫자를 제거한 후 리스트를 인쇄하는 프로그램을 작성하십시오.

힌트

리스트에서 여러 요소를 삭제하려면 리스트 컴프리헨션을 사용하십시오.

주요 저자 솔루션: 파이썬 2

```
li = [12,24,35,70,88,120,155]
# x가 5로 나누어지지 않고 그리고 7로도 나누어지지 않으면 리스트에 포함
li = [x for x in li if x%5!=0 and x%7!=0]
print li
```

내 솔루션: 파이썬 3

```
li = [12,24,35,70,88,120,155]
# x가 35 (5와 7의 최소공배수)로 나누어지지 않으면 리스트에 포함
li = [x for x in li if x % 35!=0]
print(li)
```

문제 82

문제

> ***리스트 컴프리헨션을 사용하여 [12,24,35,70,88,120,155]에서 0번째, 2번째, 4번째, 6번째 숫자를 제거한 후 리스트를 인쇄하는 프로그램을 작성하십시오.***

힌트

> ***리스트에서 여러 요소를 삭제하려면 리스트 컴프리헨션을 사용하십시오. (인덱스, 값) 튜플을 얻으려면 enumerate()를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
li = [12,24,35,70,88,120,155]
# enumerate(li)는 (인덱스, 값) 튜플을 생성합니다.
# 인덱스 i가 홀수이고(i%2 != 0) 6 이하인 요소 x만 리스트에 포함합니다. (즉, 1번째, 3번째, 5번째 요소)
li = [x for (i,x) in enumerate(li) if i%2 != 0 and i <= 6]
print li
```

참고: 문제에서는 "0번째, 2번째, 4번째, 6번째 숫자를 제거"하라고 했으므로, 결과적으로 "1번째, 3번째, 5번째" 요소를 남기는 것이 맞습니다. 위 솔루션은 이 요구사항을 정확히 반영합니다.

내 솔루션: 파이썬 3

```python
li = [12,24,35,70,88,120,155]
# range(len(li))로 인덱스를 생성하고, 인덱스 i가 홀수이고 6 이하인 요소
li[i]를 리스트에 포함합니다.
li = [li[i] for i in range(len(li)) if i%2 != 0 and i <= 6]
print(li)
```

```python
'''popomaticbubble 솔루션:
'''
orig_lst = [12,24,35,70,88,120,155]
indices_to_remove = [0, 2, 4, 6] # 제거할 인덱스 리스트

# enumerate를 사용하여 (인덱스 j, 값 i) 쌍을 가져옵니다.
# 인덱스 j가 indices_to_remove 리스트에 포함되어 있지 않으면 값 i를
new_list에 추가합니다.
new_list = [i for (j, i) in enumerate(orig_lst) if j not in
indices_to_remove]
print(new_list)
```

문제 83

문제

리스트 컴프리헨션을 사용하여 [12,24,35,70,88,120,155]에서 2번째부터 4번째까지의 숫자를 제거한 후 리스트를 인쇄하는 프로그램을 작성하십시오. (주: 인덱스 기준으로는 1번째부터 3번째까지의 요소를 제거하는 것을 의미할 수 있습니다. "2nd - 4th numbers"가 인덱스 1, 2, 3을 의미하는 것으로 해석하겠습니다. 즉, 24, 35, 70을 제거합니다.)

힌트

> ***리스트에서 여러 요소를 삭제하려면 리스트 컴프리헨션을 사용하십시오. (인덱스, 값) 튜플을 얻으려면 enumerate()를 사용하십시오.***

주요 저자 솔루션: 파이썬 2 (주: "2nd - 4th numbers"를 인덱스 1, 2, 3으로 해석하여 해당 요소를 제거)

```
li = [12,24,35,70,88,120,155]
# 인덱스 i가 1보다 작거나(i<1, 즉 0번째) 또는 3보다 큰(i>3, 즉 4번째부터) 요소 x만 포함
# 이렇게 하면 인덱스 1, 2, 3 (값 24, 35, 70)이 제거됩니다.
# 저자 솔루션은 "if i<3 or 4<i"로 되어있는데, 이는 인덱스 3, 4를 제거합니다 (값 70, 88).
# 문제의 "2nd - 4th numbers"를 값 24, 35, 70 (인덱스 1, 2, 3)로 해석하고 아래와 같이 수정 제안:
# li = [x for (i,x) in enumerate(li) if i < 1 or i > 3]
# 또는 popomaticbubble의 솔루션처럼 명시적으로 제외하는 것이 더 명확할 수 있습니다.
# 아래는 저자의 원래 코드입니다. 인덱스 3, 4 (값 70, 88)를 제거합니다.
li = [x for (i,x) in enumerate(li) if i<3 or 4<i] # 인덱스 3, 4를 제외 (즉, 0,1,2,5,6번째 요소 유지)
print li
```

내 솔루션: 파이썬 3 (저자 솔루션 로직과 동일하게 인덱스 3, 4 제거)

```
# 작성 예정
li = [12,24,35,70,88,120,155]
# 인덱스 i가 3보다 작거나(0,1,2) 또는 4보다 큰(5,6) 요소 li[i]를 리스트에 포함
li = [li[i] for i in range(len(li)) if i < 3 or i > 4]
print(li)
```

```
"""popomaticbubble 솔루션:
"""
# "2nd - 4th numbers"를 인덱스 1, 2, 3 (값 24, 35, 70)으로 해석하여 제거합니다.
orig_list = [12,24,35,70,88,120,155]
# 인덱스 j가 range(1,4) (즉, 1, 2, 3)에 포함되지 않으면 요소 i를 new_list에 추가
new_list = [i for (j, i) in enumerate(orig_list) if j not in range(1,4)]
print(new_list) # 결과: [12, 88, 120, 155]
```

