백준 문제 풀이 - 단계별로 풀어보기1

상태	완료 🙌
작성 일시	@2023년 2월 28일 오후 9:01
유형	**기본 알고리즘 문제들과 수학, 정수론, 기하, 확률, 기초 자료구조, 부루트 포스, 스택, 큐, 덱에 대한 백준 문제풀이 노트. (ver. 파이썬)**
텍스트	백준을 풀어보자!! https://www.acmicpc.net/user/yuyunjae03

백준 27522번 카트라이더 드리프트 https://www.acmicpc.net/problem/27522

```
import sys
a = []
point = [10, 8, 6, 5, 4, 3, 2, 1]
red = blue = lot = 0
for i in range(8):
    a.append(sys.stdin.readline().split())
    a[i][0] = a[i][0].replace(":", "")
    a[i][0] = int(a[i][0])
a.sort(key = lambda x : x[0])
for i in range(8):
    if a[i][1] == "R":
        red += point[lot]
    else:
        blue += point[lot]
    lot += 1
if red < blue:
    print("Blue")
else:
    print("Red")
```

백준 12865번 평범한 배낭 https://www.acmicpc.net/problem/12865

```
import sys
N, K = map(int, sys.stdin.readline().split())
WV = [0 for _ in range(K+1)]
for i in range(N):
```

```
    W, V = map(int, sys.stdin.readline().split())
    if W > K:
        continue
    else:
        wv = WV[:]
        if WV[W] < V:
            WV[W] = V
        for a in range(K):
            if (wv[a] != 0) and ((W+a) <= K) and ((V+wv[a]) > wv[W+a]):
                WV[W+a] = V+wv[a]
best = 0
for i in range(K+1):
    if best < WV[i]:
        best = WV[i]
print(best)
```

백준 4344번 평균은 넘겠지 https://www.acmicpc.net/problem/4344

문자열 출력방식과 %연산자를 이용
최근 문제가 수정이 됐다.. 파이썬의 경우 0.5 일때에도 반올림하면 1이 아닌 0이 나오는 경우가 발생(이진수로 빈 사값을 표현하다보니 그런 듯 하다)한다. 따라서 맨 아래 방식으로 수정한다..

```
#my code
import sys
C = int(input())
for i in range(C):
    N = list(map(int, sys.stdin.readline().split()))
    a = N[1:]
    average = sum(a)/N[0]
    count = 0
    for i in a:
        if average < i:
            count += 1
    answer = count/N[0]*100
    print("%0.3f" %answer, "%", sep = "")
```

```
#other good code
n = int(input())
for i in range(n):
    c = 0
    s = [int(j) for j in input().split()]
    avg=sum(s[1:])/s[0]
    for k in s[1:]:
        if avg < k:
            c+=1
        else: pass
    print(f"{(c/(len(s[1:]))*100):.3f}%")
```

```
#my fixed code
import sys
C = int(input())
for i in range(C):
    N = list(map(int, sys.stdin.readline().split()))
    a = N[1:]
    average = sum(a)/N[0]
    count = 0
    for j in a:
        if average < j:
            count += 1
    answer = count / N[0] * 100
    check = str(answer - int(answer)) + '000000'
```

```
    if check[5] >= '5':
        print("%0.3f" %round(answer+0.0001, 3), "%", sep="")
    else:
        print("%0.3f" %abs(round(answer-0.0001, 3)), "%", sep="")
#%0.3f를 이용해 소수점 3자리수까지 출력
#abs를 이용해 -0.0%일 경우, 0.0%으로 바꾸어줌.
```

,

백준 2941번 크로아티아 알파벳 https://www.acmicpc.net/problem/2941

파이썬의 .count()와 .replace()함수를 이용해 해결. c언어는 일일이 경우를 나누어야함.

```
#my code
s = input()
num = len(s)
for i in ['c=', 'c-', 'dz=', 'd-', 'lj', 'nj', 's=', 'z=']:
    num -= s.count(i)
print(num)
```

```
#other good code
# 크로아티아 알파벳

b = ['c=','c-','dz=','d-','lj','nj','s=','z=']
a = input()


for i in b:
    a = a.replace(i, '*')
print(len(a))
```

백준 1316번 그룹 단어 채커 https://www.acmicpc.net/problem/1316

```
#my code
N = int(input())
count = N
for i in range(N):
    alphabet = list("abcdefghijklmnopqrstuvwxyz")
    s = input()
    for j in range(len(s)-1):
        if s[j] != s[j+1]:
            if s[j] in alphabet:
                alphabet.remove(s[j])
            if s[j+1] not in alphabet:
                count -= 1
                break
print(count)
```

```
#other good code
#문자열 슬라이싱을 이용한 좋은 풀이
N = int(input())
cnt = N

for i in range(N):
    word = input()
    for j in range(0, len(word)-1):
        if word[j] == word[j+1]:
            pass
        elif word[j] in word[j+1:]:
```

```
            cnt -= 1
            break

print(cnt)
```

백준 25206번 너의 평점은 https://www.acmicpc.net/problem/25206

```
#my code
# def로 함수를 정의해서 사용해봄. if c == 'P': 이부분을 굳이 따로 안빼고 아래처럼 해도 됬을 듯.
import sys
sum = [0, 0]
score = ['D0', 'D+', 'C0', 'C+', 'B0', 'B+', 'A0', 'A+']
def point(p: float, grade: str):
    if grade == 'F':
        return 0
    if grade in score:
        return p*(score.index(grade)/2+1)

for _ in range(20):
    a, b, c = sys.stdin.readline().split()
    if c == 'P':
        continue
    sum[0] += point(float(b), c)
    sum[1] += float(b)
print(sum[0]/sum[1])
```

```
#other good code
#파이썬의 dictionary를 이용해 깔끔하게 해결
s = 0.0
cnt = 0.0
dic = {'A+':4.5, 'A0':4.0, 'B+':3.5, 'B0':3.0, 'C+':2.5, 'C0':2.0, 'D+':1.5, 'D0':1.0, 'F':0.0}
for i in range (20):
    sub, hak, rank = input().split()
    if rank != 'P':
        s += float(hak) * dic[rank]
        cnt += float(hak)
print(s/cnt)
```

```
//good c language code
#include <stdio.h>

int main()
{
    char n[51], g[3];
    float h, hsum=0, gsum=0;

    for(int i=0; i<20; i++)
    {
        scanf("%s %f %s", n, &h, g);

        if(g[0]!='P') hsum+=h;
        if(g[0]=='A') gsum+=h*4;
        else if(g[0]=='B') gsum+=h*3;
        else if(g[0]=='C') gsum+=h*2;
        else if(g[0]=='D') gsum+=h*1;
        if(g[1]=='+') gsum+=h*0.5;

        g[1]='0';
    }

    printf("%f\n", gsum/hsum);
```

```
    return 0;
}
```

백준 10798번 세로읽기 https://www.acmicpc.net/problem/10798

```
#my code
a = []
length = []
for i in range(5):
    a.append(list(input()))
    length.append(len(a[i]))
for i in range(max(length)):
    for j in range(5):
        if i >= length[j]:
            continue
        print(a[j][i], end ="")
```

```
#other good code
G = [input().rstrip() for _ in range(5)]
ans = ''
for i in range(15):
    for k in range(5):
        if len(G[k]) > i:
            ans += G[k][i]
print(ans)
```

백준 2563번 색종이 https://www.acmicpc.net/problem/2563

```
#my code
area = [[0]*100 for _ in range(100)]
num = int(input())
for i in range(num):
    x, y = map(int, input().split())
    for k in range(10):
        for l in range(10):
            area[x+k][y+l] = 1
sum = 0
for i in range(100):
    sum += area[i].count(1)
print(sum)
```

```
#other good code
#map의 첫 번째 인자에 자료형이 아닌 메소드를 넣을 수 있다는 것을 처음 알게 됨
n = int(input())
paper = [[0]*100 for _ in range(100)]
for i in range(n):
    x, y = map(int, input().split())
    for j in range(x,x+10):
        for k in range(y,y+10):
            paper[j][k] = 1
print(sum(map(sum,paper)))
```

