TÀI LIỆU BỒI DƯỠNG HỌC SINH GIỎI TIN HỌC BẬC THCS

CHUYÊN ĐỀ:
GIẢI ĐỀ THI BẰNG NNLT PYTHON 3.x

Tác giả: Nguyễn Tấn Phong
Trường THCS Đồng Nai, huyện Cát Tiên, tỉnh Lâm Đồng

1. ĐỀ THI HSG HUYỆN CÁT TIÊN – NĂM HỌC 2020 – 2021

Câu 1: Tính tổng

Viết chương trình tính tổng:

$$S=1+\frac{1}{2^2}+\frac{1}{3^2}+...+\frac{1}{n^2}$$

* Dữ liệu vào từ file: **TONG.INP**

- Dòng đầu tiên ghi số tự nhiên n.

* Kết quả ra file: **TONG.OUT**

- Dòng đầu tiên ghi số thực là tổng S, làm tròn đến hai chữ số thập phân.

Ví dụ:

TONG.INP	TONG.OUT
3	1,361
4	1,424

Phân tích: Kết quả file OUT dùng dấu phẩy (,) để ngăn cách phần nguyên và phần thập phân và làm tròn 3 chữ số thập phân.

Đây được xem là một ***"bẩy"*** kĩ năng đọc và phân tích đề bài (còn gọi là "bẩy" cá tính thí sinh).

Code tham khảo:

```
1  #Đọc dữ liệu vào từ file INP
2  fi = open('tong.inp')
3  n = int(fi.read())
4  fi.close()
5  #Thuật toán
6  tong = 0
7  for i in range(1,n+1):
8          tong = tong + 1/(i**2)
9  #Xử lý dấu thập phân
10 tong = str(round(tong,3)) #làm tròn 3 chữ số thập phân
11 tong.replace('.',',',1) #thay dấu chấm . bởi dấu phẩy ,
12 #Ghi vào file OUT
13 fo = open('tong.out','w')
14 fo.write(tong)
15 fo.close()
```

☼Tìm hiểu: .replace(); round(); str()

Câu 2: Tìm số tự mãn trong dãy

Qui ước: Số tự mãn bậc 3 là những số bằng tổng lập phương các chữ số của nó. Ví dụ: Số 153 là số tự mãn vì 153 = $1^3 + 5^3 + 3^3$.

Cho dãy gồm N phần tử là số nguyên dương A_1, A_2, ... , A_n; ($0<N<=10^3$; $0<A_i<=10^6$). Viết chương trình tìm những số tự mãn trong dãy số đã cho?

* Dữ liệu vào từ file: **SOTUMAN.INP**

- Dòng đầu tiên chứa số nguyên dương N.
- Dòng thứ hai chứa N số nguyên dương, các số cách nhau một khoảng trắng.

* Kết quả ra file: **SOTUMAN.OUT**

- Dòng đầu tiên ghi các số tự mãn tìm được theo thứ tự tăng dần, các số cách nhau một khoảng trắng.

Ví dụ:

SOTUMAN.INP	SOTUMAN.OUT
5 6 371 18 153 28	153 371

Phân tích: Kiểm tra lần lượt các phần tử trong dãy đã cho có là số tự mãn bậc 3 hay không? Nếu có thì lưu số đó ra một mảng kết quả.

Thuật toán: xây dựng hàm kiểm tra số tự mãn bậc 3 hoặc tính trực tiếp.

Code tham khảo:

Cách 1: Xây dựng hàm kiểm tra số tự mãn

```
#Đọc dữ liệu vào từ file INP
fi = open('SOTUMAN.INP')
n = int(fi.readline())
a = fi.readline().split()
for i in range(n): a[i] = int(a[i]) #chuyển mảng a về số nguyên
fi.close()
#Xây dựng hàm kiểm tra số tự mãn (True/False)
def sotuman(m):
    tlp=0
    for i in str(m): tlp = tlp + int(i)**3
    if tlp==m: return True
    return False
#Thuật toán xử lý
b=[]
for i in a:
    if sotuman(i): b.append(i)
b.sort() #Sắp xếp mảng b tăng dần
#Ghi kết quả vào file OUT
fo = open('SOTUMAN.OUT','w')
for i in b: fo.write(str(i)+' ')
```

```
21 fo.close()
```

Cách 2: Tối ưu cách 1 bằng kĩ thuật sử dụng List Comprehension

```
 1 #đọc dữ liệu vào
 2 fi = open('SOTUMAN.INP')
 3 n = int(fi.readline())
 4 a = list(map(int,fi.readline().split()))
 5 fi.close()
 6 #Thuật toán với List Comprehension
 7 b = []
 8 for i in a:
 9     if i == sum([int(x)**3 for x in str(i)]): b.append(i)
10 b.sort() #Sắp xếp mảng b tăng dần
11 #ghi file OUT
12 fo = open('SOTUMAN.OUT','w')
13 for i in b: fo.write(str(i)+' ')
14 fo.close()
```

☼Tìm hiểu: - Thay 2 dòng lệnh 4, 5 bằng dòng lệnh dùng hàm list(map()):

```
a = list(map(int,fi.readline().split()))
```

- Bằng kĩ thuật dùng List Comprehension ta sửa lại hàm sotuman(m) của cách 1 như sau:

```
def sotuman(m):
    return m==sum([int(i)**3 for i in str(m)])
```

Câu 3: Xếp hàng

Trong giờ sinh hoạt tập thể, lớp 9A có **n** học sinh (n <=45) xếp thành hàng dọc. Mỗi học sinh có chiều cao a[i]. Em hãy viết chương trình đếm số bạn có chiều cao bằng nhau nhiều nhất.

* Dữ liệu vào từ file: **XEPHANG.INP**

- Dòng thứ nhất chứa số tự nhiên n.
- Dòng thứ hai gồm n số tự nhiên a[i], mỗi số ứng với chiều cao của từng bạn (đơn vị cm), các số cách nhau một khoảng trắng.

* Kết quả ra file: **XEPHANG.OUT**

- Gồm một dòng ghi 2 số tự nhiên. Số thứ nhất ghi tổng số bạn có chiều cao bằng nhau nhiều nhất, số thứ 2 ghi chiều cao tương ứng, các số cách nhau một khoảng trắng.

Ví dụ:

XEPHANG.INP	XEPHANG.OUT
10 160 158 158 160 159 158 159 160 158 161	4 158

Code tham khảo:

Cách 1: Lọc phần tử không trùng nhau đôi một bằng List Comprehension

```
fi = open('xephang.inp')
n = int(fi.readline())
a = list(map(int,fi.readline().split()))
fi.close()
#Thuật toán
b=[]
[b.append(x) for x in a if x not in b] #lọc lấy các số khác nhau trong a
dem = 0
chieucao = 0
for i in b:
    if dem < a.count(i):
        dem = a.count(i)
        chieucao=i
#ghi file OUT
fo = open('xephang.out','w')
fo.write(str(dem)+' ' + str(chieucao))
fo.close()
```

☼Tìm hiểu: phương thức .count()

Cách 2: Dùng hàm có sẵn max(set(),key = .count)

```
fi = open('xephang.inp')
n = int(fi.readline())
a = list(map(int,fi.readline().split()))
fi.close()
#Thuật toán
chieucao = max(set(a), key = a.count)
dem=a.count(chieucao)
#ghi file OUT
fo = open('xephang.out','w')
fo.write(str(dem)+' ' + str(chieucao))
fo.close()
```

☼Tìm hiểu: set(), max(set(), key = .count)

Câu 4: Kangaroo

Một chú Kangaroo muốn đi thăm một người bạn trên cùng tuyến đường cách đó một khoảng n (đơn vị dm). Kangaroo chỉ có hai cách di chuyển, một là nhảy ngắn a (đơn vị dm), hai là nhảy dài b (đơn vị dm). Hỏi chú Kangaroo cần nhảy ít nhất bao nhiêu bước nhảy để đến được nhà người bạn (phải nhảy vừa đủ, không nhảy quá nhà bạn).

**** Dữ liệu vào từ file:*** **KANGAROO.INP**

- Gồm ba số nguyên dương n, a, b. Các số cách nhau một khoảng trắng (1 <= n <= 10^9, 1 <= a < b <= 20).

**** Kết quả ra file:*** **KANGAROO.OUT**

- Ghi tổng số bước nhảy ít nhất của chú Kangaroo.

Ví dụ:

KANGAROO.INP	KANGAROO.OUT
21 2 5	6

Phân tích: file OUT không nêu ra trường hợp Kangaroo nhảy không được, như vậy dữ liệu được cho hoàn toàn phù hợp (chắc chắn có đáp số đúng).
Lý thuyết số: Nếu gọi x, y lần lượt là số bước nhảy ngắn và số bước nhảy dài thì ta sẽ có phương trình nghiệm nguyên: n = ax + by.
Code tham khảo:
Cách 1: Dùng khái niệm lát cắt trên **iterrator**

```
#Đọc dữ liệu từ file INP
fi = open('kangaroo.inp')
n,a,b = map(int,fi.readline().split())
fi.close()
#Thuật toán với lát cắt trên list
c = [b]*(n//b) + [a]*(n//a) #khai báo mảng c
sobuoc = -1
for i in range(len(c)-1):
    for j in range(i+1,len(c)):
        if sum(c[i:j])==n:
            sobuoc = len(c[i:j])
            break
    if sobuoc != -1: break
#Ghi file OUT
fo = open('kangaroo.out','w')
fo.write(str(sobuoc))
fo.close()
```

Cách 2: Áp dụng tính chất chia hết của số nguyên

```
#Đọc dữ liệu từ file INP
fi = open('kangaroo.inp')
n,a,b = map(int,fi.readline().split())
fi.close()
#Thuật toán tính chất chia hết số nguyên
sobuoc=-1
for i in range(n//b,-1,-1):
   if (n-i*b)%a==0:
      sobuoc = i + ((n-i*b)//a)
      break
#Ghi file OUT
```

```
12 fo = open('kangaroo.out','w')
13 fo.write(str(sobuoc))
14 fo.close()
```

2. ĐỀ THI HSG HUYỆN ĐẠ TẺH – NĂM HỌC 2020 – 2021

Câu 1: ***(6 điểm)*** **Ước chung lớn nhất**

Cho hai số tự nhiên N, M ($1 < N, M < 10^9$). Viết chương trình tìm ước chung lớn nhất của hai số N và M?