```
"""saxenaharsh24 솔루션:
"""
# 이 솔루션은 lst.index(i)를 사용하는데, 리스트에 중복된 값이 있으면 의도치 않게 동작할 수 있습니다.
# 예를 들어 [10, 20, 10] 에서 2번째 10을 제거하려 해도 첫 번째 10의 인덱스가 사용됩니다.
# 문제의 리스트에는 중복 값이 없으므로 여기서는 동작합니다.
# "2nd - 4th numbers"를 인덱스 2, 3, 4 (값 35, 70, 88)로 해석하여 제거합니다.
lst = [12,24,35,70,88,120,155]
# 요소 i의 인덱스가 range(2,5) (즉, 2, 3, 4)에 포함되지 않으면 리스트에 추가
print([i for i in lst if lst.index(i) not in range(2,5)]) # 결과: [12, 24, 120, 155]
```

문제 84

문제

> ***리스트 컴프리헨션을 사용하여 각 요소가 0인 3x5x8 3차원 배열을 생성하는 프로그램을 작성하십시오.***

힌트

> ***배열을 만들려면 리스트 컴프리헨션을 사용하십시오.***

솔루션: (Python 2 기준 출력, Python 3에서도 동일하게 동작)

```
# 가장 안쪽부터: 8개의 0으로 된 리스트 (열)
# 중간: 위와 같은 리스트가 5개 있는 리스트 (행)
# 가장 바깥쪽: 위와 같은 2차원 리스트가 3개 있는 리스트 (깊이)
array = [[ [0 for col in range(8)] for col in range(5)] for row in
range(3)]
print array
```

이전 날로 가기

다음 날로 가기

토론

문제 85

문제

> *리스트 컴프리헨션을 사용하여 [12,24,35,70,88,120,155]에서 0번째, 4번째, 5번째 숫자를 제거한 후 리스트를 인쇄하는 프로그램을 작성하십시오.*

힌트

> *리스트에서 여러 요소를 삭제하려면 리스트 컴프리헨션을 사용하십시오. (인덱스, 값) 튜플을 얻으려면 enumerate()를 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
li = [12,24,35,70,88,120,155]
# enumerate(li)는 (인덱스, 값) 튜플을 생성합니다.
# 인덱스 i가 (0, 4, 5)에 포함되지 않은 요소 x만 리스트에 포함합니다.
li = [x for (i,x) in enumerate(li) if i not in (0,4,5)]
print li
```

내 솔루션: 파이썬 3

```
li = [12,24,35,70,88,120,155]
# range(len(li))로 인덱스를 생성하고, 인덱스 i가 (0, 4, 5)에 포함되지 않은
요소 li[i]를 리스트에 포함합니다.
li = [li[i] for i in range(len(li)) if i not in (0,4,5)]
print(li)
```

```
'''pratikb0501 솔루션:
'''
li = [12, 24, 35, 70, 88, 120, 155]
# enumerate를 사용하여 (인덱스 i, 값 j) 쌍을 가져옵니다.
# 인덱스 i가 0이 아니고, 4도 아니고, 5도 아니면 값 j를 리스트에 포함합니다.
print(list(j for i, j in enumerate(li) if i != 0 and i != 4 and i !=
5))
```

문제 86

문제

리스트 컴프리헨션을 사용하여 [12,24,35,24,88,120,155]에서 값 24를 제거한 후 리스트를 인쇄하는 프로그램을 작성하십시오. (주: 리스트 컴프리헨션을 사용하라는 것은 모든 24를 제거하라는 의미로 해석됩니다. `list.remove()` 는 첫 번째 발생만 제거합니다.)

힌트

값을 삭제하려면 리스트의 remove 메소드를 사용하십시오. (주: 이 힌트는 리스트 컴프리헨션 요구사항과 약간 상충됩니다. `remove` 는 한 번에 하나만 제거합니다.)

주요 저자 솔루션: 파이썬 2

```
li = [12,24,35,24,88,120,155]
li = [x for x in li if x!=24] # x가 24가 아니면 리스트에 포함 (모든 24 제거)
print li
```

내 솔루션: 파이썬 3

```
li = [12,24,35,24,88,120,155]
# li.remove(24)  # 이 코드는 첫 번째로 나오는 24만 제거합니다.
# 문제에서 리스트 컴프리헨션을 사용하라고 했고, 모든 24를 제거하는 것이 일반적인 의도이므로 아래와 같이 수정합니다.
li_filtered = [x for x in li if x != 24]
print(li_filtered)
```

문제 87

문제

> *주어진 두 리스트 [1,3,6,78,35,55]와 [12,24,35,24,88,120,155]에 대해, 위 두 리스트의 교집합 요소를 포함하는 리스트를 만드는 프로그램을 작성하십시오.*

힌트

> *집합 교집합 연산을 수행하려면 set()과 "&=" (또는 &)를 사용하십시오.*

주요 저자 솔루션: 파이썬 2

```
set1=set([1,3,6,78,35,55])
set2=set([12,24,35,24,88,120,155])
set1 &= set2 # set1을 set1과 set2의 교집합으로 업데이트합니다.
li=list(set1) # 교집합 결과를 리스트로 변환
print li
```

내 솔루션: 파이썬 3

```
list1 = [1,3,6,78,35,55]
list2 = [12,24,35,24,88,120,155]
set1 = set(list1)
set2 = set(list2)
intersection = set1 & set2 # & 연산자로 교집합을 구합니다.
print(list(intersection)) # 결과를 리스트로 변환하여 출력 (순서는 보장되지 않음)
```

또는

```
list1 = [1,3,6,78,35,55]
list2 = [12,24,35,24,88,120,155]
set1 = set(list1)
set2 = set(list2)
intersection = set.intersection(set1,set2) # intersection() 메소드 사용
print(list(intersection)) # 결과를 리스트로 변환하여 출력
```

문제 88

문제

> ***주어진 리스트 [12,24,35,24,88,120,155,88,120,155]에 대해, 원래 순서를 유지하면서 모든 중복 값을 제거한 후 이 리스트를 인쇄하는 프로그램을 작성하십시오.***

힌트

> ***중복 없이 여러 값을 저장하려면 set()을 사용하십시오.*** (주: set()만 사용하면 순서가 유지되지 않습니다. 순서 유지를 위해서는 다른 방법이 필요합니다.)