백준 2292번 벌집 https://www.acmicpc.net/problem/2292

```
#my code
#while 문을 생각하지 못하다니...
N = int(input())
num = 1
for i in range(30000):
    num += 6*i
    if N <= num:
        print(i+1)
        break
```

```
#other good code
N = int(input())
num = 1
count = 1
while N > num:
    num += 6*count
    count += 1
print(count)
```

백준 1193번 분수찾기 https://www.acmicpc.net/problem/1193

```
#my code
#아래 방법대로 했다면 count가 필요가 없었다..
X = int(input())
count = 1
i = 1
while X > i:
    count += 1
    i += count
left = i - X + 1
if count % 2:
    print(left, '/', count - left + 1, sep='')
else:
    print(count - left + 1, '/', left, sep='')
```

```
#other good code
#f-strings 방법 연습하기!
n = int(input())
i = 1
while n>i:
    n-=i
    i+=1
if i%2==0:
    print(f"{n}/{i-n+1}")
else:
    print(f"{i-n+1}/{n}")
```

백준 2775번 부녀회장이 될테야 https://www.acmicpc.net/problem/2775

쉬운 문제인데 너무 구상하는데 오래걸림.. 머리로 생각이 잘 안되면 그려볼 것.

```
#my code
T = int(input())
for _ in range(T):
    k = int(input())
    n = int(input())
```

```
    if k == 0:
        print(n)
    count = [[0]*n for _ in range(k+1)]
    for i in range(n):
        count[0][i] = i+1
    for i in range(k+1):
        count[i][0] = count[0][0]
    for i in range(1, k+1):
        for j in range(1, n):
            count[i][j] = count[i-1][j] + count[i][j-1]
    print(count[k][n-1])
```

```
#other good code
import sys
input = sys.stdin.readline

for _ in range(int(input())):
    k = int(input())
    n = int(input())
    arr = [[0]*15 for _ in range(15)]
    for i in range(1, 15):
        arr[0][i] = i

    r, c = 1, 1
    while True:
        arr[r][c] = sum(arr[r-1][1:c+1])
        if r == k and c == n:
            print(arr[k][n])
            break
        c += 1
        if c >= 15:
            r += 1
            c = 1
```

백준 10250번 ACM 호텔 https://www.acmicpc.net/problem/10250

```
#my code
import sys
T = int(input())
for _ in range(T):
    H, W, N = map(int, sys.stdin.readline().split()) #H층, W각 층의 방의 수, N몇 번째 손님
    room = 1
    while N > H:
        N -= H
        room += 1
    if room > 9:
        print(f'{N}{room}')
    else:
        print(f'{N}0{room}')
```

```
#other good code
#굳이 반복문을 사용하지 않아도 근과 나머지를 구하면 풀리는 문제이기에 //와 % 를 이용한 모습
import sys
input = sys.stdin.readline


t = int(input())
for _ in range(t):
    h, w, n = map(int, input().split())
    y = (n - 1) % h + 1
    x = (n - 1) // h + 1
```

```
    print(y * 100 + x)
```

백준 2869번 달팽이는 올라가고 싶다 https://www.acmicpc.net/problem/2869
수학관련 문제들은 기초적인 공식들을 먼저 생각하고 구현을 해야겠다.
(time-1)(up-down) + up ≥ height 이것만 빨리 생각했다면,
1) time ≥ (height-up)/(up-down) + 1
2) time ≥ (height-down)/(up/down)
둘중 하나로 바로 구했을 듯…

```
#my code
A, B, V = map(int, input().split())
time = 1
if A == V:
    print(time)
else:
    time += (V-A) // (A-B)
    left = (V-A) % (A-B)
    if left > 0:
        time += 1
    print(time)
```

```
#other good code
a, b, v=map(int, input().split())

h=(v-b)/(a-b)
if h==int(h):
    print(int(h))
else:
    print(int(h)+1)
```

백준 1978번 소수찾기 https://www.acmicpc.net/problem/1978
에라토스테네스의 체를 이용한 알고리즘 공부하기
#i**0.5 +1 하는 이유는 솔직히 모르겠음. i//2 +1 이 맞지않나...
#-> 약수의 특징을 이용한것이었음. ex) 80-> 1*80, 2*40, 4*20, 5*16, 8*10 결국 둘중 하나는 루트80+1보다 작게 되어있음.

```
#my code
import sys
N = int(input())
a = list(map(int, sys.stdin.readline().split()))
prime = [2]
for i in range(3, 1001):
    count = 0
    for j in range(2, i):
        if i % j == 0:
            count = 1
            break
    if count == 0:
        prime.append(i)
num = 0
for i in a:
    if i in prime:
        num += 1
print(num)
```

```
#other good code
n = int(input())
nums = list(map(int,input().split()))
cnt = 0

for i in nums:
    if i == 1:
        cnt += 1
        continue
    for j in range(2, int(i**0.5) + 1):
        if i % j == 0:
            cnt += 1
            break

print(n - cnt)
```

백준 1157번 단어공부 https://www.acmicpc.net/problem/1157

```
#my code
import sys
a = sys.stdin.readline()
a = a.upper()
num = []
alphabet = []
for i in range(65, 91):
    alphabet.append(chr(i))
for i in alphabet:
    num.append(a.count(i))
a = max(num)
cou = 0
key = 0
for i in range(len(num)):
    if num[i] == a:
        cou += 1
        key = i
if cou > 1:
    print('?')
else:
    print(chr(key+65))
```

```
#other good code
word = input().lower()
lst = [word.count(chr(i)) for i in range(97, 123)] #lst를 소문자 a~z의 사용 숫자를 담은 list로 만듦
if lst.count(max(lst)) == 1:                #나도 count(max(lst))를 이용했으면 7줄은 줄일 수 있었을 듯.
    print(chr(lst.index(max(lst)) + 65))
else:
    print("?")
```

백준 2581번 소수 https://www.acmicpc.net/problem/2581

에라토스테네스의 체

eratosthenes()라는 함수로 각 소수들의 배수 삭제.. range(i*i, b+1,i)이건 정말 효율적인듯..

```
#my code
M = int(input())
N = int(input())
sum_prime = 0
min_prime = -1
```

```
if M == 1:
    M = 2
for i in range(M, N+1):
    count = True
    for j in range(2, int(i**0.5+1)):
        if i % j == 0:
            count = False
            break
    if count:
        sum_prime += i
        if min_prime == -1:
            min_prime = i
if sum_prime > 0:
    print(sum_prime)
print(min_prime)
```

```
#other good code
def eratosthenes(a, b):
    primes = [True for _ in range(b+1)]
    primes[0] = primes[1] = False
    res = []

    for i in range(2, b+1):
        if primes[i]:
            for j in range(i*i, b+1, i):
                primes[j] = False

    for i in range(a, b+1):
        if primes[i]:
            res.append(i)
    return res

a = int(input())
b = int(input())
res = eratosthenes(a,b)

if len(res) == 0 :
    print(-1)
else :
    print(sum(res))
    print(min(res))
```

백준 11653번 소인수분해 https://www.acmicpc.net/problem/11653
속도 차이가 몇배가 난다.. 한번에 공통 약수를 제거함으로써 속도차이를 낸 모습.. 배워야한다.

```
#my code
N = int(input())
while N != 1:
    for i in range(2, N+1):
        if N % i == 0:
            N = N//i
            print(i)
            break
```

```
#other good code
n = int(input())

for i in range(2, int(n ** 0.5)+1):
    while n % i == 0:
        print(i)
        n = n // i
```

```
if n >= 2:
    print(n)
```