Dữ liệu vào từ file: **UCLN.INP**

- Dòng đầu tiên ghi hai số N và M, cách nhau một khoảng trắng.

Kết quả ra file: **UCLN.OUT**

- Dòng đầu tiên ghi ước chung lớn nhất tìm được.

Ví dụ:

UCLN.INP	UCLN.OUT
2 4	2
11 17	1

Phân tích:

Code tham khảo:

```
1  #Đọc file INP
2  fi = open('UCLN.INP')
3  n,m = map(int,fi.readline().split())
4  fi.close()
5  #Hàm tìm UCLN
6  def ucln(a,b):
7      while a!=b:
8          if a > b: a = a - b
9          else: b = b - a
10     return a
11 #ghi file OUT
12 fo = open('UCLN.OUT','w')
13 print(ucln(m,n),file=fo)
14 fo.close()
```

Câu 2: ***(7 điểm)*** **Đếm các số nguyên tố**

Cho dãy số nguyên gồm N phần tử A_i ($0 < N < 10^3$, $0 < A_i < 10^6$). Viết chương trình đếm xem trong dãy số đã cho có bao nhiêu phần tử là số nguyên tố?

Dữ liệu vào từ file: **NGUYENTO.INP**

- Dòng đầu tiên ghi hai số N
- Dòng thứ hai ghi N số nguyên, các số cách nhau một khoảng trắng.

Kết quả ra file: **NGUYENTO.OUT**

- Dòng đầu tiên ghi kết quả đếm được.

Ví dụ:

NGUYENTO.INP	NGUYENTO.OUT

2 2 4	1
5 7 13 27 13 29	4

Phân tích:

Code tham khảo:

Cách 1: Dùng thuật toán tìm kiếm trên dãy số

```
1  # Ham kiem tra so nguyen to
2  def ktnt(a):
3      if(a<2):  return False
4      for i in range(2,a//2+1):
5          if a%i==0: return False
6      return True
7  #đọc file INP
8  fi = open('NGUYENTO.INP')
9  n = int(fi.readline())
10 dayso = list(map(int,fi.readline().split()))
11 fi.close()
12 #Thuật toán tìm kiếm
13 dem=0
14 for i in dayso:
15       if ktnt(i): dem += 1
16 #ghi file out
17 fo = open('NGUYENTO.OUT','w')
18 fo.write(str(dem))
19 fo.close()
```

☼Tìm hiểu: Với cách viết câu lệnh dòng 14, 15 thì n được cho không dùng đến.

Cách 2: Dùng List Comprehension

```
1  # Ham kiem tra so nguyen to trong python
2  def ktnt(a):
3      if(a<2):  return False
4      for i in range(2,a//2+1):
5          if a%i==0: return False
6      return True
7  #đọc file INP
8  fi = open('NGUYENTO.INP')
9  n = int(fi.readline())
10 dayso = list(map(int,fi.readline().split()))
11 fi.close()
12 # thuật toán
13 b = [x for x in dayso if ktnt(x)]
14 #ghi file out
15 fo = open('NGUYENTO.OUT','w')
```

```
16  fo.write(str(len(b)))
17  fo.close()
```

☼Tìm hiểu: hàm len()

Câu 3: ***(7 điểm)*** **Dãy theo quy luật**

Quy ước: Ứng với mỗi số tự nhiên x ($0 < x < 10^6$), ta có số tự nhiên f(x) bằng tổng bình phương các chữ số của x.

Chẳng hạn: x = 12 thì $f(x) = 1^2 + 2^2 = 5$.

Từ x ta xây dựng dãy X_n theo quy ước như sau:

$X_1 = x ; X_2 = f(X_1) ; X_3 = f(X_2) ; ...; X_i = f(X_{i-1})$ (với $1 <= i <= n < 10^2$)

Viết chương trình in ra dãy (X_n)?

Dữ liệu vào từ file: **DAYXN.INP**

- Dòng đầu tiên ghi số tự nhiên x và n, cách nhau một khoảng trắng.

Kết quả ra file: **DAYXN. OUT**

- Dòng đầu tiên ghi n phần tử đầu tiên của dãy, các số cách nhau một khoảng trắng.

Ví dụ:

DAYXN.INP	DAYXN. OUT
12 4	12 5 25 29

Phân tích:

Code tham khảo:

```
1   #Hàm tính tổng bình phương các chữ số
2   def f(x):
3       return sum([int(i)**2 for i in str(x)])
4   #Đọc file INP
5   fi=open('DAYXN.INP')
6   x,n=map(int,fi.readline().split())
7   fi.close()
8   #Thuật toán_Ghi file OUT
9   fo=open('DAYXN.OUT','w')
10  for i in range(n):
11      fo.write(str(x)+' ')
12      x=f(x)
13  fo.close()
```

☼Tìm hiểu:

3. ĐỀ THI HSG HUYỆN ĐẠ HOAI – NĂM HỌC 2020 – 2021

Câu 1: ***(5 điểm)*** **Tính tổng**

Viết chương trình tính và xuất ra màn hình tổng sau:

$$S=\frac{1}{1.3}+\frac{1}{2.4}+\frac{1}{3.5}+...+\frac{1}{a.(a+2)}$$

Với a là số nguyên dương nhập từ bàn phím. Kết quả làm tròn 1 chữ số thập phân.

Ví dụ:

Dữ liệu vào (a)	Kết quả
1	0.3
2	0.5

Phân tích: Bài này đề yêu cầu nhập/xuất chuẩn, không yêu cầu dữ liệu vào ra từ file.

Code tham khảo:

```
1  a = int(input('- Nhập a = '))
2  s = 0
3  for i in range(1,a+1): s += 1/(i*(i+2))
4  print('%.1f'%s)
```

☼Tìm hiểu: - Định dạng số thập phân trong python theo chuẩn '%.1f'%

- Khi a là số lớn (từ 7 chữ số trở lên) thì thời gian chạy của vòng lặp (for, while) là rất lâu, đây được xem là một bài tập có ***"bẫy"*** về thời gian chạy và kiến thức toán học.

Kiến thức số học: Các tổng mà mỗi số hạng có tử số nhỏ hơn mẫu số (hay số hạng của tổng nhỏ hơn 1) thì tổng đó sẽ hội tụ về gần một giá trị cố định nào đó. Trong câu trên, với a>=10 thì S làm tròn 1 chữ số thập phân sẽ luôn bằng 0.7; Do vậy, để tránh "bẫy" thời gian chạy với số lớn ta sửa lại đoạn code như sau:

```
1  a = int(input('- Nhập a = '))
2  if a>=10: s = 0.7
3  else:
4        for i in range(1,a+1): s += 1/(i*(i+2))
5  print('%.1f'%s)
```

Câu 2: (5 điểm) Tổng chẵn

Cho một dãy số nguyên có n số (0<n<10000). Viết chương trình xuất ra các số chẵn có trong dãy và tổng các chữ số chẵn đó.

* Dữ liệu vào từ file: TONGCHAN.INP

- Dòng 1: Ghi số tữ nhiên n.
- Dòng 2: Ghi n số nguyên của dãy, mỗi số cách nhau một khoảng trắng.

* Kết quả ra file: TONGCHAN.OUT

- Dòng 1: Ghi các số chẵn có trong dãy, các số cách nhau một khoảng trắng.
- Dòng 2: Ghi tổng các số chẵn có trong dãy.

Ví dụ:

TONGCHAN.INP	TONGCHAN. OUT
5 2 3 6 8 7	2 6 8 16
7 8 9 6 12 20 25 45	8 6 12 20 46

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi=open('TONGCHAN.INP')
3  n=int(fi.readline())
4  a=list(map(int,fi.readline().split()))
5  fi.close()
6  #Thuật toán tìm kiếm, tính tổng và ghi file OUT
7  fo=open('TONGCHAN.OUT','w')
8  s=0
9  for i in a:
10     if i%2==0:
11         fo.write(str(i)+' ')
12         s=s+i
13 fo.write('\n')
14 fo.write(str(s))
15 fo.close()
```

Có thể dùng List Comprehension để thay thế đoạn code từ dòng 8 đến dòng 12 như sau:

```
b = [x for x in a if x%2==0]
for i in b: fo.write(str(i)+' ')
fo.write('\n')
fo.write(str(sum(b)))
```

Câu 3: (5 điểm) Số thân thiện

Số nguyên tố là số lớn hơn 1 và chỉ chia hết cho 1 và chính nó. Hai số tự nhiên được gọi là cặp số thân thiện với nhau nếu có chung ước nguyên tố lớn nhất.

Ví dụ số 14 và 21 là hai số thân thiện với nhau vì cùng có ước nguyên tố lớn nhất là 7.

Viết chương trình tìm 5 số thân thiện với số tự nhiên a ($1<a<10^5$).

* Dữ liệu vào từ file: **THANTHIEN.INP**

- Dòng 1: Ghi số tự nhiên a.

Kết quả ra file: **THANTHIEN.OUT**

- Dòng 1: Ghi 5 số thân thiện của số a, các số cách nhau một khoảng trắng.

Ví dụ:

THANTHIEN.INP	THANTHIEN.OUT
25	5 10 15 20 25
14	7 14 21 28 35

Phân tích:

-*"Hai số tự nhiên được gọi là cặp số thân thiện với nhau nếu có chung ước nguyên tố lớn nhất"* hiểu đây là một quy ước, không phải là khái niệm của **lý thuyết số**.

- Qua 2 ví dụ ta thấy kết quả file OUT là ghi 5 số thân thiện từ bé đến lớn bao gồm cả chính số. Thoạt nhìn có vẻ rất đơn giản! Nhưng bài toán này được xem là một dạng ***"bẩy"*** về kiến thức toán học!

Chúng ta sẽ phân tích và tìm vấn đề nhé!

Dựa vào quy ước và file OUT, ta dễ dàng xây dựng thuật toán chỉ đơn giản như sau: tìm ước nguyên tố lớn nhất của a, sau đó ghi vào file OUT lần lượt tích của ước nguyên tố lớn nhất với các số từ 1 đến 5. Như đoạn code dưới đây!

Code tham khảo:

```
#Hàm kiểm tra số nguyên tố
def ktnt(n):
    if n<2: return False
    for i in range(2,n//2+1):
        if n%i==0: return False
    return True
#Đọc dữ liệu file INP
fi=open('THANTHIEN.INP')
a=int(fi.readline())
fi.close()
#Tìm ước nguyên tố lớn nhất của a
unt = 1
for i in range(a,1,-1):
    if a%i==0 and ktnt(i): unt=i;break
#Ghi file OUT
fo=open('THANTHIEN.OUT','w')
for i in range(1,6): fo.write(str(i*unt)+' ')
fo.close()
```

Bây giờ ta xét với bộ dữ liệu:

THANTHIEN.INP	**THANTHIEN.OUT**
2	2 4 6 8 10

Ta thấy, số 6 và 10 có ước nguyên tố lớn nhất là 3 và 5; như vậy số 6, 10 và 2 không phải là cặp số thân thiết theo quy ước. Hay nói cách khác thì bộ dữ liệu được tạo ra từ code trên là sai.

Bây giờ ta cải tiến lại code như sau:

Cách 1: Kĩ thuật tìm kiếm với hàm tự tạo

```
#Hàm kiểm tra số nguyên tố
def ktnt(n):
    if n<2: return False
    for i in range(2,n//2+1):
        if n%i==0: return False
```

```
    return True
#Hàm tìm ước nguyên tố lớn nhất của một số
def uocnt(n):
    if ktnt(n): return n
    for i in range(n//2+1,1,-1):
        if n%i==0 and ktnt(i): return i
#Đọc dữ liệu file INP
fi=open('THANTHIEN.INP')
a=int(fi.readline())
fi.close()
#Tìm ước nguyên tố lớn nhất của a
unt = uocnt(a)
dem=1
i = 1
#Ghi file OUT
fo=open('THANTHIEN.OUT','w')
fo.write(str(unt)+' ')
while dem<5:
    i = i + 1
    if uocnt(unt*i)==unt:
        fo.write(str(unt*i)+' ')
        dem = dem + 1
fo.close()
```

Bây giờ ta kiểm chứng lại với các bộ dữ liệu:

THANTHIEN.INP	**THANTHIEN.OUT**
25	5 10 15 20 25
14	7 14 21 28 35
2	2 4 8 16 32

Cách 2: Cải tiến thuật toán với List Comprehension

```
#Hàm kiểm tra số nguyên tố
def ktnt(n):
    if n<2: return False
    for i in range(2,n//2+1):
        if n%i==0: return False
    return True
#Hàm tìm ước nguyên tố lớn nhất của một số
def uocnt(n):
    if ktnt(n): return n
    for i in range(n//2+1,1,-1):
        if n%i==0 and ktnt(i): return i
#Đọc dữ liệu file INP
fi=open('THANTHIEN.INP')
a=int(fi.readline())
fi.close()
```

```
#Tìm ước nguyên tố lớn nhất của a
unt = uocnt(a)
b=[]
[b.append(unt*i) for i in range(100) if (uocnt(unt*i)==unt) and len(b)<5]
#Ghi file OUT
fo=open('THANTHIEN.OUT','w')
for i in b:
    fo.write(str(i)+' ' )
fo.close()
```

Bài 4: (5.0 điểm) Phần tử yên ngựa

Một bảng A gồm m hàng và n cột được gọi là một ma trận m×n (m≠0, n≠0). Phần tử tại hàng thứ i cột thứ j được ký hiệu là A[i,j] hoặc A[i][j]. Phần tử A[i,j] được gọi là phần tử yên ngựa nếu nó bé nhất trong hàng i và lớn nhất trong cột j.

Viết chương trình các tìm phần tử yên ngựa (nếu có) của ma trận A có m hàng n cột.

* Dữ liệu vào từ file: **PTYN.INP**

- Dòng đầu ghi hai số tự nhiên m, n, mỗi số cách nhau 1 khoảng trắng.
- m dòng tiếp theo, mỗi dòng ghi n số nguyên là các phần tử của ma trận ma trận A, mỗi số cách nhau 1 khoảng trắng.

Dữ liệu ghi ra file: **PTYN.OUT**

- Ghi các cặp số (i,j) là chỉ số hàng, cột của phần tử A[i,j] nếu A[i,j] là phần tử yên ngựa. Mỗi cặp số ghi trên một dòng. Nếu không có phần tử yên ngựa thì ghi "khong co".

Ví dụ:

PTYN.INP	PTYN.OUT
3 3 1 2 3 4 5 6 7 8 9	(3,1)
3 3 15 10 5 55 4 6 76 1 2	khong co

Phân tích:

Code tham khảo:

Cách 1: Tìm kiếm trên ma trận

```
#Đọc dữ liệu từ file INP
fi=open('PTYN.INP')
m,n=map(int,fi.readline().split())#m = hàng, n = cột
a=[]
for i in fi: a.append(list(map(int,i.split())))
fi.close()
```

```
#lấy min hàng đưa vào mảng 1 chiều
min_h=[0]*m
for j,i in enumerate(a):
    min_h[j] = min(i)
#lấy max cột đưa vào mảng 1 chiều
max_c=[0]*n
for i in range(n):
    cot=[0]*n
    for j in range(m):
        cot[j] = a[j][i]
    max_c[i]=max(cot)
#duyệt so sánh, nếu min_h[i]==max_c[j] thì xuất i+1,j+1
fo=open('PTYN.OUT','w')
kt = False
for i in range(m): #min_h
    for j in range(n): #max_c
        if min_h[i]==max_c[j]:
            fo.write('('+str(i+1)+','+str(j+1)+')\n')
            kt = True
#trường hợp không có
if not kt: fo.write('khong co')
fo.close()
```

☼Tìm hiểu: - Hàm enumerate()

- Đọc ma trận vào mảng a và chuyển các phần tử thành int:

```
a=[]
for i in fi: a.append(list(map(int,i.split())))
```

Cách 2: Cải tiến cách 1 bằng kĩ thuật List Comprehension

```
#Đọc dữ liệu từ file INP
fi=open('PTYN.INP')
m,n=map(int,fi.readline().split())#m = hàng, n = cột
a=[]
for i in fi: a.append(list(map(int,i.split())))
fi.close()
#lấy min hàng đưa vào mảng 1 chiều
min_h=[min(i) for i in a]
#lấy max cột đưa vào mảng 1 chiều
_a=[list(i) for i in zip(*a)] #chuyển vị ma trận a thành _a
max_c=[max(i) for i in _a]
#duyệt so sánh, nếu min_h[i]==max_c[j] thì xuất i+1,j+1
fo=open('PTYN.OUT','w')
kt = False
for i in range(m): #min_h
    for j in range(n): #max_c
```

```
17          if min_h[i]==max_c[j]:
18              fo.write('('+str(i+1)+','+str(j+1)+')\n')
19              kt = True
20  #trường hợp không có
21  if not kt: fo.write('khong co')
22  fo.close()
```

☼Tìm hiểu: - hàm zip(*a)

4. ĐỀ THI HSG HUYỆN BẢO LỘC – NĂM HỌC 2020 – 2021

Câu 1: ***(5 điểm)*** **Tính tổng**

Cho ba số nguyên dương A, B và K ($0 < A < B < 10^6$, $1 < K < 10^3$). Viết chương trình tính tổng tất cả các số M ($A < M < B$), sao cho M chia hết cho K.

Dữ liệu vào từ file: **TONG.INP**

- Dòng đầu tiên ghi ba số nguyên A, B và K, cách nhau một khoảng trắng.

Kết quả ra file: **TONG.OUT**

- Dòng đầu tiên ghi kết quả tính được.

Ví dụ:

TONG.INP	TONG.OUT
2 8 2	10
3 4 3	0

Phân tích:

Code tham khảo:

```
1   #Đọc dữ liệu từ file INP
2   fi = open('TONG.INP')
3   a,b,k=map(int,fi.readline().split())
4   fi.close()
5   #Thuật toán
6   tong=0
7   for m in range(a+1,b):
8       if m%k==0: tong = tong+m
9   #Ghi file OUT
10  fo = open('TONG.OUT','w')
11  fo.write(str(tong))
12  fo.close()
```

☼Tìm hiểu:

Cải tiến cách 1 bằng kĩ thuật List Comprehension, bằng cách thay các dòng lệnh 6, 7, 8 bằng dòng lệnh sau:

tong = sum([x for x in range(a+1,b) if x%k==0])

Câu 2: ***(5 điểm)*** **Tìm số đường khép kín**

Ta qui ước:

+ Các chữ số: 1, 2, 3, 5, 7 có 0 đường khép kín.
+ Các chữ số: 0, 4, 6, 9 có 1 đường khép kín;

+ Chữ số: 8 có 2 đường khép kín;

Viết chương trình tính tổng số đường khép kín của một số nguyên dương N ($N < 10^{10}$).

Dữ liệu vào từ file: **KHEPKIN.INP**

- Dòng đầu tiên ghi số N.

Kết quả ra file: **KHEPKIN.OUT**

- Dòng đầu tiên ghi tổng số đường khép kín tính được.

Ví dụ:

KHEPKIN.INP	KHEPKIN.OUT
2369524	3

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi = open('KHEPKIN.INP')
3  n = int(fi.read())
4  fi.close()
5  #Thuật toán
6  dkk = 0
7  for i in str(n):
8      if i in '0469': dkk = dkk + 1
9      elif i=='8': dkk = dkk + 2
10 #Ghi file OUT
11 fo = open('KHEPKIN.OUT','w')
12 fo.write(str(dkk))
13 fo.close()
```

☼ Tìm hiểu:

Câu 3: ***(5 điểm)*** **Xóa ký tự trong xâu**

Cho xâu có N ký tự chữ số ($0 < N < 255$). Viết chương trình xóa đi K ký tự ($0 < K < N$) để xâu còn lại biểu diễn một số bé nhất.