주요 저자 솔루션: 파이썬 2 (순서 유지)

```
def removeDuplicate( li ):
    newli=[]
    seen = set() # 이미 본 항목을 추적하기 위한 집합
    for item in li:
        if item not in seen: # 아직 보지 못한 항목이면
            seen.add( item ) # seen 집합에 추가
            newli.append(item) # 새 리스트에 추가
    return newli

li=[12,24,35,24,88,120,155,88,120,155]
print removeDuplicate(li)
```

내 솔루션: 파이썬 3 (주: 아래 첫 번째 솔루션은 `li.remove(i)` 가 루프 중 리스트를 변경하여 일부 중복이 제대로 제거되지 않을 수 있습니다. 예를 들어 `[1, 2, 2, 2]` 에서 첫 번째 `2` 가 제거되면 다음 `2` 는 건너뛰게 됩니다. 저자의 솔루션이나 아래 두 번째 솔루션이 더 정확합니다.)

```python
li = [12,24,35,24,88,120,155,88,120,155]
# 이 방법은 리스트를 반복하면서 제거할 때 문제를 일으킬 수 있습니다.
# for i in li:
#     if li.count(i) > 1:
#         li.remove(i) # 첫 번째 발생만 제거하고, 리스트 크기가 변경되어 반복에 영향
# print(li)

# 순서를 유지하면서 중복을 제거하는 올바른 방법 중 하나:
result = []
seen = set()
for item in li:
    if item not in seen:
        result.append(item)
        seen.add(item)
print(result)
```

또는 (제너레이터 사용, 순서 유지)

```python
def removeDuplicate( li ):
    seen = {}  # 사전을 사용하여 순서 유지 (Python 3.7+에서는 기본적으로 삽입 순서 유지, 이전 버전에서는 OrderedDict 사용 가능)
               # 여기서는 단순히 'in' 연산의 효율성을 위해 사용 (set과 유사)
    for item in li:
        if item not in seen: # 아직 보지 못한 항목이면
            seen[item] = True # 봤다고 표시
            yield item # 제너레이터로 반환

li = [12, 24, 35, 24, 88, 120, 155, 88, 120, 155]
ans = list(removeDuplicate(li)) # 제너레이터 결과를 리스트로 변환
print(ans)
```

문제 89

문제

> ***Person 클래스와 그 두 하위 클래스인 Male과 Female을 정의하십시오. 모든 클래스에는 "getGender" 메소드가 있으며, Male 클래스는 "Male"을, Female 클래스는 "Female"을 인쇄(반환)할 수 있어야 합니다.*** (주: 인쇄가 아니라 값을 반환하도록 수정하는 것이 일반적입니다.)

힌트

> ***하위 클래스를 정의하려면 Subclass(Parentclass)를 사용하십시오.***

솔루션: (Python 2 기준, Python 3에서도 유사하게 동작)

```
class Person(object): # Python 2에서는 object 명시적 상속
    def getGender( self ):
        return "Unknown" # 기본 성별

class Male( Person ): # Person 클래스 상속
    def getGender( self ):
        return "Male" # Male 클래스의 getGender 메소드 재정의

class Female( Person ): # Person 클래스 상속
    def getGender( self ):
        return "Female" # Female 클래스의 getGender 메소드 재정의

aMale = Male()
aFemale= Female()
print aMale.getGender()
print aFemale.getGender()
```

```
'''popomaticbubble 솔루션:
'''
class Person(object):
    def __init__(self): # 각 인스턴스가 gender 속성을 갖도록 함
        self.gender = "unknown"

    def getGender(self): # 인스턴스의 gender 속성을 인쇄
        print(self.gender)

class Male(Person):
    def __init__(self):
        super(Male, self).__init__() # 부모 클래스의 __init__ 호출 (선
택 사항, 여기서는 Person의 __init__이 gender를 unknown으로 설정)
        self.gender = "Male" # Male 인스턴스의 gender를 "Male"로 설정

class Female(Person):
    def __init__(self):
        super(Female, self).__init__() # 부모 클래스의 __init__ 호출
        self.gender = "Female" # Female 인스턴스의 gender를 "Female"로
설정

sharon = Female()
doug = Male()
sharon.getGender()
doug.getGender()
```

이전 날로 가기

다음 날로 가기

토론

문제 90

문제

> ***콘솔에서 입력받은 문자열의 각 문자 수를 세어 인쇄하는 프로그램을 작성하십시오.***

> ***예시: 다음 문자열이 프로그램에 입력으로 주어지면:***

```
abcdefgabc
```

> ***그러면 프로그램의 출력은 다음과 같아야 합니다:*** (주: 출력 순서는 다를 수 있습니다)

```
a,2
c,2
b,2
e,1
d,1
g,1
f,1
```

힌트

> ***키/값 쌍을 저장하려면 dict를 사용하십시오. 기본값으로 키를 조회하려면 dict.get() 메소드를 사용하십시오.***

주요 저자 솔루션: 파이썬 2

```
dic = {} # 빈 딕셔너리 생성
s=raw_input() # 사용자로부터 문자열 입력
for char_s in s: # 문자열 s의 각 문자에 대해 반복 (변수명을 s에서 char_s로 변경하여 명확성 증대)
    dic[char_s] = dic.get(char_s,0)+1 # char_s를 키로 하여, dic에 이미 있으면 값에 1을 더하고, 없으면 기본값 0에 1을 더하여 저장
# dic.items()는 (키, 값) 쌍의 리스트를 반환합니다.
# 각 (키, 값) 쌍에 대해 '%s,%s' 형식으로 문자열을 만들고, 이 문자열들을 개행 문자('\n')로 연결하여 출력합니다.
print '\n'.join(['%s,%s' % (k, v) for k, v in dic.items()])
```

내 솔루션: 파이썬 3 (알파벳 소문자만 카운트)

```
import string

s = input()
for letter in string.ascii_lowercase: # string.ascii_lowercase는 'abcdefghijklmnopqrstuvwxyz'
    cnt = s.count(letter) # 문자열 s에서 letter의 등장 횟수를 셈
    if cnt > 0: # 등장 횟수가 0보다 크면
        print("{},{}".format(letter,cnt)) # "문자,횟수" 형식으로 출력
```