백준 24313번 알고리즘 수업 - 점근적 표기1 https://www.acmicpc.net/problem/24313
Big-O에 대한 공부가 될 내용…. 이문제를 너무 복잡하게 생각함.. 그냥 함수대로 입력 후 비교하면 끝났음→ 다른 사람의 코드를 보고 공부하자.

```
#my code
import sys
a1, a0 = map(int, sys.stdin.readline().split())
c = int(sys.stdin.readline().rstrip())
n0 = int(sys.stdin.readline().rstrip())
if a1 > c:
    print(0)
elif a1 == c:
    if a0 > 0:
        print(0)
    else:
        print(1)
else:
    c -= a1
    if c*n0 >= a0:
        print(1)
    else:
        print(0)
```

```
#other good code
a1, a0 = map(int,input().split())
c=int(input())
n0=int(input())
print(1 if a1*n0+a0<=c*n0 and a1 <= c else 0)
```

백준 2798번 블랙잭 https://www.acmicpc.net/problem/2798
브루트포스 알고리즘
내 코드와 아래코드의 속도 차이는 2배 가까이 난다.(80m/s, 48m/s)
원인은 우선 len(a)를 사용함으로써 매 반복문마다 한번씩 실행되면서 시간이 소요됨
물론 그것보다 큰 차이는 sort여부. 밑에서는 sort를 내림차순으로 한 후, total≤m을 만족하는 최대값을 구한다음 그 뒤의 경우는 어처피 최대값보다 작으므로 break시켜서 속도를 빠르게함.
form sys import stdin으로 메모리적으로도 이득일 듯?

```
#my code
import sys
N, M = map(int, sys.stdin.readline().split())
a = list(map(int, sys.stdin.readline().split()))
maxi = 0
for i in range(len(a)-2):
    for j in range(i+1, len(a)-1):
        for k in range(j+1, len(a)):
            Sum = a[i] + a[j] + a[k]
              if Sum <= M and Sum > maxi:
                maxi = Sum
print(maxi)
```

```
#other good code
from sys import stdin
n, m = map(int, stdin.readline().split())
cards = list(map(int, stdin.readline().split()))
cards.sort(reverse=True)
ans = 0
for i in range(n-2):
    for j in range(i+1, n-1):
        for k in range(j+1, n):
            total = cards[i] + cards[j] + cards[k]
            if total <= m:
                ans = max(ans, total)
                break
print(ans)
```

백준 1018번 체스판 다시 칠하기 https://www.acmicpc.net/problem/1018
브루트포스 알고리즘
내 코드와 다른 코드의 큰 차이점은 없는듯. 사실장 4중 for문이라 속도가 많이 느릴 것 같았는데, 생각보다 느리지 않았다. (88m/s)

```
#my code
from sys import stdin
N, M = map(int, stdin.readline().split())
a = list()
for _ in range(N):
    a.append(stdin.readline().rstrip())
min_count = N*M
for i in range(N-7):
    for j in range(M-7):
        wcount = 0 #a[0][0] == 'W' 기준
        bcount = 0 #a[0][0] == 'B' 기준
        for p in range(i, i+8):
            for q in range(j, j+8):
                if (p+q) % 2 == 0:
                    if a[p][q] == 'B':
                        wcount += 1
                    else:
                        bcount += 1
                elif (p+q) % 2 == 1:
                    if a[p][q] == 'B':
                        bcount += 1
                    else:
                        wcount += 1
        count = min(wcount, bcount)
        if min_count > count:
            min_count = count
print(min_count)
```

```
#other good code
N,M=map(int,input().split())
lst=[]
for _ in range(N):
    lst.append(list(input().strip()))
ans=65
def square(i,j):
    min_BW=0
    row=0
    for k in range(i,i+8):
        row+=1
        for l in range(j,j+8,2):
            if row%2==1:
```

```
                if lst[k][l]=='W':
                    min_BW+=1
                if lst[k][l+1]=='B':
                    min_BW+=1
            else:
                if lst[k][l]=='B':
                    min_BW+=1
                if lst[k][l+1]=='W':
                    min_BW+=1
    if min_BW>32:
        min_BW=64-min_BW
    return min_BW

for i in range(N-7):
    for j in range(M-7):
        if ans>square(i,j):
            ans=square(i,j)
print(ans)
```

백준 1038번 감소하는수 https://www.acmicpc.net/problem/1038
배끼다싶이함.. *itertools라는 라이브러리에 순열, 조합 관련 함수가 있다는 것 기억하기.*

```
#my code
import itertools
N = int(input())
a = [9,8,7,6,5,4,3,2,1,0]
dec_num = list()
for i in range(1, 11):
    tmp = itertools.combinations(a, i)
    for k in tmp:
        dec_num.append(''.join(map(str, k)))
    if len(dec_num) > N:
        break
dec_num = list(map(int, dec_num))
dec_num.sort()
if N >= len(dec_num):
    print(-1)
else:
    print(dec_num[N])
```

```
#other good code
import sys
input = sys.stdin.readline
from itertools import combinations

N = int(input())

arr = []

for i in range(1, 11):
    tmp = combinations(range(10), i)
    for j in tmp:
        arr.append(''.join(map(str, j))[::-1])

arr = list(map(int, arr))
arr.sort()
if len(arr) <= N:
    print(-1)
else:
    print(arr[N])
```

백준 1436번 영화감독 숌 https://www.acmicpc.net/problem/1436
브루트포스로 쉽게 풀리나, 다른 방법으로 한번 찾아보고싶다…
in 이 기억이 안나서 풀이가 길어짐.. ㅎ

```
#my code
N = int(input())
num = 666
count = 0
while True:
    try:
        str(num).index('666')
        count += 1
        if count == N:
            break
    except ValueError:
        continue
    finally:
        num += 1
print(num -1)
```

```
#other good code
n = int(input())
cnt = 0
six_n = 666
while True:
    if '666' in str(six_n):
        cnt += 1
    if cnt == n:
        print(six_n)
        break
    six_n += 1
```

백준 2587번 대표값 https://www.acmicpc.net/problem/2587
문제자체는 매우 쉽지만 공부도 할겸 ***선택정렬 버블정렬 병합정렬을 구현***해보았다..
다른 정렬도 같이 공부해보고 특히 병합정렬과 같은 것들은 손코딩으로 작성할 수 있도록 외우기

```
import sys


def select_sort(k: list): #내림차순 선택정렬
    for p in range(len(k)-1):
        loc = p
        for q in range(p+1, len(k)):
            if k[loc] < k[q]:
                loc = q
        k[p], k[loc] = k[loc], k[p]
    return k


def bubble_sort(k: list): #내림차순 버블정렬
    for p in range(len(k)-1):
        for q in range(len(k)-p-1):
            if k[q] < k[q+1]:
                k[q], k[q+1] = k[q+1], k[q]
    return k


def merge_sort(k: list): #내림차순 병합정렬 - 좀더 깔끔하게 만들어보기
    length = len(k)
    if length <= 1:
```

```
        return k
    group1 = merge_sort(k[:length//2])
    group2 = merge_sort(k[length//2:])
    #print(group1, group2)
    result = list()
    index1 = index2 = 0
    while index1 < len(group1) and index2 < len(group2):
        if group1[index1] > group2[index2]:
            result.append(group1[index1])
            index1 += 1
        else:
            result.append(group2[index2])
            index2 += 1
    while index1 < len(group1):
        result.append(group1[index1])
        index1 += 1
    while index2 < len(group2):
        result.append(group2[index2])
        index2 += 1
    #print(result)
    return result

a = [int(sys.stdin.readline()) for _ in range(5)]
average = 0
for i in a:
    average += i
average //= 5
#print(a)
#print(select_sort(a)) #선택정렬
#print(bubble_sort(a)) #버블정렬
#print(merge_sort(a)) #병합정렬
a.sort()
print(average, a[2])
```

백준 11650번 좌표 정렬하기 https://www.acmicpc.net/problem/11650
sys.stdin.readlines()도 공부해보자.. ctrl-z 가 누르기 전까지 계속 입력을 받는다.

```
#my code
import sys
N = int(sys.stdin.readline())
point = list()
for _ in range(N):
    a, b = map(int, sys.stdin.readline().split())
    point.append((a, b))
point.sort(key = lambda x: (x[0], x[1]))
for i in point:
    print(i[0], i[1])
```

```
#other good code
from sys import stdin, stdout

dot = stdin.readlines()[1:]

dot.sort(key=lambda x: (int(x.split()[0]), int(x.split()[1])))

stdout.write(''.join(dot))
```