Dữ liệu vào từ file: **XAU.INP**

- Dòng đầu tiên chứa xâu có N ký tự chữ số, các ký tự chữ số viết liền nhau.
- Dòng thứ hai chứa số K.

Kết quả ra file: **XAU.OUT**

- Dòng đầu tiên ghi xâu ký tự tìm được theo yêu cầu.

Ví dụ:

XAU.INP	XAU.OUT
65278934 3	27834

Phân tích:

Code tham khảo:

```
1  fi = open('XAU.INP')
2  s = list(fi.readline())
3  k = int(fi.readline())
```

```
4   fi.close()
5   #Thuật toán
6   for i in range(k):
7       j=0
8       while (j<len(s)) and (s[j]<=s[j+1]): j += 1
9       del(s[j])
10  #Ghi file OUT
11  fo = open('XAU.OUT','w')
12  fo.write(''.join(s))
13  fo.close()
```

☼ Tìm hiểu:

- Hàm del(), ''.join()

- Trong đề bài cho "xâu có N ký tự chữ số" nhưng tại dòng lệnh 2: s=list(fi.readline()) là đọc từng chữ số và gán vào một mảng một chiều s, thay vì lệnh đọc s=fi.readline(). Vì trong NNLT python qui định chuỗi str là dữ liệu có tính "bất biến", do vậy không có hàm hay phương thức nào cho phép xóa một ký tự bất kỳ trong chuỗi s. Vì vậy, ta phải đọc vào mảng một chiều s để áp dụng hàm del().

- Tại dòng lệnh 8: while (j<len(s)) and (s[j]<=s[j+1]): j += 1 nếu thay đổi dấu "<=" thành dấu ">=" thì bài toán trở thành là tìm "số lớn nhất".

Câu 4: ***(5 điểm)*** **Bộ ba số**

Cho dãy N ($1 <= N <= 10^3$) số nguyên dương A_1, A_2, ..., A_N ($A_i <= 10^3$). Với bộ ba chỉ số i, j và k ($1 <= i < j < k <= N$), hãy tìm giá trị $\mathbf{S = 2A_i - 3A_j + 5A_k}$ sao cho S đạt giá trị lớn nhất.

Dữ liệu vào từ file: **BOBASO.INP**

- Dòng đầu tiên chứa số N

- Dòng thứ hai chứa N số nguyên dương A_1, A_2, ..., A_N; các số cách nhau một khoảng trắng.

Kết quả ra file: **BOBASO.OUT**

- Dòng đầu tiên ghi giá trị của S tìm được.

Ví dụ:

BOBASO.INP	**BOBASO.OUT**
3 3 2 1	5
7 3 5 2 6 4 5 7	39

Phân tích:

Code tham khảo:

```
1   #Đọc dữ liệu từ file INP
2   fi=open('BOBASO.INP')
3   n=int(fi.readline())
4   a=list(map(int,fi.readline().split()))
5   fi.close()
```

```
#Thuật toán
s=0
for i in range(0,n-2):
        for j in range(i+1,n-1):
                for z in range(j+1,n):
                        if 2*a[i]-3*a[j]+5*a[z]>s: s=2*a[i]-3*a[j]+5*a[z]
#Ghi file OUT
fo=open('BOBASO.OUT','w')
fo.write(str(s))
fo.close()
```

☼ Tìm hiểu:

5. ĐỀ THI HSG HUYỆN BẢO LÂM – NĂM HỌC 2020 – 2021

Câu 1: ***(6 điểm)*** **Diện tích tam giác vuông**

Cho ba số tự nhiên a, b, c ($0 < a, b, c < 10^6$). Hãy viết chương tính diện tích tam giác nếu ba số đã cho là số đo ba cạnh của một tam giác vuông.

Dữ liệu vào từ file: **TAMGIAC.INP**

- Dòng đầu tiên ghi ba số tự nhiên a, b và c, các số cách nhau khoảng trắng.

Kết quả ra file: **TAMGIAC.OUT**

- Dòng đầu tiên ghi diện tích tính được (làm tròn một chữ số thập phân), nếu nếu ba số đã cho không phải là ba cạnh của tam giác vuông thì ghi số -1.

Ví dụ:

TAMGIAC.INP	TAMGIAC.OUT
3 4 5	6.0
3 4 9	-1

Phân tích:

Code tham khảo:

```
#Đọc dữ liệu từ file INP
fi = open('TAMGIAC.INP')
a,b,c = map(int,fi.readline().split())
fi.close()
#Thuật toán
s = -1
if a*a == b*b + c*c: s = round(0.5*b*c,1)
elif b*b == a*a + c*c: s = round(0.5*a*c,1)
elif c*c == b*b + a*a: s= round(0.5*a*b,1)
#Ghi file OUT
fo = open('TAMGIAC.OUT','w')
fo.write(str(s))
fo.close()
```

Cải tiến code trên bằng cách sử dụng mảng một chiều như sau:

```
#Đọc dữ liệu từ file INP
fi = open('TAMGIAC.INP')
a = list(map(int,fi.readline().split()))
```

```
4   fi.close()
5   #Thuật toán
6   a.sort()
7   s = -1
8   if a[2]**2 == a[0]**2 + a[1]**2: s = round(0.5*a[0]*a[1],1)
9   #Ghi file OUT
10  fo = open('TAMGIAC.OUT','w')
11  fo.write(str(s))
12  fo.close()
```

☼ Tìm hiểu:

Câu 2: ***(7 điểm)*** **Tính số ngày**

Với lịch dương thì các tháng có 31 ngày là: 1, 3, 5, 7, 8, 10, 12 và các tháng có 30 ngày là: 4, 6, 9, 11. Riêng tháng 2 có thể có 28 hoặc 29 ngày tùy theo năm. Từ năm 2012 đến năm 2099, các năm chia hết cho 4 đều là năm nhuận và tháng 2 của năm nhuận có thêm ngày thứ 29.

Cho hai bộ ba số {d1, m1, y1} và {d2, m2, y2}, mỗi bộ ba số là lần lượt là ngày, tháng, năm (2012 <= y1, y2 <=2099). Hãy viết chương trình tính số ngày trong khoảng thời gian được cho.

Dữ liệu vào từ file: **SONGAY.INP**

- Hai dòng đầu, mỗi dòng ghi lần lượt ba số là ngày, tháng và năm, các số cách nhau khoảng trắng.

Kết quả ra file: **SONGAY.OUT**

- Dòng đầu tiên ghi số ngày tính được.

Ví dụ:

SONGAY.INP	**SONGAY.OUT**
25 1 2012 26 3 2012	61
20 1 2018 20 11 2020	1035

Phân tích:

Code tham khảo:

Cách 1: Dựa theo cách tính toán số học (phép đếm)

```
1   #Đọc dữ liệu vào từ file INP
2   fi = open('SONGAY.INP')
3   d1,m1,y1 = map(int,fi.readline().split())
4   d2,m2,y2 = map(int,fi.readline().split())
5   fi.close()
6   #Thuật toán
7   a=[0,31,28,31,30,31,30,31,31,30,31,30,31]
8   songay=0
9   if y1==y2:
10      if y1%4==0: a[2] = 29
11      a[m1] = a[m1] - d1
```

```
12          a[m2] = d2
13          if m2==m1: songay = d2 - d1
14          else: songay = sum(a[m1:m2+1])
15  elif y1<y2:
16          #(1) Tính số ngày từ d1/m1 đến cuối năm y1
17          if y1%4==0: a[2] = 29
18          a[m1] = a[m1] - d1
19          songay=sum(a[m1:])
20          #(2) Tính số ngày từ đầu năm y2 đến d1/m1 đến cuối năm y1
21          a=[0,31,28,31,30,31,30,31,31,30,31,30,31]
22          if y2%4==0: a[2] = 29
23          else: a[2] = 28
24          a[m2] = d2
25          songay=songay+sum(a[:m2+1])
26          #(3) Tính số ngày của những năm ở giữa y2 - y1
27          songay=songay+365*(y2-y1-1)
28          #(4) Nếu năm giữa y2 - y1 có năm nhuận thì thêm 1 ngày
29          for i in range(y1+1,y2):
30                  if i%4==0: songay +=1
31  #Ghi file OUT
32  fo = open('SONGAY.OUT','w')
33  fo.write(str(songay))
34  fo.close()
```