또는 (알파벳 소문자만 카운트, ord/chr 사용)

```
s = input()
for letter_code in range(ord('a'),ord('z')+1):    # ord()는 문자의 ASCII(유니코드) 값을 반환합니다. 'a'부터 'z'까지 반복
    letter = chr(letter_code)                     # chr()은 ASCII(유니코드) 값을 문자로 변환합니다.
    cnt = s.count(letter)
    if cnt > 0:
        print("{},{}".format(letter,cnt))
```

```
'''Utkarsh4697 솔루션: (입력 문자열의 모든 문자 카운트, 정렬된 순서로 출력)
'''
s = 'abcdefgabc' # 예제 입력
# set(s)는 문자열 s에 포함된 고유한 문자들의 집합을 만듭니다.
# sorted(set(s))는 이 고유한 문자들을 알파벳순으로 정렬합니다.
for i in sorted(set(s)):
    print(f'{i}, {s.count(i)}') # 각 문자 i와 그 빈도수를 f-string을 사용하여 출력
```

```
'''popomaticbubble 솔루션: (딕셔너리 사용, 모든 문자 카운트)
'''
def character_counter(text):
    characters_list = list(text) # 문자열을 문자 리스트로 변환
    char_count = {} # 빈 딕셔너리
    for x in characters_list:
        if x in char_count.keys(): # 문자가 이미 딕셔너리 키에 있으면
            char_count[x] += 1 # 카운트 증가
        else: # 없으면
            char_count[x] = 1 # 새로 추가하고 카운트를 1로 설정
    return char_count


def dict_viewer(dictionary): # 딕셔너리를 특정 형식으로 출력하는 함수
    for x, y in dictionary.items(): # 딕셔너리의 모든 (키, 값) 쌍에 대해 반복
        print(f"{x},{y}") # f-string을 사용하여 "키,값" 형식으로 출력


text = input("> ")
dict_viewer(character_counter(text))
```

문제 91

문제

콘솔에서 문자열을 입력받아 역순으로 인쇄하는 프로그램을 작성하십시오.

예시: 다음 문자열이 프로그램에 입력으로 주어지면:* (주: 예시 입력의 마지막 '*'는 오타로 보입니다.)

```
rise to vote sir
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
ris etov ot esir
```

힌트

리스트를 역순으로 반복하려면 list[::-1]을 사용하십시오. (주: 문자열에도 적용 가능)

주요 저자 솔루션: 파이썬 2

```
s=raw_input()
s = s[::-1] # 문자열 s를 처음부터 끝까지 -1 간격으로 슬라이싱 (즉, 역순)
print s
```

내 솔루션: 파이썬 3

```
s = input()
s = ''.join(reversed(s)) # reversed(s)는 문자열 s의 문자들을 역순으로 가리키는 이터레이터 반환
                         # ''.join()은 이 이터레이터의 문자들을 빈 문자열을 구분자로 하여 합쳐 새 문자열 생성
print(s)
```

문제 92

문제

> ***콘솔에서 문자열을 입력받아 짝수 인덱스의 문자들을 인쇄하는 프로그램을 작성하십시오.***

> ***예시: 다음 문자열이 프로그램에 입력으로 주어지면:***

```
H1e2l3l4o5w6o7r8l9d
```

> ***그러면 프로그램의 출력은 다음과 같아야 합니다:***

```
Helloworld
```

힌트

> ***리스트를 2단계씩 건너뛰며 반복하려면 list[::2]를 사용하십시오.*** (주: 문자열에도 적용 가능)

주요 저자 솔루션: 파이썬 2

```
s=raw_input()
s = s[::2] # 문자열 s를 처음부터 끝까지 2칸씩 건너뛰며 슬라이싱 (즉, 짝수 인덱스 문자들)
print s
```

내 솔루션: 파이썬 3

```
s = "H1e2l3l4o5w6o7r8l9d" # 예제 입력
# range(len(s))로 0부터 문자열 길이 -1까지 인덱스 생성
# 인덱스 i가 짝수이면 s[i] (해당 인덱스의 문자)를 리스트에 포함
s_filtered = [ s[i] for i in range(len(s)) if i%2 ==0 ]
print(''.join(s_filtered)) # 리스트의 문자들을 합쳐 문자열로 출력
```

또는

```
s = "H1e2l3l4o5w6o7r8l9d" # 예제 입력
ns ='' # 새 문자열 초기화
for i in range(len(s)):
    if i % 2 == 0: # 인덱스 i가 짝수이면
        ns+=s[i] # 해당 문자를 ns에 추가
print(ns)
```

문제 93

문제

[1,2,3]의 모든 순열을 인쇄하는 프로그램을 작성하십시오.

힌트

리스트의 순열을 얻으려면 itertools.permutations()를 사용하십시오.

솔루션: (Python 2 기준, Python 3에서는 list()로 감싸야 함)

```
import itertools
# itertools.permutations([1,2,3])는 [1,2,3]의 모든 순열을 포함하는 이터레이터 반환
# Python 2에서는 print가 이터레이터의 내용을 바로 출력하려고 시도할 수 있으나,
# 명시적으로 list()로 변환하는 것이 일반적입니다.
print list(itertools.permutations([1,2,3])) # Python 3에서는 print(list(itertools.permutations([1,2,3])))
```

```
"""popomaticbubble 솔루션:
"""
from itertools import permutations # itertools에서 permutations 함수 가져오기

def permuation_generator(iterable):
    p = permutations(iterable) # 주어진 iterable의 모든 순열을 생성하는 이터레이터 p
    for i in p: # 각 순열 i에 대해 반복
        print(i) # 순열 (튜플 형태) 출력


x = [1,2,3]
permuation_generator(x)
```

문제 94

문제

> ***고전적인 중국 고대 퍼즐을 푸는 프로그램을 작성하십시오: 농장에 있는 닭과 토끼의 머리 수를 세니 35개이고 다리 수를 세니 94개입니다. 토끼와 닭은 각각 몇 마리입니까?***