백준 18870번 좌표압축 https://www.acmicpc.net/problem/18870

index[x]는 O(N)이라 이진탐색O(logN)으로 바꾸었는데도 시간초과가 떠서 결국 다른 코드를 참고했다..
dictionary를 사용했더니 시간복잡도가 해당 인덱스를 찾는데 O(1)로 줄었다. dict의 장점을 확실히 공부해야겠다.
아래 사이트 참고...
https://velog.io/@kimwoody/Python-리스트와-딕셔너리의-주요-연산-시간-복잡도

```
#my code
from sys import stdin
N = int(stdin.readline())
x = list(map(int, stdin.readline().split()))
x_p = list(set(x))
x_p.sort()
for i in range(N):
    print(x_p.index(x[i]), end=' ')
```

```
#my code
from sys import stdin
N = int(stdin.readline())
x = list(map(int, stdin.readline().split()))
x_p = list(set(x))
x_p.sort()
for i in range(N):
    mid = len(x_p) // 2
    while True:
        if x_p[mid] == x[i]:
            print(mid, end=' ')
            break
        elif x_p[mid] > x[i]:
            mid //= 2
        else:
            mid += mid//2
```

```
#other good code
from sys import stdin
n = int(input())
numbers = list(map(int, stdin.readline().split()))
a = list(set(numbers))
a.sort()
num_dict = {}
for i in range(len(a)):
    num_dict[a[i]] = i
for i in numbers:
    print(num_dict[i], end=' ')
```

백준 10815번 숫자카드 https://www.acmicpc.net/problem/10815
집합으로 푸는 문제였다.. *set에서는 A in B가 평균 시간복잡도가 O{1}이고 list는 O(n)이다.(set과 dict은 파이썬에서 해시 테이블로 저장되기 때문)* 이진탐색을 이용해서 열심히 풀었는데, set를 이용했다면 애초에 sort()나 이진탐색을 할 필요도 없었다. 시간또한, 2배정도 차이가 났다. 문제에서 숫자카드에 대한 조건인 '두 숫자 카드에 같은 수가 적혀있는 경우는 없다.' 라는 조건을 잘 봤어야했다. 그리고 정수 하나하나를 print(1, end=' ')로 경우마다 출력하는 것 보다 ans = list(), ans.append('1')처럼 하나하나를 리스트에 담아서 한번에 print(' '.join(ans))로 출력하는 것이 시간이 더 빠르다는 것을 알 수 있었다.

아래 사이트 참고..
https://velog.io/@ready2start/Python-세트set의-시간-복잡도
https://wayhome25.github.io/python/2017/06/14/time-complexity/

```
#my code
from sys import stdin
N = int(stdin.readline())
first = list(map(int, stdin.readline().split()))
first.sort()
M = int(stdin.readline())
second = list(map(int, stdin.readline().split()))
for i in range(M):
    index = 0
    exist = False
    if (second[i] <= first[N-1]) or (second[i] >= first[0]):
        left = 0
        right = N-1
        while left <= right:
            index = (left+right)//2
            if second[i] == first[index]:
                exist = True
                break
            elif second[i] > first[index]:
                left = index+1
            else:
                if index == 0:
                    break
                right = index-1
        if second[i] == first[(left+right)//2]:
            exist = True
    if exist:
        print(1, end=' ')
    else:
        print(0, end=' ')
```

```
#other good code
import sys
input = sys.stdin.readline

# hash
def main():
    input()
    cards = set(input().split())

    input()
    nums = input().split()

    ans = []
    for num in nums:
        if num in cards:
            ans.append('1')
        else:
            ans.append('0')

    print(' '.join(ans))

main()

# 이분탐색..
# def main():
#     n = int(input())
#     cards = list(map(int, input().split()))
#     cards.sort()

#     m = int(input())
#     nums = list(map(int, input().split()))

#     check = set()

#     ans = []
#     for num in nums:
```

```
#         if num in check:
#             ans.append('1')
#             continue

#         if num < cards[0] or num > cards[-1]:
#             ans.append('0')
#             continue

#         left = 0
#         right = n-1

#         flag = 0
#         while left <= right:
#             mid = (left + right)//2

#             if cards[mid] == num:
#                 ans.append('1')
#                 check.add(num)
#                 flag = 1
#                 break
#             elif cards[mid] < num:
#                 left = mid + 1
#             else:
#                 right = mid - 1

#         if not flag:
#             ans.append('0')

#     print(' '.join(ans))

# main()
```

백준 14425번 문자열 집합 https://www.acmicpc.net/problem/14425
집합에 대해 알고나니 매우 쉽게 풀 수 있었다. trie라는 트리구조의 알고리즘으로 일부러 어렵게 푸시는 분들도 계신듯.. 아래 코드는 여러 코드들 중에 내가 이해할 만한 & 재밌는 코드라 가져왔다.

```
#my code
from sys import stdin
N, M = map(int, stdin.readline().split())
S_set = {stdin.readline().rstrip() for _ in range(N)}
check = [stdin.readline().rstrip() for _ in range(M)]
count = 0
for string in check:
    if string in S_set:
        count += 1
print(count)
```

```
#other good code
import sys
n, m = map(int, sys.stdin.readline().split())
answer = set()
for _ in range(n):
    answer.add(sys.stdin.readline())

s = 0
for _ in range(m):
    if sys.stdin.readline() in answer:
        s += 1
print(s)
```

```
//other good code
#include<iostream>

#include<string>
#include<vector>
#include<unordered_map>

using namespace std;



int main(void){
    int N, M;

    cin.tie(NULL);
    ios::sync_with_stdio(false);

    cin>>N>>M;

    unordered_map<string, bool> mp;

    string str;
    for(int i=0; i <N; i++){
        cin>>str;
        mp.insert({str, true});
    }

    int res =0;
    for(int i=0; i < M; i++){
        cin>>str;

        if(mp.find(str) != mp.end()){
            res++;
        }
    }

    printf("%d", res);
    return 0;
}
```

백준 1620번 나는야 포켓몬 마스터 이다솜 https://www.acmicpc.net/problem/1620
dictonary에서 value로 key를 찾는 법을 고민하느라 많은 시간을 썼다. 그런데 사실 key value를 바꾼 dict하나를 더 만들면 간단한 거였다.
여러 예제들의 코드 실행속도를 비교해본 결과 isdigit()함수의 속도가 내가 구현한 check_digit()보다 훨씬 빠르다(효율적이다)는 것을 알수 있었다.
그리고 백준에서 많은 사람들이 input = sys.stdin.readline 을 사용하는데 이를 이용할 때 실행속도가 조금 더 빨라졌다.(이건 이유를 모름..)

```
#my code
from sys import stdin


def check_digit(inp: str):
    try:
        int(inp)
    except ValueError:
        return False
    return True


N, M = map(int, stdin.readline().split())
en1 = dict()
```

```
en2 = dict()
for i in range(N):
    en1[i+1] = stdin.readline().rstrip()
    en2[en1[i+1]] = i+1
for _ in range(M):
    find = stdin.readline().rstrip()
    if check_digit(find):
        print(en1.get(int(find)))
    else:
        print(en2.get(find))
```

```
#my code
from sys import stdin
input = stdin.readline


def check_digit(inp: str):
    try:
        int(inp)
    except ValueError:
        return False
    return True


N, M = map(int, input().split())
en1 = dict()
en2 = dict()
for i in range(N):
    en1[i+1] = input().rstrip()
    en2[en1[i+1]] = i+1
for _ in range(M):
    find = input().rstrip()
    if find.isdigit():
        print(en1.get(int(find)))
    else:
        print(en2.get(find))
```

```
#other good code
import sys

n, m = map(int, sys.stdin.readline().split())
dict_num = {}
dict_pk = {}

ans = []
for i in range(1, int(n) + 1):
    pk = sys.stdin.readline().rstrip()
    dict_num[i] = pk
    dict_pk[pk] = i

for _ in range(m):
    pm = sys.stdin.readline().rstrip()
    if pm.isdigit():
        ans.append(dict_num[int(pm)])
    else:
        ans.append(str(dict_pk[pm]))

print("\n".join(ans))
```

