Bài 1:

Vào những năm 1 nghìn 7 trăm hồi đó. Nhà toán học Goldbach trong lúc gửi thư tâm sự với nhà toán học Euler có đoán đại là có lẽ trên đời này bất cứ cái số chẵn nào lớn hơn hai thì cũng có thể viết thành tổng của 02 số nguyên tố. Nhưng sau đó cả hai nhà toán học đều không chứng minh hay phủ định phỏng đoán trên và đó trở thành một trong những bài toán chưa có lời giải nổi tiếng nhất đến ngày nay.  
  
Bạn Bình cũng muốn tìm được một trường hợp phủ định của phỏng đoán trên để trở nên nổi tiếng. Bình thấy chỉ cần viết một chương trình tìm cách phân tích một số chẵn thành tổng hai số nguyên tố rồi thử từng số nếu tìm ra được số nào không thể phân tích là chắc chắn Bình sẽ giàu to. Bạn hãy giúp Bình viết chương trình trên  
  
INPUT  
Một số chẵn lớn hơn 2  
  
OUTPUT  
Số lượng cách phân tích input thành tổng hai số nguyên tố

n = int(input())

def ktra(t):

dem = 0

for i in range(1,t+1):

if(t % i == 0):

dem+=1

if(dem == 2):

return 1

else:

return 0

count = 0

a = 1

for a in range(0,int(n/2+1)):

if(ktra(a)==1):

b = n - a

if(ktra(b)==1):

count+=1

print(count)

Bài 2:

Có một vài số nguyên tố khi viết đảo lại nó cũng là số nguyên tố. Ví dụ như 17 hay 71 đều là số nguyên tố. Cho hai số a và b hãy tìm xem có bao nhiêu số x nằm trong đoạn [a, b] sao cho khi viết xuôi hay viết đảo số đó đều là số nguyên tố.  
  
INPUT  
Hai số nguyên dương a, b (a <= b)  
  
OUTPUT  
Số lượng số x

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

import math

def ktra(b):

if b < 2:

return False

elif b == 2:

return True

elif b % 2 == 0:

return False

else:

for d in range(3, 1 + math.floor(math.sqrt(b)), 2):

if b % d == 0:

return False

return True

count = 0

if int(a) <= 12:

if int(a)>=3:

count+=3

if int(a)>=5:

count-=1

if int(a)>=7:

count-=1

for i in range(11,int(b)+1,2):

if(ktra(i)==True) and (ktra(int("".join(reversed(str(i)))))==True):

count+=1

else:

if int(a)%2 != 0:

for i in range(int(a),int(b)+1,2):

if(ktra(i)==True) and (ktra(int("".join(reversed(str(i)))))==True):

count+=1

else:

for i in range(int(a)+1,int(b)+1,2):

if(ktra(i)==True) and (ktra(int("".join(reversed(str(i)))))==True):

count+=1

print(count)

Bài 3:

Số nguyên tố lớn thứ 2  
  
Số nguyên tố là số nguyên lớn hơn 1, chỉ chia hết cho 1 và chính nó.  
  
Yêu cầu: cho số nguyên dương N, hãy chỉ ra số nguyên tố lớn thứ 2 không vượt quá N.  
  
Ví dụ: với N = 10, số nguyên tố lớn thứ 2 không vượt quá 10 là 5  
  
Dữ liệu vào: 1 số nguyên dương N (3 ≤ N ≤ 10^14)

n = int(input())

k = 5

import random

#miller rabin

def snt(n, k):

if n <= 1:

return False

if n == 2 or n == 3:

return True

if n % 2 == 0:

return False

r, m = 0, n - 1

while m % 2 == 0:

r += 1

m //= 2

for \_ in range(k):

a = random.randint(2, n - 2)

x = pow(a, m, n)

if x == 1 or x == n - 1:

continue

for \_ in range(r - 1):

x = pow(x, 2, n)

if x == n - 1:

break

else:

return False

return True

s = []

dem = 0

if n > 5:

for i in range(n,0,-1):

if snt(i,k) == True:

dem+=1

s.append(i)

if dem == 3:

print(s[1])

exit()

else:

print(2)

Bài 4:

Thông thường khi làm việc với con số, người ta quan tâm tới các chữ số theo thứ tự từ đầu tới cuối. Tuy nhiên đôi khi có những ông thầy vừa mập vừa xấu vừa khó lại còn ác ổng hay quan tâm tới những chữ số ở giữa. Chữ số ở giữa theo định nghĩa là nó nằm "ngay chính giữa" của con số. Ví dụ với 532 thì chữ số "3" đứng chính giữa, nhưng trường hợp 5321 thì có tới hai chữ số ở chính giữa nên ta xem số chính giữa trong trường hợp này là "32". Hãy viết chương trình tìm số có chữ số chính giữa là lớn nhất trong một đống số.  
  
