#First palindrome string

def pal(a):

for i in a:

if i==i[::-1]:

return i

return None

n=int(input("Enter range: "))

a=[]

for i in range(n):

a.append(input("Enter the string: "))

print(pal(a))

#2

def t(n1,n2):

n=len(n1)

m=len(n2)

res=[]

c,d=0,0

for i in range(n):

for j in range(m):

if n1[i]==n2[j]:

c+=1

break

for i in range(m):

for j in range(n):

if n2[i]==n1[j]:

d+=1

break

res.append([c,d])

return res

n1=[4,3,2,3,1]

n2=[2,2,5,2,3,6]

x=t(n1,n2)

for i in x:

print(i)

#3

def c(n1):

n = len(n1)

x = []

for i in range(n):

for j in range(i + 1, n+1):

s = list(n1[i:j])

x.append(s)

su=0

for i in x:

z=set(i)

su+=len(z)\*\*2

return su

n1 = [1, 3, 1]

print(c(n1))

#4

def r(n1,k):

n=len(n1)

c=0

for i in range(n):

for j in range(i+1,n):

if n1[i]==n1[j] and (i\*j)%k==0:

c+=1

return c

n1=[3,1,2,2,2,1,3]

k=2

print(r(n1,k))

#5

a=[1,2,3,4,5]

print(max(a))

#6

def s(a):

if len(a)<=1:

return a

m=len(a)//2

l=s(a[:m])

r=s(a[m:])

i,j=0,0

b=[]

while i<len(l) and j<len(r):

if l[i]<r[j]:

b.append(l[i])

i+=1

else:

b.append(r[j])

j+=1

while i<len(l):

b.append(l[i])

i+=1

while j<len(r):

b.append(r[j])

j+=1

return b

a=[1, 7, 3, 4, 5]

print(s(a)[-1])

#7

def s(a):

b=[]

for i in a:

if i not in b:

b.append(i)

return b

a=[3,7,3,5,2,5,9,2]

print(s(a))

#8

def bub(a,n):

for i in range(n):

for j in range(n):

if a[j]>a[i]:

x=a[i]

a[i]=a[j]

a[j]=x

return a

a=[5,7,9,4]

n=len(a)

print(bub(a,n))

#9

def bs(a,t):

n=len(a)

l,h=0,n-1

while l<=h:

m=l+h//2

if a[m]==t:

return m

elif a[m]<t:

l=m+1

else:

h=m-1

return None

a=[1,2,3,4,5,6,7,8,9]

t=8

print("Index of Target in array: ",bs(a,t))

#10

def s(a):

if len(a)<=1:

return a

m=len(a)//2

l=s(a[:m])

r=s(a[m:])

i,j=0,0

b=[]

while i<len(l) and j<len(r):

if l[i]<r[j]:

b.append(l[i])

i+=1

else:

b.append(r[j])

j+=1

while i<len(l):

b.append(l[i])

i+=1

while j<len(r):

b.append(r[j])

j+=1

return b

a=[1, 7, 3, 4, 5]

print(s(a))

#11

def f(m, n, N, i, j):

MOD = 10\*\*9 + 7

dp = [[[0] \* n for \_ in range(m)] for \_ in range(N + 1)]

d = [(-1, 0), (1, 0), (0, -1), (0, 1)]

dp[0][i][j] = 1

for k in range(N):

for r in range(m):

for c in range(n):

if dp[k][r][c] > 0:

for dr, dc in d:

nr, nc = r + dr, c + dc

if 0 <= nr < m and 0 <= nc < n:

dp[k+1][nr][nc] = (dp[k+1][nr][nc] + dp[k][r][c]) % MOD

else:

dp[k+1][r][c] = (dp[k+1][r][c] + dp[k][r][c]) % MOD

result = 0

for r in range(m):

for c in range(n):

if r == 0 or r == m-1 or c == 0 or c == n-1:

result = (result + dp[N][r][c]) % MOD

return result

print(f(2, 2, 2, 0, 0))

#12

def k(h):

if len(h) == 0:

return 0

if len(h) == 1:

return h[0]

def ms(arr):

s1, s2 = 0, 0

for x in arr:

s1, s2 = s2, max(s1 + x, s2)

return s2

mf = ms(h[1:])

ml = ms(h[:-1])

return max(mf, ml)

h = list(map(int, input("Enter houses separated by space: ").split()))

print(k(h))

#13

def l(n):

a, b = 1, 1

for i in range(n - 1):

a, b = b, a + b

return b

n = int(input("Enter number of steps: "))

print(l(n))

#14

def mi(m, n):

dp = [[1] \* n for \_ in range(m)]

for i in range(1, m):

for j in range(1, n):

dp[i][j] = dp[i-1][j] + dp[i][j-1]

return dp[-1][-1]

m = int(input("Enter m: "))

n = int(input("Enter n: "))

print(mi(m, n))

#15

def n(s):

r = []

i = 0

while i < len(s):

j = i

while j < len(s) and s[j] == s[i]:

j += 1

if j - i >= 3:

r.append([i, j - 1])

i = j

return r

s = input("Enter string: ")

print(n(s))

#16

def o(b):

m, n = len(b), len(b[0])

d = [[b[i][j] for j in range(n)] for i in range(m)]

for i in range(m):

for j in range(n):

c = 0

for x in range(max(0, i-1), min(m, i+2)):

for y in range(max(0, j-1), min(n, j+2)):

if (x, y) != (i, j) and b[x][y] == 1:

c += 1

if b[i][j] == 1 and (c < 2 or c > 3):

d[i][j] = 0

elif b[i][j] == 0 and c == 3:

d[i][j] = 1

return d

b = []

for \_ in range(int(input("Enter number of rows: "))):

b.append(list(map(int, input().split())))

print(o(b))

#17

def pi(p, r, g):

q = [[0.0] \* (r + 1) for \_ in range(r + 1)]

q[0][0] = p

for i in range(r):

for j in range(i + 1):

o = (q[i][j] - 1.0) / 2.0

if o > 0:

q[i + 1][j] += o

q[i + 1][j + 1] += o

return min(1.0, q[r][g])

p = int(input("Enter poured amount: "))

r = int(input("Enter query row: "))

g = int(input("Enter query glass: "))

print(pi(p, r, g))