백준 11478번 서로 다른 부분 문자열의 개수 https://www.acmicpc.net/problem/11478
파이썬의 문자열 슬라이싱 기능을 이용해 쉽게 풀 수 있었다. *inp[j:j+i+1] 에서 j와 j+i+1이 문자열의 길이를 초과해도 문제가 발생하지 않고 아무것도 리턴되지 않아 오류발생x*

처음에는 맨 아래 코드처럼 부분 문자열을 모두 구한 다음에 한번에 길이를 측정해서 출력하는 방식을 했는데, S에 너무 많는 부분 문자열이 들어가게 되면서 메모리 초과가 떴다. 그래서 부분 문자열의 길이를 기준으로 각 문자열을 set에 넣고 길이를 ans에 더하는 식으로 바꾸었다.

```
#my code
from sys import stdin
inp = stdin.readline().rstrip()
length = len(inp)
ans = 0
for i in range(length):
    S = set()
    for j in range(length - i):
        S.add(inp[j:j+i+1])
    ans += len(S)
print(ans)
```

```
#my first_code (out of memory)
from sys import stdin
S = set()
inp = stdin.readline().rstrip()
length = len(inp)
for i in range(length):
    for j in range(length - i):
        S.add(inp[j:j+i+1])
print(len(S))
```

백준 13241번 최소공배수 https://www.acmicpc.net/problem/13241
나름 잘 계산했다고 생각했는데, 대부분의 다른 풀이를 보니 A*B를 최대공약수로 나눈 방식으로 값을 구했다. 유클리드 호제법으로 이를 구하면 O(N)이 아닌 O(logN)의 시간복잡도로 찾아낼 수 있기 때문이다… 자세한 설명은 아래 사이트 참고.
https://myjamong.tistory.com/138 → 유클리드 호제법 알고리즘에 대한 설명 시간복잡도 O(logN)
if A%B == 0 최대공약수 :B else mod(A,B) = mod(B, A%B)

```
#my code
from sys import stdin
A, B = map(int, stdin.readline().split())
factor = 1
divisor = 2
if A*B<0:
    print("A와 B는 배수 관계가 아니다.")
if A<0 and B<0:
    factor *= -1
while (abs(A) != 1) and (abs(B) != 1):
    if divisor > A or divisor > B:
        break
    if (A % divisor == 0) and (B % divisor == 0):
        A //= divisor
        B //= divisor
        factor *= divisor
    else:
        divisor += 1
print(factor*A*B)
```

```
#other good code
import sys
```

```
A, B = map(int, sys.stdin.readline().split())

def gcd(x, y):
    mod = x % y
    while mod > 0:
        x = y
        y = mod
        mod = x % y
    return y

print(A * B // gcd(A, B))
```

백준 1269번 대칭 차집합 https://www.acmicpc.net/problem/1269
A.update(B)보다 hash를 이용한 A.intersection(B)가 훨씬 빠르다. update는 비교하고나서 A집합에 B의 원소를 추가하는 것도 시간이 소요되기 때문이다.

```
#my code
from sys import stdin
a, b = map(int, stdin.readline().split())
num1 = a+b
A = set(map(int, stdin.readline().split()))
B = set(map(int, stdin.readline().split()))
num2 = len(A.intersection(B))
print(num1-2*num2)
```

```
#other good code
#dictionary의 in은 키(key)에 한해서 동작합니다.
import sys
input = sys.stdin.readline

a, b = map(int,input().rstrip().split())


list_a = list(input().rstrip().split())
list_b = list(input().rstrip().split())
dict = {}
cnt=0

for i in list_a:
    dict[i] = 1

for i in list_b:
    if i in dict:
        cnt+=1


print(a+b-cnt*2)
```

백준 1735번 분수합 https://www.acmicpc.net/problem/1735
최대공약수를 이용해 숫자가 커지는 것을 방지한 후 구했다.

```
#my code
from sys import stdin
a1, a2 = map(int, stdin.readline().split())
b1, b2 = map(int, stdin.readline().split())


def gcd(x, y):
```

```
    mod = x % y
    while mod > 0:
        x = y
        y = mod
        mod = x % y
    return y


gcd_a = gcd(a2, a1)
gcd_b = gcd(b2, b1)
a1 //= gcd_a
a2 //= gcd_a
b1 //= gcd_b
b2 //= gcd_b
ans1 = a1*b2 + b1*a2
ans2 = b2*a2
gcd_ans = gcd(ans1, ans2)
print(ans1//gcd_ans, ans2//gcd_ans)
```

백준 2485번 가로수 https://www.acmicpc.net/problem/2485
내 풀이는 입력받은 값들의 차이(각 가로수의 거리)의 최솟값을 구한 후, 그 최솟값의 약수들 중에서 거리의 최대공약수를 구한 후, 이를 가로수의 위치의 차에 나누어 값을 구하는 방식이다.
다른 풀이는 입력 값이 최대 10만개 이기 때문에 각 거리들을 차례대로 최대공약수를 구한 다음 이를 가로수의 위치의 차에 나누어 값을 구하는 방식이다.
나는 내방식이 더 효율적이라고 생각했는데, 어림도 없었다.

```
#my code
from sys import stdin
tree = list()
N = int(stdin.readline().rstrip())
a = int(stdin.readline().rstrip())
first = int(stdin.readline().rstrip())-a
tree.append(first)
min_distance = first
distance = {min_distance}
for i in range(1, N-1):
    tree.append(int(stdin.readline().rstrip())-a)
#tree.sort()
for i in range(1, N-1):
    distance.add(tree[i]-tree[i-1])
    if min_distance > (tree[i]-tree[i-1]):
        min_distance = tree[i]-tree[i-1]
factor = list()
count = 1
while min_distance+1 > count**2:
    if min_distance % count == 0:
        factor.extend([count, min_distance // count])
    count += 1
factor = list(set(factor))
factor.sort()
#print(tree)
#print(min_distance, distance, factor)
gcd = 1
for i in range(len(factor)):
    flag = True
    for j in distance:
        if j % factor[i] != 0:
            flag = False
            break
    if flag:
        gcd = factor[i]
#print(gcd)
print(max(tree)//gcd-N+1)
```

```
#other good code
import sys
from math import gcd

input = sys.stdin.readline

N = int(input())
ar = [int(input()) for _ in range(N)]
ar.sort()

ans = ar[1] - ar[0]
for i in range(1, N - 1):
  d = ar[i + 1] - ar[i]
  ans = gcd(ans, d)

print((ar[-1] - ar[0]) // ans - N + 1)
```