Cách 2: Cải tiến thuật toán ở cách 1 bằng cách lấy cố định 1 thời điểm ban đầu là ngày 01/01/2000 để tính số ngày đến d/m/y như sau:

```
1   #Đọc dữ liệu vào từ file INP
2   fi = open('SONGAY.INP')
3   d1,m1,y1 = map(int,fi.readline().split())
4   d2,m2,y2 = map(int,fi.readline().split())
5   fi.close()
6   #Thuật toán
7   def songay(d,m,y):
8           sn = 365*(y-2000)
9           for i in range(2000,y):
10                  if i%4==0: sn +=1
11          a=[0,31,28,31,30,31,30,31,31,30,31,30,31]
12          if y%4==0: a[2] = 29
13          a[m] = d
14          return sn + sum(a[:m+1])
15  #Ghi file OUT
16  fo = open('SONGAY.OUT','w')
17  fo.write(str(songay(d2,m2,y2) - songay(d1,m1,y1)))
18  fo.close()
```

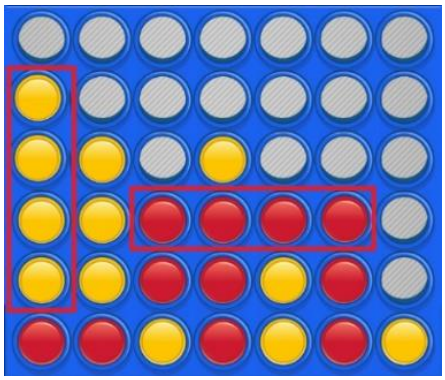
Cách 3: Dùng hàm date() từ thư viện datetime

```
1  #Đọc dữ liệu vào từ file INP
2  fi = open('SONGAY.INP')
3  d1,m1,y1 = map(int,fi.readline().split())
4  d2,m2,y2 = map(int,fi.readline().split())
5  fi.close()
6  #Dùng hàm của thư viện datetime
7  from datetime import date
8  n1 = date(y1,m1,d1)
9  n2 = date(y2,m2,d2)
10 songay = (n2 - n1).days
11 #Ghi file OUT
12 fo = open('SONGAY.OUT','w')
13 fo.write(str(songay))
14 fo.close()
```

☼ Tìm hiểu:

Câu 3: ***(7 điểm)*** **Connect Four**

Connect Four là một bộ **cờ thả Caro 3D**, là trò chơi rèn luyện tư duy toán học, thường được làm từ vật liệu gỗ thân thiện với môi trường, không độc hại và không gây kích ứng, an toàn và lành mạnh cho trẻ em. Luật chơi cho các bé dưới 5 tuổi rất đơn giản như sau: Có hai bé cùng chơi, mỗi bé có 21 quân cờ màu đỏ hoặc màu vàng, khi chơi bé luân phiên thả các quân cờ vào trong 7 hàng dọc, mỗi hàng chứa được 6 quân cờ, cho đến khi thả hết các quân cờ. Bé nào đạt được 4 quân cờ trở lên nối liền nhau theo chiều ngang hoặc chiều dọc sẽ chiến thắng. Nếu cả hai bé cùng được hoặc cùng không được 4 quân cờ trở lên nối liền nhau sẽ hòa nhau.



(Hình bàn cờ **Connect Four**)

Ta kí hiệu số 1 là đại diện quân cờ đỏ, số 0 là đại diện quân cờ vàng. Hãy viết chương trình xác định kết quả trò chơi của hai bé?

Dữ liệu vào từ file: **CARO3D.INP**

- Gồm 6 dòng, mỗi dòng lần lượt ghi 7 số 0 hoặc 1, các số cách nhau một khoảng trắng, dùng để mô tả vị trí chứa các quân cờ trên bàn cờ **Connect Four.**

Kết quả ra file: **CARO3D.OUT**

- Dòng đâu tiên ghi kết quả trò chơi như sau: Nếu quân đỏ thắng ghi 1, nếu quân vàng thắng ghi 0, nếu hòa ghi -1.

Ví dụ:

CARO3D.INP	CARO3D.OUT
1 0 1 0 1 0 0 1 1 1 0 1 1 0 1 0 1 **0 0 0 0** 0 1 0 1 1 0 1 1 0 1 0 1 0 1 0 1 0 1 0 1 0	0
1 0 **1** 0 1 0 1 1 0 **1** 0 0 1 0 1 0 **1 0 0 0 0** 0 **1 1 1 1** 0 1 1 0 **1** 0 1 0 1 0 1 0 1 0 1 0	-1

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi = open('caro3d.inp')
3  a = [i.split() for i in fi]
4  fi.close()
5  #Giả sử ban đầu không thỏa
6  vang = False
7  do = False
8  #Thuật toán tìm theo hàng ngang
9  for i in range(6):
10     for j in range(4):
11         if a[i][j]=='0' and a[i][j+1]=='0' and a[i][j+2]=='0' and a[i][j+3]=='0':
12 vang = True
13         if a[i][j]=='1' and a[i][j+1]=='1' and a[i][j+2]=='1' and a[i][j+3]=='1':
14 do = True
15 #Thuật toán tìm theo hàng doc
16 for i in range(3):
17     for j in range(6):
18         if a[i][j]=='0' and a[i+1][j]=='0' and a[i+2][j]=='0' and a[i+3][j]=='0':
19 vang = True
20         if a[i][j]=='1' and a[i+1][j]=='1' and a[i+2][j]=='1' and a[i+3][j]=='1':
21 do = True
22 #Ghi file OUT
23 fo = open('caro3d.out','w')
24 if do and vang: fo.write('-1')
25 elif not do and not vang: fo.write('-1')
26 elif do: fo.write('1')
27 elif vang: fo.write('0')
28 fo.close()
```

6. ĐỀ THI HSG HUYỆN DI LINH – NĂM HỌC 2020 – 2021

Bài 1: ***(5 điểm)*** **Đổi quà**

Một cửa hàng đang có chương trình khuyến mãi "mua hàng tích lũy điểm để đổi quà". Theo đó, người mua được tích lũy điểm bằng cách như sau: Mua lần thứ nhất được tính 1 điểm; mua lần thứ hai được tính 2 điểm, …, cứ như thế nếu mua đến lần thứ n thì được tính n điểm. Điều kiện được đổi quà là tổng số điểm tích lũy sau nhiều lần mua phải lớn hơn hoặc bằng k điểm.

Viết chương trình cho biết lần mua thứ n và tổng số điểm tích lũy được tới lần mua đó để đủ điều kiện đổi quả của lần đầu tiên.

Dữ liệu vào từ file: **bai1.inp**

- Gồm 1 dòng duy nhất chứa số nguyên k là số điểm giới hạn để đổi quà ($0<k\leq10^9$).

Kết quả ra file: **bai1.out**

Gồm 2 dòng:

+ Dòng 1: ghi số nguyên n.

+ Dòng 2: ghi tổng số điểm tích lũy được đến lần mua thứ n.

Ví dụ:

bai1.inp	bai1.out	Giải thích
10	4 10	Đến lần mua thứ 4 có tổng số điểm tích lũy là 10 điểm.
1000	45 1035	Đến lần mua thứ 45 có tổng số điểm tích lũy là 1035 điểm.

Phân tích:

Là bài toán tính S = 1 + 2 + 3 + … < k

Code tham khảo:

```
#Đọc dữ liệu từ file INP
fi = open('bai1.inp')
k = int(fi.readline())
fi.close()
#Thuật toán
s = 0; i = 0
while s<k: i += 1; s += i
#Ghi file OUT
fo = open('bai1.out','w')
fo.write(str(i)+'\n' + str(s))
fo.close()
```

Bài 2: ***(5 điểm)*** **Sắp xếp ký tự số**

Cho trước một xâu ký tự, trong xâu đó có thể chứa các ký tự "số" và ký tự không là "số". Viết chương trình tách các ký tự "số" của xâu đó và sắp xếp lại theo thứ tự giảm dần. Nếu không có ký tự "số" nào thì ghi "khong".

Dữ liệu vào từ file: **bai2.inp**

- Gồm 1 dòng duy nhất chứa một xâu kí tự.

Kết quả ra file: **bai2.out**

Gồm 1 dòng ghi xâu ký tự "số" đã được sắp xếp theo thứ tự giảm dần hoặc "khong" nếu xâu đó không chứa ký tự "số".

Ví dụ:

bai2.inp	**bai2.out**
Viet3135 Nam	5331
Nguyen Van Nam	khong
046054219871	987654421100

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi = open('bai2.inp')
3  xau = fi.readline()
4  fi.close()
5  #Thuật toán
6  a = [x for x in xau if x in '0123456789']
7  #Ghi file OUT
8  fo = open('bai2.out','w')
9  if len(a) ==0: fo.write('khong')
10 else: fo.write(''.join(sorted(a,reverse=True)))
11 fo.close()
```

☼ Tìm hiểu: Tìm hiểu cách dùng các hàm isdigit(), isalpha(), isalnum()

Bài 3: ***(5 điểm)*** **Chia kẹo**

Có n gói kẹo lần lượt chứa a_1, a_2, .., a_n viên kẹo. Bạn thứ nhất được chia các gói kẹo từ 1 đến i ($1 \le i < n$), bạn thứ hai được chia các gói kẹo từ i+1 đến n. Viết chương trình chia n gói kẹo đó cho hai bạn sao cho chênh lệch giữa số viên kẹo nhận được của hai bạn là nhỏ nhất (yêu cầu không bóc các gói kẹo).

Dữ liệu vào từ file: **bai3.inp**

Gồm 2 dòng:

+ Dòng thứ nhất chứa số tự nhiên n ($1< n \le 100$).

+ Dòng thứ hai gồm n số từ a_1, a_2, .., a_n ($1< a_i \le 10^9$; i = 1,2,3,…,n) mỗi số cách nhau một ký tự khoảng trắng.

Kết quả ra file: **bai3.out**

Gồm nhiều dòng:

Mỗi phương án được ghi trên hai dòng liên tiếp: Dòng thứ nhất ghi số kẹo có trong từng gói từ 1 đến i; dòng thứ hai ghi số kẹo có trong từng gói từ I + 1 đến n, mỗi số cách nhau một khoảng trắng.