힌트

> ***모든 가능한 해를 반복하려면 for 루프를 사용하십시오.***

솔루션:

```
def solve(numheads,numlegs):
    ns='해답 없음!' # 'No solutions!'를 번역
    for i in range(numheads+1): # i는 닭의 수 (0부터 numheads까지)
        j=numheads-i # j는 토끼의 수 (총 머리 수 - 닭의 수)
        if 2*i+4*j==numlegs: # 닭의 다리 수(2*i) + 토끼의 다리 수(4*j)
== 총 다리 수
            return i,j # 닭의 수와 토끼의 수 반환
    return ns,ns # 해답을 찾지 못하면 ns, ns 반환

numheads = 35
numlegs = 94
solutions=solve(numheads,numlegs)
print(solutions) # Python 3에서는 print() 사용
# 결과: (23, 12) -> 닭 23마리, 토끼 12마리
```

```python
"""popomaticbubble 솔루션: (itertools 사용)
"""
import itertools

def animal_counter(lst): # 다리 수 리스트를 받아 닭과 토끼 수 계산
    chickens = 0
    rabbits = 0
    for i in lst: # 리스트의 각 요소 (다리 수)
        if i == 2: # 다리가 2개면 닭
            chickens += 1
        elif i == 4: # 다리가 4개면 토끼
            rabbits += 1
    print(f"닭의 수: {chickens}\n토끼의 수: {rabbits}") # "Number of
chickens is" 등을 번역


def animal_calculator(total_legs, total_heads,
legs_of_each_species):
    # combinations_with_replacement는 주어진 종의 다리 수 리스트에서
total_heads 만큼 중복을 허용하여 조합을 만듭니다.
    # 즉, 각 동물이 어떤 종인지 모든 가능한 경우의 수를 만듭니다.
    combinations =
itertools.combinations_with_replacement(legs_of_each_species,
total_heads)
    correct_combos = []
    for i in list(combinations): # 각 조합 (total_heads 만큼의 다리 수
튜플)
        if sum(i) == total_legs: # 조합된 다리 수의 합이 총 다리 수와 같
으면
            correct_combos.append(i) # 올바른 조합 리스트에 추가
    print("가능한 다리 조합:", correct_combos) # "Correct combos" 번역
    for i in correct_combos:
        animal_counter(i) # 각 올바른 조합에 대해 동물 수 계산 및 출력

animal_calculator(94, 35, legs_of_each_species=[2,4])
```

이전 날로 가기

다음 날로 가기

토론

이 페이지부터 저장소의 확장된 부분이 시작됩니다. 이전 94개의 문제는 소개에 언급된 저장소에서 수집했습니다. 다음 문제들은 Hackerrank 및 인터넷의 다른 자료들에서 수집했으며, 모든 제공된 솔루션은 파이썬 3 기준입니다.

문제 95

문제

대학교 운동회의 참가자 점수표가 주어졌을 때, 준우승자의 점수를 찾아야 합니다. 점수들이 주어지면 리스트에 저장하고 준우승자의 점수를 찾으십시오.

다음 문자열이 프로그램에 입력으로 주어지면:

```
5
2 3 6 6 5
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
5
```

힌트

> ***점수들을 고유하게 만든 다음 두 번째로 좋은 숫자를 찾으십시오.***

내 솔루션: 파이썬 3

```
n = int(input()) # 첫 번째 입력은 사용되지 않지만, 문제 형식상 입력받음
arr = map(int, input().split()) # 두 번째 줄의 점수들을 공백 기준으로 나누어 정수형으로 변환
arr = list(set(arr)) # set으로 변환하여 중복 제거 후 다시 리스트로 변환
arr.sort() # 오름차순 정렬
print(arr[-2]) # 뒤에서 두 번째 요소 (즉, 준우승자 점수) 출력
```

```
'''
mishrasunny-coder 솔루션:
'''
num = int(input("숫자 입력: ")) # "Enter num: " 번역
L = []

while True:
    L.append(num)
    num = int(input("다른 숫자 입력 (0 입력 시 종료): ")) # "Enter another: " 번역 및 종료 조건 명시
    if num == 0: # 0을 입력하면 루프 종료 (임의의 종료 조건)
        break

L1 = list(set(L[:])) # 리스트 L의 복사본으로 set을 만들어 중복 제거 후 다시 리스트로
L2 = sorted(L1) # 정렬
print(L2)

print(f'준우승자는 {L2[-2]}입니다') # "The runner up is" 번역
```

```
'''KailashS3 솔루션:
'''
num_count = int(input()) # 변수명을 num에서 num_count로 변경하여 명확성 증대
scores = list(map(int, input().split(' ')))
winner = max(scores) # 최고 점수
lst = [] # 준우승자 후보 리스트

if len(scores) != num_count: # 입력된 점수의 개수와 첫 줄의 숫자 (num_count)가 다르면
    print('점수의 길이가 주어진 입력과 다릅니다') # "length of score is greater than input given" 번역
else:
    for score in scores:
        if winner > score: # 최고 점수보다 작은 점수들만
            lst.append(score) # 준우승자 후보 리스트에 추가

runnerup = max(lst) # 후보 중에서 최댓값 (즉, 준우승자 점수)
print(runnerup)
```

문제 96

문제

문자열 S와 너비 W가 주어집니다. 주어진 너비로 문자열을 단락으로 줄 바꿈하는 것이 당신의 과제입니다.

다음 문자열이 프로그램에 입력으로 주어지면:

```
ABCDEFGHIJKLIMNOQRSTUVWXYZ
4
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
ABCD
EFGH
IJKL
IMNO
QRST
UVWX
YZ
```

힌트

---

textwrap 모듈의 wrap 함수를 사용하십시오.