백준 1929번 소수 구하기 https://www.acmicpc.net/problem/1929
#시간복잡도가 최대 O(Nlog(N))이라고 생각해서 풀었는데, 시간초과가 발생했다. 시간이 충분할 줄 알았는데, 평균 O((N-M)log(N))으로는 어림도 없었다. 이에, 질문게시판에서 약간 힌트를 받아 두 번째 코드를 작성했다.
#두 번째 코드는 두 for문을 바꿔주어 break를 사용할 수 있게 했다. break덕에 약수를 찾자마자 바로바로 넘겨서 시간을 줄였다.
#다른 분의 코드는 실행시간이 말도 안되게 짧다.. 갓.. list 슬라이싱을 이용해 멋지게 풀었는데 최대한 이해해보자.
sieve[i*i::2*i] = [False]*len(sieve[i*i::2*i]) 이부분은

```
a = [1,2,3]
b = a[:]
print(a, b)
a[1:] = [4, 5]
print(a, b)
b[1:] = [6, 7]
print(a, b)
#을 이해하면 알수 있다.
```

```
#my code1 time_over
from sys import stdin
M, N = map(int, stdin.readline().split())
prime = [False for _ in range(N+1)]
prime[0] = prime[1] = True
for i in range(2, int(N**0.5+1)):
    for j in range(i+1, N+1):
        if j % i == 0:
            prime[j] = True
for i in range(M, N+1):
    if not prime[i]:
        print(i)
```

```
#my code2 1000~5000m/s
from sys import stdin
M, N = map(int, stdin.readline().split())
prime = [False for _ in range(N+1)]
prime[0] = prime[1] = True
for i in range(M, N+1):
    for j in range(2, int(i**0.5+1)):
        if i % j == 0:
            prime[i] = True
```

```
            break
    if not prime[i]:
        print(i)
```

```
#other good code 100m/s
def prime(n):
    sieve = [True] * n
    for i in range(3,int(n**.5)+1,2):
        if sieve[i]:
            sieve[i*i::2*i] = [False]*len(sieve[i*i::2*i])
    return [2] + [i for i in range(3,n,2) if sieve[i]]

m,n=map(int,input().split())

for i in prime(n+1):
    if i>=m: print(i)
```

백준 4948번 베르트랑 공준 https://www.acmicpc.net/problem/4948
일단 입력값이 0으로 주어질 때 까지 계속 입력을 받아야하기에 미리 n=123456*2까지의 소수를 구해놓는 것도 좋은 방법인 것 같다. n이 무리하게 크지도 않았고 얼마나 입력을 받을 지 모르기 때문이다.
일단 소수를 구하는 내방식이 윗문제부터 너무 시간적으로 손해를 보는 알고리즘 이라는 것을 알게되었다. 에라토스테네스의 체에 대한 이해가 아직 부족한 것 같다.
마지막 코드는 다른 사람의 코드에서 binary_search와 소수 탐색 코드를 조금 더 빠르게 고쳐보았다. 매우 효율적인 코드가 되었다. 앞으로 이런 유형의 문제는 바로바로 풀 수 있도록 해야할 것같다.

```
#my code
from sys import stdin
while True:
    n = int(stdin.readline().rstrip())
    if n == 0:
        break
    count = 0
    #ans= list()
    if n%2 == 0:
        for i in range(n+1, 2 * n, 2):
            flag = True
            for j in range(2, int(i**0.5+1)):
                if i%j == 0:
                    flag = False
                    break
            if flag:
                count += 1
                #ans.append(i)
    elif n == 1:
        count = 1
    else:
        for i in range(n+2, 2*n, 2):
            flag = True
            for j in range(2, int(i ** 0.5+1)):
                if i % j == 0:
                    flag = False
                    break
            if flag:
                count += 1
                #ans.append(i)
    #print(count, ans)
    print(count)
```

```
#other good code
import sys
from bisect import *
input=sys.stdin.readline
def sol():
    n=123456*2
    prime=[True]*(n+1)
    prime[:2]=[False,False]
    for i in range(2,int(n**0.5)+1):
        if prime[i]:
            prime[i*2::i]=[False]*(n//i-1)
    primes=[i for i in range(n+1) if prime[i]]
    while True:
        nn=int(input())
        if nn==0:
            break
        start=bisect_right(primes,nn) #primes에서 nn이 있다면 그 값의 다음 인덱스를 반환. 없으면 그냥 똑같은 binary_search
        end=bisect_right(primes,2*nn)
        print(end-start)

sol()
```

```
#fixed code
from sys import stdin
def bisect_right(li: list, x:int):
    left = 0
    right = len(li)-1
    flag = True
    while left <= right:
        mid = (left+right)//2
        if x > li[mid]:
            left = mid+1
            flag = False
        elif x < li[mid]:
            right = mid-1
        else:
            return mid + 1
    if flag:
          return right+1
    else:
        return left

n = 123456 * 2
prime = [True] * (n + 1)
for i in range(3, int(n ** 0.5) + 1, 2):
    if prime[i]:
        prime[i*i::i] = [False] * (n // i - i+1)
primes = [2]+[i for i in range(3, n + 1, 2) if prime[i]]
while True:
    nn = int(stdin.readline().rstrip())
    if nn == 0:
        break
    start = bisect_right(primes, nn)
    end = bisect_right(primes, 2 * nn)
    print(end - start)
```

백준 17103번 골드바흐 파티션 https://www.acmicpc.net/problem/17103
4948번인 베르트랑 공준에서 binary_search와 소수찾는 함수를 이용해서 풀었다.
밑의 방식은 처음에는 이해가 잘 안됐는데, 알고보니 내 코드에서의 prime을 이용한 것이었다. 굳이 set을 만들어 in 을 쓰지 않아도 if prime[N-primes_list[i]] == True 인지 확인하면 되는 것이었다.

n = 1000000
*prime = [True] * (n+1)*
*for i in range(3, int(n**0.5)+1, 2):*
if prime[i]:
*prime[i*i::i] = [False] * len(prime[i*i::i])*
primes_list = [2] + [i for i in range(3, n+1, 2) if prime[i]]
→ 이방식은 정말 소수를 list로 가장 빨리 구하는 방법인 것 같다.

```
#my code 508m/s
from sys import stdin
def binary_search(li:list, x:int):
    start = 0
    end = len(li)-1
    while start <= end:
        mid = (start + end) // 2
        if li[mid] == x:
            return mid
        elif li[mid] > x:
            end = mid - 1
        else:
            start = mid + 1
    return start


n = 1000000
prime = [True] * (n+1)
for i in range(3, int(n**0.5)+1, 2):
    if prime[i]:
        prime[i*i::i] = [False] * len(prime[i*i::i])
primes_list = [2] + [i for i in range(3, n+1, 2) if prime[i]]
primes_set = set(primes_list)
T = int(stdin.readline().rstrip())
for _ in range(T):
    ans = 0
    N = int(stdin.readline().rstrip())
    if N//2 in primes_set:
        ans += 1
    for i in range(binary_search(primes_list, N//2)):
        if (N-primes_list[i]) in primes_set:
            ans += 1
    print(ans)
```

```
#other good code 208m/s
n = 10 ** 6
a = [0,0] + [1] * n
primes = []
for i in range(2, n):
  if a[i] == 0:
    continue
  primes.append(i)
  for j in range(i*i,n,i):
    a[j] = 0

for _ in range(int(input())):
  n = int(input())
  ans = 0
  for v in primes:
    if v + v > n:
      break
    ans += a[n - v]
  print(ans)
```

백준 25192번 인사성 밝은 곰곰이 https://www.acmicpc.net/problem/25192
쉬운 문제 & 발상의 전환. 다른 사람의 정답을 보던 중, 나처럼 입력 때마다 확인하는 것이 아니라, ENTER가 입력되면 한번에 set의 길이만큼 ans에 더하는 것이 더 깔끔하고 좋은 방식인 것 같아 가져왔다.

```
#my code 100m/s
from sys import stdin
N = int(stdin.readline().rstrip())
chatting = set()
ans = 0
for _ in range(N):
    chat = stdin.readline().rstrip()
    if chat == 'ENTER':
        chatting.clear()
    elif chat not in chatting:
        chatting.add(chat)
        ans += 1
print(ans)
```

```
#other good code 80m/s
import sys
input = sys.stdin.readline
a = int(input())
n = 0
s = set()
for _ in range(a):
    name = input()
    if name == 'ENTER\n':
        n += len(s)
        s = set()
    else:
        s.add(name)
n += len(s)
print(n)
```

백준 2108번 통계학 https://www.acmicpc.net/problem/2108
round함수가 math모듈에 있는 줄 알았는데, 기본으로 제공해주는 메소드였다. 반올림을 할 때 사용하면 좋다.
round(실수, 반올림 할 소수점 자릿수(default값 0))

```
#my code
from sys import stdin
N = int(stdin.readline().rstrip())
li = list()
mode_count = [0]*8001
mode = 4001
for _ in range(N):
    inp = int(stdin.readline().rstrip())
    mode_count[inp+4000] += 1
    li.append(inp)
li.sort()
flag = False
top = max(mode_count)
for i in range(len(mode_count)):
    if mode_count[i] == top:
        if not flag:
            flag = True
            mode = i
        else:
            mode = i
```

```
            break
print(round(sum(li)/N), li[N//2], mode-4000, li[-1] - li[0], sep='\n')
```