Ví dụ:

bai3.inp	**bai3.out**	**Giải thích**
4 3 6 8 5	3 6 8 5	Độ chênh lệch nhỏ nhất là 4
5 3 4 6 2 5	3 4 6 2 5	Độ chênh lệch nhỏ nhất là 6

	3 4 6 2 5	

Phân tích:

Code tham khảo:

```
1   #Đọc dữ liệu từ file INP
2   fi = open('bai3.inp')
3   n = int(fi.readline())
4   a = list(map(int,fi.readline().split()))
5   fi.close()
6   #Thuật toán: tìm độ chênh lệch nhỏ nhất
7   dcl = sum(a) #gán tạm giá trị ban đầu
8   for i in range(n):
9       if abs(sum(a[:i+1]) - sum(a[i+1:])) < dcl:
10          dcl = abs(sum(a[:i+1]) - sum(a[i+1:]))
11  #Ghi file OUT
12  fo = open('bai3.out','w')
13  for i in range(n):
14      if abs(sum(a[:i+1]) - sum(a[i+1:])) == dcl:
15          fo.write(' '.join(map(str,a[:i+1]))+'\n'+' '.join(map(str,a[i+1:]))+'\n')
16  fo.close()
```

☼ Tìm hiểu:

Bài 4: ***(5 điểm)*** **Số lượng số nguyên tố trên hàng**

Số nguyên tố là số tự nhiên lớn hơn 1 và chỉ có hai ước là 1 và chính nó. Cho một bảng số nguyên gồm m hàng và n cột ($0<m,n\leq 100$). Hãy viết chương trình cho biết hàng nào có chứa nhiều số nguyên tố nhất.

Dữ liệu vào từ file: **bai4.inp**

Gồm nhiều dòng:

+ Dòng thứ nhất chứa hai số tự nhiên m và n, cách nhau bởi một ký tự khoảng trắng.

+ m dòng tiếp theo: mỗi dòng chứa n số nguyên, các số cách nhau một ký tự khoảng trắng.

Kết quả ra file: **bai4.out**

Gồm một hoặc nhiều dòng, mỗi dòng ghi số thứ tự của hàng có chứa nhiều số nguyên tố nhất. Nếu tất cả các hàng đều không có chứa số nguyên tố thì ghi "khong".

Ví dụ:

bai4.inp	**bai4.out**	**Giải thích**
3 4 4 2 8 6 2 1 9 3 6 3 5 8	2 3	Hàng số 2 và 3 đều có 2 số nguyên tố (Số lượng nhiều nhất)
5 5 3 2 5 7 9	1 4	Hàng số 1 và 4 đều có 4 số nguyên tố (Số lượng nhiều nhất)

4 3 9 1 6 8 4 9 1 6 5 2 1 3 5 6 10 2 1 3		
3 3 1 4 6 8 9 10 1 12 16	khong	Không hàng nào có chứa số nguyên tố.

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi=open('bai4.inp')
3  m,n=map(int,fi.readline().split())#m=hàng, n = cột
4  a=[]
5  for i in fi: a.append(list(map(int,i.split())))
6  fi.close()
7  #Hàm kiểm tra số nguyên tố
8  def ktnt(k):
9      if k<2: return False
10     for i in range(2,k//2+1):
11         if k%i==0: return False
12     return True
13 #Hàm đếm số nguyên tố trên List 1 chiều
14 def demnt(b):
15         demt=0
16         for i in b:
17                 if ktnt(i): demt += 1
18         return demt
19 #Thuật toán
20 maxnt = 0
21 for i in a:
22     if demnt(i)>maxnt: maxnt=demnt(i)
23 #Ghi file OUT
24 fo=open('bai4.out','w')
25 if maxnt ==0: fo.write('khong')
26 else:
27         for j,i in enumerate(a):
28                 if demnt(i) == maxnt: fo.write(str(j+1)+'\n')
29 fo.close()
```

☼ Tìm hiểu:

7. ĐỀ THI HSG HUYỆN ĐỨC TRỌNG – NĂM HỌC 2020 – 2021

Bài 1: ***(5 điểm)*** **Nguyên tố**

Cho số nguyên dương n ($n < 10^{25}$). Viết chương trình kiểm tra tổng các chữ số của số đã cho có phải là số nguyên tố hay không.

Dữ liệu vào từ file: **NGUYENTO.INP**

- Gồm 1 dòng chứa số nguyên dương n.

Kết quả ra file: **NGUYENTO.OUT**

- Dòng 1: In ra tổng của các chữ số của số n.
- Dòng 2: In ra "YES" nếu tổng tìm được là số nguyên tố hoặc "NO" nếu tổng tìm được không là số nguyên tố.

Ví dụ:

NGUYENTO.INP	NGUYENTO.OUT
14	5 YES
56137	22 NO

Phân tích:

Code tham khảo:

```
#Hàm kiểm tra số nguyên tố
def ktnt(k):
    if k<2: return False
    for i in range(2,k//2+1):
        if k%i==0: return False
    return True
#Hàm tính tổng các chữ số của 1 số
def tongcs(n):
        return sum(list(map(int,str(n))))
#Đọc dữ liệu từ file INP
fi = open('NGUYENTO.INP')
n = int(fi.readline())
fi.close()
#Ghi file OUT
fo = open('NGUYENTO.OUT','w')
fo.write(str(tongcs(n)) + '\n')
if ktnt(tongcs(n)): fo.write('YES')
else: fo.write('NO')
fo.close()
```

Bài 2: ***(5 điểm)*** **Phân biệt**

Cho dãy số A[n] với n < 100 gồm các hạng tử A[i] là các số nguyên (0<A[i]<10000). Viết chương trình tìm các phần tử phân biệt (các phần tử phân

biệt là phần tử đôi một khác nhau) trong dãy số trên và in ra các số vừa tìm được với thứ tự không thay đổi.

Dữ liệu vào từ file: **PHANBIET.INP**

- Dòng 1: Ghi số n.
- Dòng 2: Ghi các phần tử của dãy số, mỗi số viết cách nhau ít nhất một khoảng trắng.

Kết quả ra file: **PHANBIET.OUT**

- Dòng 1: In ra số phần tử phân biệt của dãy số.
- Dòng 2: In ra các phần tử phân biệt tìm được.

Ví dụ:

PHANBIET.INP	**PHANBIET.OUT**
8 7 12 4 9 8 12 9 15	6 7 12 4 9 8 15

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi = open('PHANBIET.INP')
3  n = int(fi.readline())
4  a = list(map(int,fi.readline().split()))
5  fi.close()
6  #Thuật toán
7  b=[]
8  [b.append(x) for x in a if x not in b]
9  #Ghi file OUT
10 fo = open('PHANBIET.OUT','w')
11 fo.write(str(len(b))+'\n')
12 fo.write(' '.join(map(str,b)))
13 fo.close()
```

Bài 3: ***(5 điểm)*** **Dãy tăng**

Cho dãy số C[n] với n < 100 gồm các hạng tử C[i] là các số nguyên có giá trị trong khoảng -10000 <C[i]<10000. Viết chương trình in ra dãy con liên tục tăng dài nhất theo chiều từ trái sang phải của mảng C (nếu tìm được nhiều dãy con thì lấy dãy con cuối cùng).

Dữ liệu vào từ file: **DAYTANG.INP**

- Dòng 1: Ghi số n.
- Dòng 2: Ghi các phần tử của dãy số, mỗi số viết cách nhau ít nhất một khoảng trắng.

Kết quả ra file: **DAYTANG.OUT**

- Dòng 1: In ra số phần tử của dãy con tìm được.
- Dòng 2: In ra các phần tử của dãy con tìm được, mỗi số viết cách nhau ít nhất một khoảng trắng.

Ví dụ:

DAYTANG.INP	**DAYTANG.OUT**

13 3 4 -1 2 13 7 8 9 -5 -2 3 10 9	4 -5 -2 3 10
11 2 13 7 8 9 -5 21 3 9 10 -7	3 3 9 10

Phân tích:

Code tham khảo:

```
#Đọc dữ liệu từ file INP
fi = open('DAYTANG.INP')
n = int(fi.readline())
a = list(map(int,fi.readline().split()))
fi.close()
#Hàm dãy con tăng
def daycontang(a):
    for i in range(len(a)-1):
        if a[i]>a[i+1]: return False
    return True
#Tìm dãy con tăng lớn nhất
b=[]
for i in range(n-1):
    for j in range(i+1,n+1):
        if daycontang(a[i:j]):
            if len(a[i:j])>=len(b): b=list(a[i:j])
#Ghi file OUT
fo = open('DAYTANG.OUT','w')
fo.write(str(len(b))+'\n' + ' '.join(map(str,b)))
fo.close()
```

Bài 4: ***(5 điểm)*** **Ma trận**

Cho mảng hai chiều A có kích thước n dòng và m cột (2 <n, m<100) gồm các phần tử A[i,j] là các số nguyên (0 < A[i,j] < 100). Ma trận C là mảng con của A có kích thước k dòng k cột (k < n, k < m) gồm các phần tử liền kề nhau trong mảng A. Hãy viết chương trình chọn ma trận C trong trường hợp phần tử nhỏ nhất của C là số lớn nhất trong các ma trận C tìm được.

Dữ liệu vào từ file: **MATRAN.INP**

- Dòng 1: Ghi số n, m, k mỗi số cách nhau ít nhất một khoảng trắng.
- n dòng tiếp theo chứa các phần tử A[i,j] của mảng A, mỗi số viết cách nhau ít nhất một khoảng trắng.

Kết quả ra file: **MATRAN.OUT**

- In ra phần tử nhỏ nhất của ma trận C đã chọn.