내 솔루션: 파이썬 3

```
import textwrap

def wrap(string, max_width):
    # textwrap.wrap()은 문자열을 주어진 너비로 줄 바꿈한 문자열 리스트를 반환합니다.
    string_wrapped = textwrap.wrap(string,max_width)
    # "\n".join()은 리스트의 각 문자열 사이에 개행 문자를 넣어 하나의 문자열로 합칩니다.
    string_joined = "\n".join(string_wrapped)
    return string_joined

if __name__ == '__main__':
    string, max_width = input(), int(input()) # 문자열과 너비를 입력받음
    result = wrap(string, max_width)
    print(result)
```

```
"""mishrasunny-coder 솔루션:
"""
import textwrap

string = input()
width = int(input())

# textwrap.fill()은 문자열을 주어진 너비로 줄 바꿈한 후, 전체를 하나의 문자열로 반환합니다 (개행문자 포함).
print(textwrap.fill(string,width))
```

```
"""Prashanth 솔루션:
"""
from textwrap import wrap # textwrap 모듈에서 wrap 함수만 가져옴
x = str(input(': '))
w = int(input())
z = list(wrap(x, w)) # wrap 함수는 리스트를 반환하므로 list()로 다시 감쌀 필요 없음
for i in z:
    print(i)
```

```
"""saxenaharsh24 솔루션:
"""
import textwrap
string = input('')
# textwrap.wrap()의 결과를 바로 '\n'.join()으로 처리
print('\n'.join(textwrap.wrap(string, width= int(input('')))))
```

```
"""popomaticbubble 솔루션: (itertools 사용)
"""
import itertools
string = input("> ")
width_length = int(input("그룹화 너비는 얼마입니까? ")) # "What is the
width of the groupings? " 번역

def grouper(string, width):
    iters = [iter(string)] * width # 문자열의 이터레이터를 너비만큼 복제한
리스트
    # zip_longest는 여러 이터레이블의 요소들을 묶어 튜플로 반환합니다. 짧은
이터레이블은 fillvalue로 채웁니다.
    return itertools.zip_longest(*iters, fillvalue='')

def displayer(groups):
    for x in groups: # 각 그룹 (튜플)
        if x == '': # fillvalue로 채워진 빈 그룹은 건너뜀
            continue
        else:
            print(''.join(x)) # 튜플의 문자들을 합쳐 출력

displayer(grouper(string, width_length))
```

문제 97

문제

> ***정수 N이 주어집니다. 크기 N의 알파벳 랑골리를 인쇄하는 것이 당신의 과제입니다. (랑골리는 패턴 생성에 기반한 인도 민속 예술의 한 형태입니다.)***

> ***다양한 크기의 알파벳 랑골리가 아래에 나와 있습니다:***

```
#크기 3

----c----
--c-b-c--
c-b-a-b-c
--c-b-c--
----c----

#크기 5

--------e--------
------e-d-e------
----e-d-c-d-e----
--e-d-c-b-c-d-e--
e-d-c-b-a-b-c-d-e
--e-d-c-b-c-d-e--
----e-d-c-d-e----
------e-d-e------
--------e--------
```

힌트

> *주어진 방식으로 랑골리의 절반을 먼저 인쇄하고 각 줄을 리스트에 저장하십시오. 그런 다음 나머지 부분을 얻으려면 리스트를 역순으로 인쇄하십시오.*

내 솔루션: 파이썬 3

```
import string
def print_rangoli(size):
    n = size
    alph = string.ascii_lowercase # 'abcdefghijklmnopqrstuvwxyz'
    width = 4 * n - 3 # 전체 너비 계산

    ans = [] # 각 줄을 저장할 리스트
    for i in range(n): # 위쪽 절반 생성
        # 왼쪽 부분 생성 (예: n=3, i=0 -> 'c'; i=1 -> 'c-b'; i=2 ->
'c-b-a')
        left_chars = alph[n - 1 - i : n] # alph[2:3] -> 'c';
alph[1:3] -> 'cb'; alph[0:3] -> 'abc' (i에 따라 역순으로 접근)
                                         # 수정: alph[n-1]부터 i+1개 문
자를 역순으로 가져와야 함.
                                         # 예: n=3. i=0: 'c'. i=1:
'c-b'. i=2: 'c-b-a'
                                         # 올바른 접근: alph[n-1-i : n]
-> alph[i] 부터 n-1까지 역순으로
                                         # s = '-'.join(alph[j] for j
in range(n-1, n-1-i-1, -1)) 와 같은 형태가 되어야 함
                                         # 또는 문제의 의도대로 alph[n-
1]부터 시작하여 점점 확장되는 형태

        # 현재 코드의 left 생성 방식:
        # n=3, i=0: alph[2:3] -> 'c'
        # n=3, i=1: alph[1:3] -> 'c-b' (alph[1]=b, alph[2]=c)
        # n=3, i=2: alph[0:3] -> 'c-b-a' (alph[0]=a, alph[1]=b,
alph[2]=c)
        # 위 방식은 alph[n-i-1:n] 이므로, (n-i-1) 인덱스부터 (n-1) 인덱스
까지의 문자들을 사용합니다.
        # 예: n=3, i=0: left_part = alph[2:3] -> ('c'). join -> 'c'
        # 예: n=3, i=1: left_part = alph[1:3] -> ('b','c'). join ->
'b-c'
        # 예: n=3, i=2: left_part = alph[0:3] -> ('a','b','c'). join
-> 'a-b-c'
        # 이것을 뒤집어서 사용해야 예제와 맞음.
        # 올바른 문자열 구성:
        line_chars = [alph[n-1-j] for j in range(i+1)] # 예: i=0 ->
[c]; i=1 -> [c,b]; i=2 -> [c,b,a]
        # 오른쪽 부분 추가
        current_line_alphabets = line_chars + line_chars[-2::-1] #
예: i=1 -> [c,b] + [c] -> [c,b,c]
                                                                #
예: i=2 -> [c,b,a] + [b,c] -> [c,b,a,b,c]

        mid_pattern = '-'.join(current_line_alphabets)
```

```
        final = mid_pattern.center(width, '-') # 중앙 정렬 및 '-' 채우
기
        ans.append(final)

    # 아래쪽 절반은 위쪽 절반(ans 리스트)을 뒤집어서 추가 (마지막 줄 제외)
    if len(ans) > 1: # n > 1 인 경우에만
        for i in range(n - 2, -1, -1): # ans의 (n-2)번째 인덱스부터 0번
째 인덱스까지 역순으로
            ans.append(ans[i])

    ans_str = '\n'.join(ans) # 리스트의 각 줄을 개행문자로 연결
    print(ans_str)


if __name__ == '__main__':
    n_input = int(input()) # 변수명을 n에서 n_input으로 변경 (함수 내의 n
과 구분)
    print_rangoli(n_input)
```

```
'''suggula jaswanth 솔루션:
'''
def rangoli(n):
    l1=list(map(chr,range(97,123))) # 알파벳 소문자 리스트 ('a'부터
'z'까지)
    # x는 중앙 라인의 문자 패턴 (예: n=3 -> ['c','b','a','b','c'])
    x=l1[n-1::-1]+l1[1:n] # l1[n-1::-1]은 n번째 문자부터 'a'까지 역순,
l1[1:n]은 'b'부터 n번째 문자까지
                          # 예: n=3 -> l1[2::-1] = ['c','b','a'],
l1[1:3]=['b','c'] => ['c','b','a','b','c']
    mid_len=len('-'.join(x)) # 중앙 라인의 전체 길이 (하이픈 포함)

    # 위쪽 절반 (중앙 라인 제외)
    for i in range(1,n):
        # l1[n-1:n-i:-1] : n번째 문자부터 (n-i+1)번째 문자까지 역순 (예:
i=1, n=3 -> l1[2:2:-1] -> []) -> 이부분 수정 필요
        # l1[n-i:n] : (n-i+1)번째 문자부터 n번째 문자까지
        # 의도한 바: i가 증가함에 따라 중앙에서 멀어지는 문자들 추가
        # 예: n=3. i=1 (첫째 줄): c. i=2 (둘째 줄): c-b-c
        # 수정된 패턴:
        chars_for_line = l1[n-1 : n-1-i : -1] + l1[n-1-i : n] # n-1
부터 시작, i개만큼, 그리고 다시 돌아오기
        # 더 정확한 패턴:
        part1 = l1[n-1 : n-i-1 : -1] # 바깥쪽에서 안쪽으로 (예: n=3,
i=1 -> [c])
        part2 = l1[n-i : n]      # 안쪽에서 바깥쪽으로 (예: n=3, i=1 ->
[c])
                                 # 합치면 [c,c]가 됨.
        # 올바른 접근:
        current_chars = l1[n-1:n-1-i:-1] + l1[n-i:n] if i>0 else
[l1[n-1]] # suggula jaswanth의 로직은 복잡하여,

# 아래와 같이 일반적인 랑골리 패턴으로 수정
        temp_list = l1[n-1:n-i-1:-1] # 예: n=3, i=1 -> [c]. i=2 ->
[c,b]
        line_str = '-'.join(temp_list + l1[n-i:n])
        print(line_str.center(mid_len,'-'))

    # 아래쪽 절반 (중앙 라인 포함)
    for i in range(n,0,-1): # n부터 1까지 감소
        temp_list = l1[n-1:n-i-1:-1]
        line_str = '-'.join(temp_list + l1[n-i:n])
        print(line_str.center(mid_len,'-'))

# 위 suggula jaswanth 솔루션은 패턴 구성에 수정이 필요해 보입니다.
```

```
# 원본 My Solution을 참고하는 것이 좋습니다.
# rangoli(5) # 함수 호출 예시
```