백준 20920번 영단어 암기는 어려워 https://www.acmicpc.net/problem/20920
lambda함수와 dictionary에 대해서 다시 공부 할 수 있었다. 아래 코드는 고수분의 코드인데 이해하기가 어렵다. 일단 패스! (dict(얼마나 많이 나왔는지)안에 dict(단어의 길이)을 넣고 거기에 다시 list로 단어들을 넣었다.)

```
#my code
from sys import stdin
N, M = map(int, stdin.readline().split())
word_book = dict()
for _ in range(N):
    word = stdin.readline().rstrip()
    if len(word) < M:
        continue
    elif word in word_book:
        word_book[word] += 1
    else:
        word_book[word] = 1
word_book = list(word_book.items())
word_book.sort(key=lambda x: (-x[1], -len(x[0]), x[0]))
for i in word_book:
    print(i[0])
```

```
#other good code
import sys
input = sys.stdin.readline

N, M = map(int, input().split())
countDict = {}

for _ in range(N):
    word = input().rstrip()

    if len(word) >= M:
        countDict[word] = countDict.get(word, 0) + 1

orderDict = {}

for k, v in countDict.items():
    length = len(k)

    if v not in orderDict:
        orderDict[v] = {}

    if length not in orderDict[v]:
        orderDict[v][length] = []

    orderDict[v][length].append(k)

for i in sorted(orderDict, reverse=True):
    for j in sorted(orderDict[i], reverse=True):
        for v in sorted(orderDict[i][j]):
            print(v)
```

백준 10828번 스택 https://www.acmicpc.net/problem/10828
스택의 기본을 배워보자.

```
#my code
from sys import stdin
command = list()


def push(i: int):
    command.append(i)


def size():
    return len(command)


def pop():
    if size() > 0:
        return command.pop(-1)
    else:
        return -1


def empty():
    if size():
        return 0
    else:
        return 1


def top():
    if size():
        return command[-1]
    else:
        return -1


N = int(stdin.readline().rstrip())
for _ in range(N):
    com = stdin.readline().split()
    if com[0] == 'push':
        push(int(com[1]))
    elif com[0] == 'pop':
        print(pop())
    elif com[0] == 'size':
        print(size())
    elif com[0] == 'empty':
        print(empty())
    elif com[0] == 'top':
        print(top())
```

```
#other good code
from sys import stdin
input = stdin.readline
li = []
for i in range(int(input())):
  s = input().split()
  if s[0] == 'push':
    li.append(s[1])
  elif s[0] == 'pop':
    if len(li) != 0:
      print(li.pop())
    else:
      print(-1)
  elif s[0] == 'size':
    print(len(li))
  elif s[0] == 'empty':
    print(1 if len(li) == 0 else 0)
  else:
    print(li[-1] if len(li) > 0 else -1)
```

백준 1874번 스택 수열 https://www.acmicpc.net/problem/1874
문제 자체는 생각보다 쉬웠다. 시간도 많이 소요될 줄 알았는데 그러지 않았다.
'\n'.join(answer)가 속도적인 측면에서 굉장히 효율적인 방식이라는 것을 알 수 있었다. (80m/s단축)

```
#my code 200m/s
from sys import stdin
n = int(stdin.readline().rstrip())
stack = list()
command = list()
num = 0
for _ in range(n):
    element = int(stdin.readline().rstrip())
    if num < element:
        command.extend(['+']*(element - num))
        while num < element:
            num += 1
            stack.append(num)
    if num >= element:
        check = stack.pop()
        if check == element:
            command.append('-')
        else:
            print('NO')
            command.clear()
            break
if command:
    for i in command:
        print(i)
```

```
#other good code 100m/s
# 1874번 : 스택 수열 - Silver 2
import sys
input = sys.stdin.readline
"""

"""
def solution():
    n = int(input())
    stack = []
    count = 1 #넣을수
    answer = []
    for i in range(n):
        num = int(input())
        while count <= num:
            stack.append(count)
            answer.append("+")
            count += 1
        if stack[-1] == num:
            stack.pop()
            answer.append("-")
        else:
            answer = ["NO"]
            break
    print('\n'.join(answer))
solution()
```

백준 18258번 큐 2 https://www.acmicpc.net/problem/18258
지금까지 list를 너무 만능이라고 생각한 것 같다. *pop()은 스택연산이라 시간복잡도 O(1)이지만, pop(0)는 나머*

지 부분의 전체 복사가 필요하므로 시간복잡도 O(n)이라는 것을 알게되었다. 또한 collections라는 모듈에 deque(데크)가 있다는 것과, 이것을 사용하면 pop(0)을 popleft()로 사용할 수 있어 시간복잡도가 O(1)로 감소한다는 것을 알 수 있었다.

```
#my code timeover (upper 3000m/s)
from sys import stdin


def push(x: int, li: list):
    li.append(x)


def pop(li: list):
    if not li:
        print(-1)
    else:
        print(li.pop(0))


def size(li: list):
    print(len(li))


def empty(li: list):
    if li:
        print(0)
    else:
        print(1)


def front(li: list):
    if not li:
        print(-1)
    else:
        print(li[0])


def back(li: list):
    if not li:
        print(-1)
    else:
        print(li[-1])


queue = list()
n = int(stdin.readline().rstrip())
for _ in range(n):
    command = stdin.readline().split()
    if command[0] == 'push':
        push(int(command[1]), queue)
    elif command[0] == 'pop':
        pop(queue)
    elif command[0] == 'size':
        size(queue)
    elif command[0] == 'empty':
        empty(queue)
    elif command[0] == 'front':
        front(queue)
    elif command[0] == 'back':
        back(queue)
```

```
#my other code (1600m/s)
from sys import stdin
from collections import deque

queue = deque()
```

```
n = int(stdin.readline().rstrip())
for _ in range(n):
    command = stdin.readline().split()
    if command[0] == 'push':
        queue.append(command[1])
    elif command[0] == 'pop':
        if queue:
            print(queue.popleft())
        else:
            print(-1)
    elif command[0] == 'size':
        print(len(queue))
    elif command[0] == 'empty':
        if queue:
            print(0)
        else:
            print(1)
    elif command[0] == 'front':
        if queue:
            print(queue[0])
        else:
            print(-1)
    elif command[0] == 'back':
        if queue:
            print(queue[-1])
        else:
            print(-1)
```

백준 1966번 프린터 큐 https://www.acmicpc.net/problem/1966
문제를 너무 복잡하게 생각한 것 같다. max(queue)가 시간을 많이 차지하리라 믿었는데, 생각해보니 N이 크지 않아서 시간문제는 없었다. 인덱스와 값을 동시에 deque에 넣으려고 많은 시간을 소요했는데, 아래 방식처럼 다른 index 리스트나 데크를 만들면 되는 것이었다.

```
#my code 64m/s
from sys import stdin
from collections import deque

test_case = int(stdin.readline().rstrip())
for _ in range(test_case):
    q = deque()
    N, M = map(int, stdin.readline().split())
    a = list((map(int, stdin.readline().split())))
    top = max(a)
    num_count = [a.count(i) for i in range(0, top+1)]
    ans = 0
    for i in range(N):
        if i == M:
            q.append([a[i], True])
        else:
            q.append([a[i], False])

    while True:
        element = q.popleft()
        if element[1]:
            if element[0] == top:
                print(ans+1)
                break
            else:
                q.append(element)
        elif element[0] == top:
            ans += 1
            num_count[top] -= 1
            if num_count[top] == 0:
                for i in range(top, 0, -1):
                    if num_count[i] != 0:
```

```
                        top = i
                        break
        else:
            q.append(element)
```

```
#other good code 40m/s
import sys
t = int(input())

while t > 0:
  n, k = map(int,sys.stdin.readline().split())
  queue = list(map(int,sys.stdin.readline().split()))
  index = list(range(len(queue)))#인덱스
  index[k] = -1#찾아야 되는 인덱스를 따로 표시한다

#만약 중요도가 제일 높은데 그것이 찾아야하는 문서이면 출력
#중요도가 더 높은 문서가 있으면 큐 맨 오른쪽으로 옮긴다.

  cnt = 0
  while True:
    now = queue[0]
    if now == max(queue):
      cnt += 1
      if index[0] == -1:
        print(cnt)
        break
      else:
        queue.pop(0)
        index.pop(0)
    else:
      queue.append(queue.pop(0))
      index.append(index.pop(0))
  t -= 1
```