Ví dụ:

MATRAN.INP	**MATRAN.OUT**	**Giải thích**
3 4 2 1 3 5 7 2 4 6 8	5	Ma trận C tìm được là: 5 7 6 8

1 2 3 5		Phần tử nhỏ nhất của C là 5.

Phân tích: Cách diễn đạt của đề bài và ví dụ chưa rõ ý. Có thể phân tích lại từ ví dụ như sau:

Các mảng con C 2x2, được trích ra từ A 3x4 gồm:					
1 3 2 4	3 5 4 6	5 7 6 8	2 4 1 2	4 6 2 3	6 8 3 5
Phần tử nhỏ nhất của các mảng con C:					
1	3	5	1	2	3
Dữ liệu ra là 5, là số lớn nhất trong các số nhỏ nhất của các mảng con C.					

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi=open('MATRAN.INP')
3  m,n,k=map(int,fi.readline().split())
4  a=[]
5  for i in fi: a.append(list(map(int,i.split())))
6  fi.close()
7  #Thuật toán
8  b=[]
9  for i in range(m-k+1): #duyệt qua (m-k) hàng
10         i1 = -1 #gán -1 để phép cộng dòng lệnh 12 là "cột 0"
11         for j in range(n-k+1): #duyệt qua (n-k) cột
12                 i1 +=1 #tăng liên tiếp cột để lấy ma trận con C
13                 mint=1000 #một số lớn không có trong A, để dòng lệnh 15 thực thi ngay
14                 for l in range(k): #duyệt qua k hàng liên tiếp trên A, tìm min của đoạn con k phần tử
15                         if mint > min(a[i+l][i1:i1+k]): mint = min(a[i+l][i1:i1+k])
16                 b.append(mint) #lưu phần tử min đó vào mảng 1 chiều
17 #Ghi file OUT
18 fo = open('MATRAN.OUT','w')
19 fo.write(str(max(b)))
20 fo.close()
```

☼ Tìm hiểu: - Bổ sung vào đoạn code trên thêm 2 dòng lệnh (như hình) để quan sát kết quả tách các ma trận con C 2x2 từ ma trận A.

```
 7 #Thuật toán
 8 b=[]
 9 for i in range(m-k+1): #duyệt qua (m-k) hàng
10         i1 = -1 #gán -1 để phép cộng dòng lệnh 12 là "cột 0"
11         for j in range(n-k+1): #duyệt qua (n-k) cột
12                 print('Ma trận con C thứ: ',len(b)+1)
13                 i1 +=1 #tăng liên tiếp cột để lấy ma trận con C
14                 mint=1000 #một số lớn không có trong A, để dòng lệnh 15 thực thi ngay
15                 for l in range(k): #duyệt qua k hàng liên tiếp trên A, tìm min của đoạn con k phần tử
16                         if mint > min(a[i+l][i1:i1+k]): mint = min(a[i+l][i1:i1+k])
17                         print(a[i+l][i1:i1+k])
18                 b.append(mint) #lưu phần tử min đó vào mảng 1 chiều
```

- Từ đoạn code trên ta bổ sung thêm các lệnh sau, sẽ lưu lại được các ma trận con C 2x2 vào mảng C (như hình):

```
#Thuật toán
b=[]
c=[]
for i in range(m-k+1): #duyệt qua (m-k) hàng
    i1 = -1 #gán -1 để phép cộng dòng lệnh 12 là "cột 0"
    for j in range(n-k+1): #duyệt qua (n-k) cột
        tam=[]
        i1 +=1 #tăng liên tiếp cột để lấy ma trận con C
        mint=1000 #một số lớn không có trong A, để dòng lệnh 15 thực thi ngay
        for l in range(k): #duyệt qua k hàng liên tiếp trên A, tìm min của đoạn con k phần tử
            if mint > min(a[i+l][i1:i1+k]): mint = min(a[i+l][i1:i1+k])
            tam.append(a[i+l][i1:i1+k])
        c.append(tam)
        b.append(mint) #lưu phần tử min đó vào mảng 1 chiều
```

Kết quả của ma trận C:

```
>>> for i in c: print(i)

[[1, 3], [2, 4]]
[[3, 5], [4, 6]]
[[5, 7], [6, 8]]
[[2, 4], [1, 2]]
[[4, 6], [2, 3]]
[[6, 8], [3, 5]]
```

8. ĐỀ THI HSG HUYỆN LÂM HÀ – NĂM HỌC 2020 – 2021

Câu 1: ***(6 điểm)*** **Đếm số trong chuỗi**

Cho một chuỗi ký tự S (S không quá 255 ký tự), trong S có các ký tự là chữ số. Viết chương trình đếm có bao nhiêu ký tự là chữ số xuất hiện trong chuỗi.

Dữ liệu vào từ file: **DEMCHUSO.INP**

- Dòng đầu tiên chứa chuỗi S không quá 255 ký tự.

Kết quả ra file: **DEMCHUSO.INP**

- Dòng đầu tiên ghi tổng các ký tự là chữ số đếm được.

Ví dụ:

DEMCHUSO.INP	DEMCHUSO.OUT
Ky2 thi ho5c sinh gioi3	3
Nam hoc 2020 - 2021	8

Phân tích:

Code tham khảo:

```
#Đọc dữ liệu từ file INP
fi = open('DEMCHUSO.INP')
s = fi.readline()
fi.close()
#Thuật toán
```

```
dem = 0
for i in s:
    if i in '0123456789': dem += 1
#Ghi file OUT
fo = open('DEMCHUSO.OUT','w')
fo.write(str(dem))
fo.close()
```

Câu 2: ***(7 điểm)*** **Tính lãi suất**

Sau một đơn vị thời gian (kỳ hạn), tiền lãi được gộp vào vốn và được tính lãi, cách tính lãi này được gọi là ***Lãi kép (hay lãi cộng dồn).***

Chẳng hạn: Khi gửi tiết kiệm số tiền gốc là 100 (triệu đồng) vào ngân hàng với lãi suất 6.9% một năm thì sau một năm, ta nhận được số tiền cả gốc và lãi là:

100 + 100*6.9% = 106.9 (triệu đồng)

Toàn bộ số tiền 106.9 (triệu đồng) được gọi là tiền gốc để tính lãi cho năm tiếp theo (lãi suất cho các năm tiếp theo không thay đổi).

Dữ liệu vào từ file: **LAIKEP.INP**

- Dòng đầu tiên chứa ba số G, L và N, lần lượt là số tiền gốc, lãi suất một năm và số năm, các số cách nhau một khoảng trắng ($0<G<10^{16}$, $4.0<L<12.0$, $0<N<50$).

Kết quả ra file: **LAIKEP.OUT**

- Dòng đầu tiên ghi tổng số tiền nhận được, làm tròn hai chữ số thập phân.

Ví dụ:

LAIKEP.INP	LAIKEP.OUT	Giải thích
100 6.9 2	114.28	Sau năm thứ 2, số tiền cả gốc và lãi là: 106.9 + 106.9*6.9% = 114.28 (triệu đồng) (trong giải thích trên: dấu * là phép nhân, dấu . là dấu phẩy số thập phân)

Phân tích:

Code tham khảo:

```
#Đọc dữ liệu từ file INP
fi = open('LAIKEP.INP')
g,l,n = fi.readline().split()
g,l,n = int(g), float(l), int(n)
fi.close()
#Thuật toán
tien = g*((1 + l/100)**n)
#Ghi file OUT
fo = open('LAIKEP.OUT','w')
print(round(tien,2),file=fo)
fo.close()
```

Câu 3: ***(7 điểm)*** **Đoạn con**

Cho dãy A gồm N số nguyên dương ($0 < N < 10^3$). Một đoạn con của dãy A, kí hiệu A[i..j] là dãy gồm các phần tử đứng liên tiếp nhau trong dãy A, tính cả phần tử A_i đến phần tử A_j ($1 <= i <= j <= N$). Hãy tìm đoạn con dài nhất gồm các phần tử đôi một khác nhau.

Dữ liệu vào từ file: **DOANCON.INP**

- Dòng đầu tiên chứa số N.
- Dòng thứ hai chứa N số nguyên dương, các số cách nhau một khoảng trắng.

Kết quả ra file: **DOANCON.OUT**

- Dòng đầu tiên ghi chỉ số đầu tiên của đoạn dài nhất tìm được trong dãy A.
- Dòng thứ hai ghi số phần tử của doạn dài nhất tìm được.

Ví dụ:

DOANCON.INP	DOANCON.OUT	Giải thích
10 5 2 4 1 5 3 2 7 6 9	3 8	Tính từ phần tử thứ 3 có 8 số đôi một khác nhau là: 4 1 5 3 2 7 6 9

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi=open("doancon.inp")
3  n=int(fi.readline())
4  a=list(map(int,fi.readline().split()))
5  fi.close()
6  #Hàm xác định đoạn con phân biệt
7  def dcon(a):
8      for i in a:
9          if a.count(i)>1: return False
10     return True
11 #Thuật toán với kĩ thuật lát cắt trên List
12 c=[] #để lưu bảng phương án khả thi
13 max_c=0
14 for i in range(n):
15     for j in range(i,n):
16         if dcon(a[i:j+1]):
17             c.append([i+1,len(a[i:j+1])])
18             if max_c<len(a[i:j+1]): max_c = len(a[i:j+1])
19 #Ghi file OUT
20 fo = open('DOANCON.OUT','w')
21 for i in c:
22     if i[1]==max_c: fo.write(str(i[0])+'\n'+str(i[1]))
23 fo.close()
```

☼ Tìm hiểu: - Nếu bỏ lưu bảng phương án khả thi c[] và thay bằng kĩ thuật lính cầm canh, thuật toán sẽ chạy nhanh hơn.