문제 98

문제

> ***날짜가 주어집니다. 해당 날짜의 요일을 찾는 것이 당신의 과제입니다.***

입력

> ***한 줄의 입력으로 월, 일, 연도가 각각 공백으로 구분되어 MM DD YYYY 형식으로 주어집니다.***
>
> ```
> 08 05 2015
> ```

출력

> ***정확한 요일을 대문자로 출력하십시오.***
>
> ```
> WEDNESDAY
> ```

힌트

> ***calender 모듈의 weekday 함수를 사용하십시오.*** (주: calendar 모듈)

솔루션:

```
import calendar

month, day, year = map(int, input().split()) # 입력받은 문자열을 공백 기준으로 나누어 각각 정수형으로 변환

dayId = calendar.weekday(year, month, day) # calendar.weekday()는 요일을 정수 형태로 반환 (월요일=0, 일요일=6)
print(calendar.day_name[dayId].upper()) # calendar.day_name은 요일 이름의 리스트. 해당 인덱스의 요일 이름을 가져와 대문자로 변경하여 출력
```

문제 99

문제

두 정수 집합 M과 N이 주어졌을 때, 그들의 대칭 차집합을 오름차순으로 인쇄하십시오. 대칭 차집합이라는 용어는 M 또는 N에 존재하지만 둘 다에는 존재하지 않는 값들을 나타냅니다.

입력

첫 번째 줄에는 정수 M이 입력됩니다. 두 번째 줄에는 M개의 공백으로 구분된 정수가 입력됩니다. 세 번째 줄에는 정수 N이 입력됩니다. 네 번째 줄에는 N개의 공백으로 구분된 정수가 입력됩니다.

```
4
2 4 5 9
4
2 4 11 12
```

출력

대칭 차집합 정수들을 오름차순으로, 한 줄에 하나씩 출력하십시오.

```
5
9
11
12
```

힌트

대칭 차집합 연산을 수행하려면 '^'를 사용하십시오.

솔루션:

```
if __name__ == '__main__':
    n_count = int(input()) # 변수명 n을 n_count로 변경 (아래 m과 구분)
    set1 = set(map(int,input().split())) # 첫 번째 집합 입력

    m_count = int(input()) # 변수명 m을 m_count로 변경
    set2 = set(map(int, input().split())) # 두 번째 집합 입력

    ans = list(set1 ^ set2) # '^' 연산자로 대칭 차집합 계산 후 리스트로
변환
    ans.sort() # 오름차순 정렬
    for i in ans:
        print(i) # 각 요소를 한 줄에 하나씩 출력
```

이전 날로 가기

다음 날로 가기

토론

문제 100

문제

단어들이 주어집니다. 일부 단어는 반복될 수 있습니다. 각 단어에 대해 발생 횟수를 출력하십시오. 출력 순서는 단어가 입력에 나타난 순서와 일치해야 합니다. 명확성을 위해 샘플 입출력을 참조하십시오.

다음 문자열이 프로그램에 입력으로 주어지면:

```
4
bcdef
abcdefg
bcde
bcdef
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
3
2 1 1
```

힌트

입력 순서를 얻기 위해 리스트를 만들고 단어 빈도를 계산하기 위해 사전을 만드십시오.