백준 10866번 덱 https://www.acmicpc.net/problem/10866
덱을 list로 구현한 것이 사실 맞지만, 시간 초과가 날까봐 deque를 그냥 이용해버렸다. del[0]와 insert 모두 빅O는 O(N)인데도 list를 사용한 방식이 더 빨랐다…

```
#my code 80m/s
from sys import stdin
from collections import deque

N = int(stdin.readline().rstrip())
deq = deque()
for _ in range(N):
    command = stdin.readline().split()
    if command[0] == 'push_front':
        deq.appendleft(int(command[1]))
    elif command[0] == 'push_back':
        deq.append(int(command[1]))
    elif command[0] == 'pop_front':
        if deq:
            print(deq.popleft())
        else:
            print(-1)
    elif command[0] == 'pop_back':
        if deq:
            print(deq.pop())
        else:
            print(-1)
    elif command[0] == 'size':
        print(len(deq))
    elif command[0] == 'empty':
```

```
        if deq:
            print(0)
        else:
            print(1)
    elif command[0] == 'front':
        if not deq:
            print(-1)
        else:
            print(deq[0])
    elif command[0] == 'back':
        if not deq:
            print(-1)
        else:
            print(deq[-1])
```

```
#other good code 44m/s
import sys
n = int(sys.stdin.readline())
deque = []

for i in range(n):
    cmd = sys.stdin.readline().split()

    if cmd[0] == 'push_front':
        deque.insert(0,int(cmd[1]))
    elif cmd[0]=='push_back':
        deque.append(int(cmd[1]))

    elif cmd[0]=='pop_front':
        if len(deque)>0:
            print(deque[0])
            del deque[0]
        else:
            print(-1)

    elif cmd[0]=='pop_back':
        if len(deque)>0:
            print(deque.pop())
        else:
            print(-1)

    elif cmd[0]=='size':
        print(len(deque))

    elif cmd[0]=='empty':
        if len(deque)>0:
            print(0)
        else:
            print(1)

    elif cmd[0] =='front':
        if len(deque)>0:
            print(deque[0])
        else:
            print(-1)

    elif cmd[0]=='back':
        if len(deque)>0:
            print(deque[-1])
        else:
            print(-1)
```

백준 1021번 회전하는 큐 https://www.acmicpc.net/problem/1021
문제를 이해하는데 시간이 조금 걸렸던 것 같다. 더 문제를 많이 풀어서 큐, 덱, 스택을 쓰는 문제에 대한 이해를

높여야 할 것 같다.
+ 반복문 대신 que.rotate(f_index)를 써도 되었을 것 같다.

```
#my code 68m/s
from sys import stdin
from collections import deque

N, M = map(int, stdin.readline().split())
que = deque()
find_num = deque()
que.extend(range(1, N+1))
find_num.extend(map(int, stdin.readline().split()))
ans = 0
while find_num:
    if find_num[0] == que[0]:
        find_num.popleft()
        que.popleft()
    else:
        f_index = que.index(find_num[0])
        if f_index > len(que) - f_index:
            for _ in range(len(que) - f_index):
                que.appendleft(que.pop())
                ans += 1
        else:
            for _ in range(f_index):
                que.append(que.popleft())
                ans += 1

print(ans)
```

```
#other good code 36m/s
n, m = map(int, input().split())
nums = map(int, input().split())
queue = list(range(1,n+1))
ans = 0
for num in nums:
    idx = queue.index(num)
    left = len(queue)-idx
    if idx<left:
        ans += idx
        for _ in range(idx):
            queue.append(queue.pop(0))
    else:
        ans += left
        for _ in range(left):
            queue.insert(0, queue.pop())
    queue.pop(0)
print(ans)
```

백준 5430번 AC https://www.acmicpc.net/problem/5430 골드5 문제인데, 너무 오래걸렸다.
R이 나왔을 때, 배열을 뒤집어야하지만, 사실 뒤집게 되면 너무 많은 시간이 소요된다. 처음에는 이를 정직하게 reverse()했더니 어림도 없었고, 다른 분들의 힌트를 얻어서 rev변수를 이용해 겨우겨우 풀었다.
print('[' + ','.join(map(str, arr)) + ']') 여기서도 계속 에러가 발생하여 원인을 찾았는데, *''.join()은 list에 str이 아닌 정수형이나 실수형같은 다른 자료형이 담겨있을 경우, 문자열형으로 바꿔줘야 한다는 것이다.* 다시한번 join()에 대해 공부할 수 있었다.

다른분의 코드는 독보적이다. deque를 사용하지 않고 popleft(), pop()대신에 front_del_num, back_del_num을 이용해 범위를 줄인 다음, 나중에 list를 슬라이싱해서 출력하는 방식으로 풀었다.!

```
#my code (timeover.. + error)
from sys import stdin
from collections import deque

T = int(stdin.readline().rstrip())
for _ in range(T):
    p = stdin.readline().rstrip()
    n = int(stdin.readline().rstrip())
    arr = stdin.readline().rstrip().replace('[', '').replace(']', '')
    if arr:
        arr = deque(map(int, arr.split(',')))
    else:
        arr = deque()
    flag = True
    for i in range(len(p)):
        if p[i] == 'R':
            arr.reverse()
        elif p[i] == 'D':
            if arr:
                arr.popleft()
            else:
                print('error')
                flag = False
                break
    if flag:
        print(list(arr))
```

```
#my other code 268m/s
from sys import stdin
from collections import deque

T = int(stdin.readline().rstrip())
for _ in range(T):
    p = stdin.readline().rstrip()
    n = int(stdin.readline().rstrip())
    arr = stdin.readline().rstrip()[1:-1]
    # print(arr, p)
    if p.count('D') > n:
        print('error')
    else:
        if arr:
            arr = deque(map(int, arr.split(',')))
        else:
            arr = deque()
        rev = True
        for i in range(len(p)):
            if p[i] == 'R':
                if rev:
                    rev = False
                else:
                    rev = True
            elif p[i] == 'D':
                if rev:
                    arr.popleft()
                else:
                    arr.pop()
        if rev:
            print('[' + ','.join(map(str, arr)) + ']')
        else:
            arr.reverse()
            print('[' + ','.join(map(str, arr)) + ']')
```

```
#other good code 120m/s
# 구현
# AC
```

```
# 골드 V

import sys


def solution():

    T = int(sys.stdin.readline())

    for _ in range(T):
        p = sys.stdin.readline().rstrip()
        n = int(sys.stdin.readline().rstrip())

        A = sys.stdin.readline().rstrip()
        if A == '[]':
            A = []
        else:
            A = A[1:-1].split(",")


        front_del_num = 0
        back_del_num = 0

        reversed = False

        for function in p:
            if function == 'R':
                reversed = False if reversed else True

            elif function == 'D':
                if n - (front_del_num+back_del_num+1) < 0:
                    A = None
                    break

                elif reversed:
                    back_del_num += 1

                else:
                    front_del_num += 1


        if A is None:
            print('error')
        else:
            A = A[front_del_num:n-back_del_num]
            if reversed:
                print(f"[{','.join(A[::-1])}]")
            else:
                print(f"[{','.join(A)}]")

solution()
```