# BANH

#Tính tổng số bánh bông lan kem rồi chia lấy dư với 28 học sinh để tìm ra số bánh còn dư sau khi phát cho 28 học sinh

with open('BANH.inp', 'r') as inp:

#Mở file ‘BANH.inp’ và gán nó vào biến inp

l = int(inp.readline().rstrip('\n'))

n = int(inp.readline().rstrip('\n'))

#Nhập l và n, rstrip('\n') dùng để xóa chuỗi xuống dòng sau khi nhập

with open('BANH.out', 'w') as out:

#Mở file ‘BANH.out’ và gán nó vào biến out

out.write('{}'.format((l \* 8 + n \* 3) % 28))

#In ra (l \* 8 + n \* 3) % 28 vào file ‘BANH.out’ thông qua biến out

# CHUOI

#Tạo hàm kiểm tra xem chuỗi đó có phải chuỗi đối xứng hay không rồi dùng hàm đó kiểm tra các chuỗi con có độ dài từ lớn đến bé

def checkReverse(s):

#hàm để kiểm tra xem chuỗi có phải là một chuỗi đối xứng không

if s == s[::-1]:

#nếu chuỗi s bằng chuỗi đảo ngược của s

return len(s)

#đúng thì trả về độ dài chuỗi s

else:

return 1

#sai thì trả về 1

def solve(s):

#hàm giải ra kết quả

for i in range(len(s), 1, -1):

#lặp lần lượt qua các số kí tự có thể là chuỗi con của chuỗi s

#(chạy từ lớn đến bé để tiện cho việc tìm chuỗi con đối xứng có độ dài lớn nhất)

#range(x,y,-1) có nghĩa là chạy từ y-1 tới x với số bước là -1(downto)

for j in range(len(s) - i + 1):

#lặp qua các giá trị vị trí đầu tiên của chuỗi con có i kí tự của s

if checkReverse(s[j:j+i]) > 1:

#nếu chuỗi con là chuỗi đối xứng

return checkReverse(s[j:j+i])

#trả về độ dài của chuỗi con đó

return 1

#nếu chạy hết vòng lặp mà chưa trả về giá trị nào thì trả về 1 vì theo như đề bài thì 'x' cũng là một chuối đối xứng

with open('CHUOI.INP', 'r') as inp:

#Mở file ‘CHUOI.inp’ và gán nó vào biến inp

s = inp.readline().rstrip('\n') #Nhập s từ file inp

with open('CHUOI.OUT', 'w') as out:

#Mở file ‘CHUOI.out’ và gán nó vào biến out

out.write('{}'.format(solve(s))) #In ra kết quả từ solve(s)

# DAUTHAU

#Tìm giá thầu cao nhất và in ra tên của người đấu thầu

result = ''

#biến chứa kết quả

max = 0

#gán max bằng 0 vì giá thầu là một số nguyên dương nên sẽ luôn lớn hơn 0

with open('DAUTHAU.INP', 'r') as inp:

n = int(inp.readline().rstrip('\n'))

for i in range(n):

name = inp.readline().rstrip('\n')

cost = int(inp.readline().rstrip('\n'))

if cost > max:

#nếu giá thầu của name lớn hơn max

max = cost

result = name

#gán max = giá thầu đó và result = name

with open('DAUTHAU.OUT', 'w') as out:

out.write('{}'.format(result))

#in ra result

# MATHUAT

#Xử lí file input và tạo list 2 chiều có các list là tổng giá trị các hàng và các cột sau đó tính tổng của hàng đầu tiên và so sánh số đó với các tổng của các hàng và các cột còn lại

def solve(squarelst):

#hàm kiểm tra xem list 2 chiều squarelst có phải hình vuông ma thuật hay không

total = sum(squarelst[1])

#tổng cố định của hàng đầu tiên

for i in range(1, 8):

if sum(squarelst[i]) != total:

#kiểm tra xem có khác với các hàng còn lại không

return 'not magic'

#nếu khác thì trả về 'not magic'

return 'magic'

#chạy xong vòng lặp mà chưa trả về giá trị nào thì trả về 'magic'

square = []

#list 2 chiều gồm các list con là các giá trị của các hàng và các cột

lst = []

#list để xử lí

with open('MATHUAT.INP', 'r') as inp:

for i in range(4):

lst = list(inp.readline().rstrip('\n').split(' '))

# chuyển từ line trong input thành một list, split(' ') để cắt chuỗi thành một phần tử và thêm vào lst

lst = [int(j) for j in lst]# vì lúc này list chứa các phần tử số ở dạng xâu kí tự nên phải dùng int()để chuyển các phần tử lại thành số

square.append(lst)#thêm lst vừa xử lí vào list square tạo thành list 2 chiều

for i in range(4):

lst = [square[j][i] for j in range(4)]

square.append(lst)

#thêm các list chứa các phần từ ở cột từ 1 tới 4 vào square

with open('MATHUAT.OUT', 'w') as out:

out.write('{}'.format(solve(square)))