내 솔루션: 파이썬 3

```python
n = int(input()) # 입력될 단어의 총 개수

word_list = [] # 입력된 단어의 순서를 유지하기 위한 리스트
word_dict = {} # 각 단어의 빈도수를 저장하기 위한 딕셔너리

for i in range(n):
    word = input()
    if word not in word_dict: # 단어가 딕셔너리에 처음 등장하면
        word_list.append(word) # 순서 리스트에 추가
    word_dict[word] = word_dict.get(word, 0) + 1 # 해당 단어의 빈도수 증가

print(len(word_list)) # 고유한 단어의 개수 출력
for word in word_list: # 입력된 순서대로 각 단어에 대해
    print(word_dict[word], end=' ') # 해당 단어의 빈도수 출력 (공백으로 구분)
```

문제 101

문제

문자열이 주어집니다. 문자열의 문자 빈도를 계산하고 빈도수의 내림차순으로 문자를 인쇄하는 것이 당신의 과제입니다. (주: 빈도수가 같을 경우 문자 순서에 대한 언급이 없으므로, 일반적인 정렬 순서(예: 알파벳 순)를 따르거나 입력 순서를 고려할 수 있습니다. 아래 솔루션은 빈도수 내림차순 후 문자 오름차순으로 정렬합니다.)

다음 문자열이 프로그램에 입력으로 주어지면:

```
aabbbccde
```

그러면 프로그램의 출력은 다음과 같아야 합니다:

```
b 3
a 2
c 2
d 1
e 1
```

힌트

> *딕셔너리로 빈도를 계산하고 딕셔너리 아이템에서 값으로 정렬하십시오.*

내 솔루션: 파이썬 3

```
word = input()
dct = {} # 문자 빈도를 저장할 딕셔너리
for i in word: # 문자열의 각 문자에 대해
    dct[i] = dct.get(i,0) + 1 # 해당 문자의 빈도수 증가

# dct.items()는 (키, 값) 쌍의 리스트를 반환합니다.
# sorted() 함수의 key 인자에 람다 함수를 사용하여 정렬 기준을 지정합니다.
# -x[1]은 빈도수(값)의 내림차순으로 정렬하고,
# x[0]은 빈도수가 같을 경우 문자(키)의 오름차순(알파벳 순)으로 정렬합니다.
dct_sorted = sorted(dct.items(),key=lambda x: (-x[1],x[0]))
for i in dct_sorted:
    print(i[0],i[1]) # 문자(키)와 빈도수(값) 출력
```

```
'''yuan1z 솔루션:'''

X = input()
my_set = set(X) # 문자열에 포함된 고유한 문자 집합
arr = []
for item in my_set:
    arr.append([item,X.count(item)]) # 각 고유 문자와 해당 빈도수를 리스트로 만들어 arr에 추가
# arr를 정렬합니다. 정렬 기준은 위와 동일 (빈도수 내림차순, 문자 오름차순)
tmp = sorted(arr,key = lambda x: (-x[1],x[0]))

for i in tmp:
    print(i[0]+' '+str(i[1])) # 결과 출력
```

```
'''StartZer0 솔루션:'''

s = list(input()) # 입력 문자열을 문자 리스트로 변환

# 각 문자를 키로, 해당 문자의 빈도수를 값으로 하는 딕셔너리 생성 (리스트 컴프리헨션 사용)
dict_count_ = {k:s.count(k) for k in s}
# 딕셔너리의 (키, 값) 쌍을 튜플 리스트로 변환
list_of_tuples = [(k,v) for k,v in dict_count_.items()]
# 튜플 리스트를 빈도수(x[1])의 내림차순으로 정렬
list_of_tuples.sort(key = lambda x: x[1], reverse = True)
# (주: 빈도수가 같을 경우 원래 순서 또는 예측 불가능한 순서가 될 수 있습니다.
#  안정적인 정렬을 원하면 추가 정렬 기준 필요)

for item in list_of_tuples:
  print(item[0], item[1])
```

문제 102

문제

문자열을 입력받아 숫자와 문자의 개수를 계산하는 파이썬 프로그램을 작성하

십시오.

입력

```
Hello321Bye360
```

출력

```
Digit - 6
Letter - 8
```

힌트

isdigit() 및 isalpha() 함수를 사용하십시오.

솔루션:

```
word = input()
digit_count = 0 # 변수명 digit을 digit_count로 변경 (가독성)
letter_count = 0 # 변수명 letter를 letter_count로 변경
for char in word:
    digit_count += char.isdigit() # char.isdigit()는 문자가 숫자이면
True(1), 아니면 False(0) 반환
    letter_count += char.isalpha() # char.isalpha()는 문자가 알파벳이면
True(1), 아니면 False(0) 반환

print('Digit -',digit_count)
print('Letter -',letter_count)
```

문제 103

문제

숫자 N이 주어집니다. 재귀를 사용하여 1부터 N까지의 합을 찾으십시오.

입력

```
5
```

출력

```
15
```

힌트

합계를 얻기 위해 재귀 함수를 만드십시오.

솔루션:

```
def rec(n): # 재귀 함수 정의
    if n == 0: # 기본 케이스: n이 0이면 0 반환
        return n
    return rec(n-1) + n # 재귀 단계: rec(n-1)의 결과에 n을 더하여 반환


n = int(input())
sum_val = rec(n) # 변수명 sum을 sum_val로 변경 (sum은 내장 함수와 충돌 방지)
print(sum_val)
```

```
"""popomaticbubble 솔루션: (꼬리 재귀 형태는 아님, current를 누적)
"""
def summer(counter, n, current_sum): # 변수명 current를 current_sum으로 변경
    if n == 0: # N이 0이면 합계는 0
        return 0
    if counter == n: # 카운터가 N에 도달하면
        return current_sum + n # 현재 합계에 N을 더하여 반환
    else: # 카운터가 N보다 작으면
        current_sum = current_sum + counter # 현재 합계에 카운터 값을 더함
        counter += 1 # 카운터 증가
        return summer(counter, n, current_sum) # 재귀 호출


N = int(input("> "))
print(summer(1, N, 0)) # counter=1, n=N, current_sum=0으로 시작
```

이전 날로 가기

토론

계속됩니다...