INPUT  
Số tự nhiên n (n < 10000), đây là số lượng con số sẽ được nhập vào.  
  
n con số tiếp theo chính là một đống số mà chúng ta quan tâm  
  
OUTPUT  
Con số có chữ số ở giữa là lớn nhất. Nếu có nhiều số cùng số chính giữa, in ra số được nhập vào sau cùng.

h = []

a = []

while True:

try:

l=input()

except EOFError:

break

for i in l.split():

if i!='':

h.append(int(i))

n = h[0]

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

k = str(h[i])

if len(k) % 2 == 0:

newstr = int(k[(len(k)//2)-1:(len(k)//2)+1])

a.append(newstr)

else:

newstr = int((k[len(k)//2]))

a.append(newstr)

maxw = max(a)

for j in range(len(a)-1,-1,-1):

if(a[j] == maxw):

index = j

Break

print(h[index+1])

Bài 5:

Các nhà địa chất có nhiệm vụ khảo sát tìm khoáng sản ở một vùng đài nguyên mênh mông.  
Thiết bị và nhu yếu phẩm được chở tới trại Trung tâm tại điểm có tọa độ (0, 0) trên bản đồ.  
Nhóm khảo sát đi thăm dò từng điểm có tọa nguyên trên bản đồ. Nhóm di chuyển từ điểm có  
tọa độ nguyên này tới điểm có tọa độ nguyên khác và luôn đi theo một trong các hướng  
“Đông”(E), “Tây” (W), “Nam” (S) hoặc “Bắc” (N). Việc đổi hướng có thể được thực hiện tại  
các điểm có tọa độ nguyên. Khoảng cách bằng một đơn vị tọa độ được gọi là bước. Đường đi  
của họ được thiết bị tự động mang theo báo về dưới dạng xâu ghi nhận từng nhóm số bước di  
chuyển và hướng. Ví dụ, dòng thông báo “7N5E2S3E” cho biết nhóm đã đi 7 bước theo hướng  
nam, sau đó đi 5 bước về hướng đông, đi tiếp theo hướng bắc 2 bước rồi đi theo hướng đông 3  
bước.  
Có thể tại một nơi nào đó nhóm tìm thấy những mẫu vật đặc biệt, cần phân tích ngay để có  
hành động phù hợp tiếp theo và yêu cầu Trung tâm gửi thiết bị bay tới lấy về. Các thiết bị bay  
mang theo chỉ được lập trình cho đường bay theo các hướng E, W, S và N. Khi nạp thông báo  
về hành trình của nhóm khảo sát vào bộ nhớ, các microchips trong máy bay sẽ tính toán, tìm  
đường đi có xâu (biểu diễn dưới dạng của thiết bị ghi nhận đường đi đã nói ở trên) ngắn nhất.  
Có nhiều cách ngắn nhất để đi tới đích. Hãy xác định một trong số các xâu biểu diễn đường đi  
do thiết bị bay tạo ra. Đảm bảo đích cần tới không trùng với gốc tọa độ.  
Dữ liệu: Vào từ file văn bản ROUTE.INP gồm một dòng chứa xâu độ dài không quá 250 xác  
định hành trình của nhóm khảo sát. Các hệ số có giá trị nguyên dương và không vượt quá 107  
.  
Kết quả: Đưa ra file văn bản ROUTE.OUT xâu do thiết bị bay tạo ra.

location = list(map(str,input().split()))

a = []

b = []

c = []

for i in range(0,len(location)):

if int(location[i]) < 0:

b.append(i)

a.append((location[i]))

n = 0

m = 0

while i < len(location) and m < len(b):

v = int(''.join(location[n:b[m]]))

c.append(v)

n = b[m] + 1

m += 1

x,y = 0,0

for j in range(0,len(c)):

if a[j] == '-1':

y +=c[j]

if a[j] == '-2':

x-=c[j]

if a[j] == '-3':

y-=c[j]

if a[j] == '-4':

x+=c[j]

if x > 0 and y > 0:

print(str(y),'-1',str(x),'-4')

elif x > 0 and y < 0:

print(str(-y),'-3',str(x),'-4')

elif x < 0 and y > 0:

print(str(y),'-1',str(-x),'-2')

elif x < 0 and y < 0:

print(str(-y),'-3',str(-x),'-2')

elif x == 0 and y < 0:

print(str(-y) + '-3')

elif x == 0 and y > 0:

print(str(y),'-1')

elif x < 0 and y == 0:

print(print(str(-x),'-2'))

elif x > 0 and y == 0:

print(print(str(x),'-4'))

Bài 6:

Bạn Bình đi mua tablet. Bình là người rất cầu toàn, Bình muốn cả chiều dài, chiều rộng của tablet phải là số nguyên. Giả sử Bình muốn kích thước của tablet cũng phải là một số nguyên n thì có bao nhiêu lựa chọn kích thước màn hình tablet Bình có thể cân nhắc?  
  
INPUT  
Một số nguyên dương n  
  
OUTPUT  
Số lượng lựa chọn có thể có của Bình

n = int(input())

a = n - 1

dem = 0

import math

if n > 100000:

while n > 0:

b = math.sqrt(n\*n - a\*a)

if b == int(b) and a >= int(b):

dem+=1

a-=1

elif a < int(b):

print(dem)

exit()

if a % 2 == 0:

a-=2

else:

a-=1

else:

while n > 0:

b = math.sqrt(n\*n - a\*a)

if b == int(b) and a >= int(b):

dem+=1

elif a < int(b):

print(dem)

exit()

a-=1