**문제 1. 비밀번호 ( # 1816 )**

확인하고자 하는 수가 주어집니다.

확인하고자 하는 수가 적절한 비밀번호 인지 아닌지를 구하는 프로그램을 작성해주세요.

적절하다면 YES, 적절하지 않다면 NO를 출력하면 됩니다.

적절한 비밀번호

* 모든 소인수가 1,000,000 보다 크다면 적절한 비밀번호입니다.

소인수란?

* 1을 제외한 약수를 말합니다.

**3**

**1000036000099**

**1500035500153**

**20000000000002**

**YES**

**NO**

**NO**

**문제 2. 사탕 ( # 14568 )**

친구 A,B,C에게 사탕을 나누어 주려고 합니다.

조건은 아래와 같습니다.

1. 남는 사탕이 없어야 합니다.

2. A는 B보다 2개 이상 많은 사탕을 가져야 합니다.

3. 셋 중 사탕을 하나도 못 받는 친구는 없어야 합니다.

4. C가 받는 사탕의 수는 짝수입니다.

분배 가능한 경우의 수를 출력하는 프로그램을 작성해주세요.

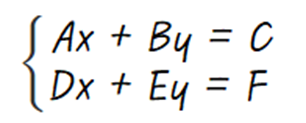
**6**

**1**

**문제 3. 연립방정식 ( # 19532 )**

숫자 A,B,C,D,E,F 가 주어집니다.

다음 연립방정식에서 x와 y값을 계산하는 프로그램을 작성하세요.



**범위**

* X와 Y는 -10000이상 10000이하인 정수이다.

**1 3 -1 4 1 7**

**2 -1**

**문제 4. 숫자야구 ( # 2503 )**

A는 3자리 숫자로 된 정답을 하나 정합니다.

B는 3자리 숫자를 제시해서 A가 생각하고 있는 정답을 맞히려고 합니다.

B가 말한 숫자가 정답에 포함되어 있다면 1 Ball입니다.

B가 말한 숫자가 정답에 포함되어 있고, 자리도 동일하다면 1 Strike입니다.

다른 숫자로 이루어진 세 자리수

Strike와 Ball의 결과를 보고, 가능한 숫자를 계산하는 프로그램을 작성하세요.

**4**

**123 1 1**

**356 1 0**

**327 2 0**

**489 0 1**

**2**

**문제 5. 모이기 ( # 1090 )**

N명의 학생들이 모각코를 하기 위해서 한 곳에서 모이려고 합니다.

학생들은 어디에 모여도 괜찮으나, 모든 사람들의 이동 거리를 합쳤을 때, 가장 적은 이동 거리였으면 좋겠다고 주장합니다.

N명의 학생의 집의 위치가 Y, X 2차원으로 주어졌을 때,

1,2,3, …, N명의 학생들이 모일 수 있는 최소 거리를 계산하는 프로그램을 작성하세요.

**조건**

* **N - 50 이하의 수**
* **X,Y좌표는 1\_000\_000이하의 수**

**4**

**15 14 - A 짱구**

**15 16 - B 철수**

**14 15 - C 맹구**

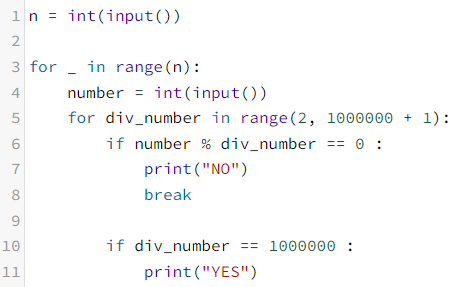
**16 15 - D 유리**

**훈이x**

**0 2 3 4**

정답코드

(1)



(2)

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

(3)

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

(4)

n = int(input())

numbers = [list(map(str,input().split())) for \_ in range(n)]

# string을 미리 넣어준 이유는, 나중에 쪼개기 위해서

answer = 0

# (1) 세 자리 숫자 만들기

for a in range(1,10): # 100의 자리수

for b in range(1,10): # 10의 자리수

for c in range(1,10): # 1의 자리수

counter = 0

# (2) 다른 세 자리수

if( a == b or b == c or c == a):

continue

# continue, 그 숫자를 넘김

# break, 반복문을 넘김

# (3) 배열에 넣은 조건을 넣어주기

for array in numbers:

check = list(array[0]) # ['1','2','3']

strike = int(array[1])

ball = int(array[2])

strike\_count = 0

ball\_count = 0

#스트라이크 계산기

if (a == int(check[0])):

strike\_count += 1

if (b == int(check[1])):

strike\_count += 1

if (c == int(check[2])):

strike\_count += 1

#볼 계산기

if (a == int(check[1]) or a == int(check[2])):

ball\_count += 1

if (b == int(check[0]) or b == int(check[2])):

ball\_count += 1

if (c == int(check[0]) or c == int(check[1])):

ball\_count += 1

#(4) 매칭 여부 확인하기

if (strike != strike\_count):

break

if (ball != ball\_count):

break

counter += 1

if counter == n:

answer += 1

print(answer)

(5)

n = int(input())

arr = []

arr\_y = []

arr\_x = []

answer = [-1]\*n

# 입력 받기

for \_ in range(n):

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

arr.append([a,b])

arr\_y.append(b)

arr\_x.append(a)

# 만날 장소 정하기

for y in arr\_y:

for x in arr\_x:

dist = []

# 만날 장소로 각각의 점들이 오는 비용 계산하기

for ex,ey in arr:

d = abs(ex-x) + abs(ey-y)

dist.append(d)

# 가까운 순서대로 정렬하기

dist.sort()

tmp = 0

for i in range(len(dist)):

d = dist[i]

tmp += d

if answer[i] == -1:

answer[i] = tmp

else :

answer[i] = min(tmp, answer[i])

print(\*answer)