9. ĐỀ THI HSG HUYỆN ĐƠN DƯƠNG – NĂM HỌC 2020 – 2021

Câu 1: ***(6 điểm)*** **Chữ số lớn nhất**

Cho số tự nhiên N ($1000 < N < 10^{12}$). Viết chương trình tìm chữ số lớn nhất của số N?

Dữ liệu vào từ file: **CHUSO.INP**

- Dòng đầu tiên ghi hai số nguyên dương N.

Kết quả ra file: **CHUSO.OUT**

- Dòng đầu tiên ghi chữ số lớn nhất của số N tìm được.

Ví dụ:

CHUSO.INP	CHUSO.OUT
2369524	9
333333333	3

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi = open('CHUSO.INP')
3  s = fi.readline()
4  fi.close()
5  #Ghi file OUT
6  fo = open('CHUSO.OUT','w')
7  fo.write(str(max(s)))
8  fo.close()
```

Câu 2: ***(7 điểm)*** **Tách số**

Tách số nguyên dương N (N < 1000) thành tổng của ba số nguyên dương a, b và c, sao cho a là ước chung khác 1 của b và c. Viết chương trình tính xem có bao nhiêu cách tách số như vậy?

Dữ liệu vào từ file: **TACHSO.INP**

- Dòng đầu tiên nguyên dương N.

Kết quả ra file: **TACHSO.OUT**

- Dòng đầu tiên ghi kết quả đếm được.

Ví dụ:

TACHSO.INP	TACHSO.OUT
18	12

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi = open('TACHSO.INP')
3  n = int(fi.readline())
4  fi.close()
5  #Thuật toán
6  dem = 0
7  for a in range(2,n//3+1):
```

```
8       for b in range(a,n-a+1):
9           c = n - a - b
10          if b%a==0 and c%a==0 and c>=a: dem +=1
11  #Ghi file OUT
12  fo = open('TACHSO.OUT','w')
13  fo.write(str(dem))
14  fo.close()
```

Câu 3: ***(7 điểm)*** **Tắc kè hoa**

Tắc kè hoa là loài vật có khả năng thay đổi màu sắc của da. Màu sắc da là ngôn ngữ được tắc kè hoa sử dụng để bảo vệ lãnh thổ, thể hiện cảm xúc, giao tiếp với đồng loại và điều tiết thân nhiệt.

Trong một khu vườn có ba loại tắc kè hoa màu đỏ, xanh, vàng với số lượng lần lượt là: x, y, z (x, y, z < 500). Mỗi khi hai con khác màu chạm vào nhau thì chúng sẽ đổi sang màu còn lại. Viết chương trình liệt kê bảng phương án cho các con tắc kè hoa trong vườn chạm nhau cho đến khi chỉ còn lại một màu.

Dữ liệu vào từ file: **TACKE.INP**

- Dòng đầu tiên ghi lần lượt ba số x, y, z; mỗi số cách nhau một khoảng trống.

Kết quả ra file: **TACKE.OUT**

- Gồm nhiều dòng, mỗi dòng gồm 3 số tự nhiên là số lượng của từng loại tắc kè hoa sau một bước va chạm. Nếu không có phương án nào thì ghi "KHONG'.

Ví dụ:

TACKE.INP	TACKE.OUT	GIẢI THÍCH KẾT QUẢ
3 4 1	5 3 0 4 2 2 6 1 1 8 0 0	1 con xanh chạm 1 con vàng 1 con đỏ chạm 1 con xanh 1 con xanh chạm 1 con vàng 1 con xanh chạm 1 con vàng
2 3 4	KHONG	

9. ĐỀ THI HSG LỚP 9 TỈNH LÂM ĐỒNG – NĂM HỌC 2020 – 2021

Câu 1: ***(5 điểm)*** **Tìm số nguyên tố**

Cho hai số nguyên n, m. Viết chương trình tìm các số nguyên tố không vượt quá n sao cho tổng các chữ số của mỗi số đều bằng m.

Dữ liệu vào từ file: **TIMSNT.INP**

- Dòng duy nhất ghi 2 số nguyên n và m ($0 < n, m \leq 10^6$), cách nhau một khoảng trắng.

Kết quả ra file: **TIMSNT.OUT**

- Dòng đầu tiên ghi số lượng các số nguyên tố tìm được
- Dòng thứ hai ghi các số nguyên tố thỏa mãn yêu cầu, mỗi số cách nhau một khoảng trắng.

Ví dụ:

TIMSNT.INP	TIMSNT.OUT
50 5	3 5 23 41
1000 18	0

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi=open('TIMSNT.INP')
3  n,m=map(int,fi.readline().split())
4  fi.close()
5  #Thuật toán
6  def ktnt(a):
7      if a<2: return False
8      for i in range(2,a//2+1):
9          if a%i==0: return False
10     return True
11 def tongcs(a):
12     return sum(list([int(i) for i in str(a)]))
13 #Ghi dữ liệu vào file OUT
14 fo=open('TIMSNT.OUT','w')
15 a=[]
16 for i in range(n+1):
17     if ktnt(i) and tongcs(i)==m: a.append(i)
18 if len(a)!=0: fo.write(str(len(a))+'\n'+' '.join(map(str,a)))
19 else: fo.write(str(0))
20 fo.close()
```

Câu 2: ***(5 điểm)*** **Ký tự đầu tiên**

Một chuỗi ký tự S gồm toàn chữ cái tiếng Anh in thường, hãy tìm vị trí của ký tự đầu tiên xuất hiện k lần trong S. Các ký tự trong chuỗi đánh số từ trái qua phải lần lượt là 1, 2,… Nếu không có ký tự nào như vậy thì in ra -1.

Dữ liệu vào từ file: **KYTUDT.INP**

- Dòng đầu tiên ghi số nguyên k ($0 < k \leq 10^6$)
- Dòng thứ hai ghi chuỗi ký tự S có độ dài không quá 10^6

Kết quả ra file: **KYTUDT.OUT**

- Dòng duy nhất ghi vị trí của ký tự thỏa yêu cầu

Ví dụ:

KYTUDT.INP	KYTUDT.OUT
2 abcdbc	2
3 abbaccdd	-1

Phân tích:

Code tham khảo:

```
#Đọc dữ liệu từ file INP
fi=open('KYTUDT.INP')
k=int(fi.readline())
s=fi.readline()
fi.close()
#Thuật toán
vt=-1
for i in range(len(s)-1):
    if s.count(s[i])==k:
        vt=s.index(s[i])
        break
#Ghi dữ liệu ra file OUT
fo=open('KYTUDT.OUT','w')
if vt==-1: fo.write(str(-1))
else: fo.write(str(vt+1))
fo.close()
```

Câu 3: ***(5 điểm)*** **Tạo ma trận**

Một bảng có kích thước 5 x 5 gồm các số nguyên dương được sắp xếp theo quy luật sau:

1	2	3	4	5
2	4	6	8	10
3	6	9	2	4
4	8	2	4	6
5	10	4	6	8

Hãy tạo bảng n x n theo quy luật trên.

Dữ liệu vào từ file: **MATRAN.INP**

- Dòng duy nhất ghi số n ($1 \leq n \leq 600$)

Kết quả ra file: **MATRAN.OUT**

- Gồm n dòng, mỗi dòng gồm n số được sắp xếp theo quy luật, mỗi số cách nhau một khoảng trắng.

Ví dụ:

MATRAN.INP	MATRAN.OUT
6	1 2 3 4 5 6 2 4 6 8 10 12 3 6 9 12 2 4 4 8 12 2 4 6 5 10 2 4 6 8 6 12 4 6 8 10

Phân tích:

Code tham khảo:

```
1  #Đọc dữ liệu từ file INP
2  fi = open('TAOBANG.INP')
3  n = int(fi.read())
4  fi.close()
5  # tăng thêm 1 hàng và 1 cột để dễ tính (không quan trọng)
6  a=[]
7  for i in range(n+1): a.append([0]*(n+1))#tạo mảng full số 0
8  #Điền hàng 1, cột 1
9  for i in range(1,n+1): #tạo từ hàng 1 cột 1, bỏ hàng 0 cột 0
10       a[1][i] = i;       a[i][1] = a[1][i]
11 #Điền bảng số theo quy luật được cho
12 i =1 #gán = 1 là vì hàng 1, cột 1 đã điền số tăng dần
13 while i<=n:
14       i += 1
15       for j in range(i, n+1):
16            t = False
17            for k  in range(2,j):#các hàng phía sau chưa điền số
18                 if a[k-1][i]>a[k][i]: t = True
19            if t:#nếu chưa điền số
20                 if a[j-1][i] + 2>n*2: a[j][i] = 2 #nếu >2*n bắt đầu lại từ 2
21                 else: a[j][i]=a[j-1][i]+2 #số trước + thêm 2
22                 a[i][j]=a[j][i] #gán phần tử chuyển vị
23            else:
24                 if a[j-1][i] + i>n*2: a[j][i] = 2
25                 else: a[j][i]=a[j-1][i]+i
26                 a[i][j]=a[j][i]
27 #Ghi vào file kết quả
28 fo = open('TAOBANG.OUT','w')
29 for i in range(1,n+1): #xuất từ hàng 1, cột 1 đến n
30       for j in range(1,n+1): fo.write(str(a[i][j]) + ' ')
31       fo.write('\n')
32 fo.close()
```

Câu 4: ***(5 điểm)*** **Mua bất động sản**

Anh Hoàng được thừa kế một khoản tiền n tỷ đồng. Anh ấy quyết định đầu tư vào việc kinh doanh bất động sản bằng cách mua đất tại những vị trí đắc địa nhất của thành phố. Các mảnh đất anh mua có dạng hình vuông, kích thước là số nguyên, mỗi mét vuông đất có giá trị 1 tỷ đồng.

Hãy chỉ cách cho anh Hoàng mua các mảnh đất có kích thước hợp lý sao cho tổng số tiền đúng bằng n tỷ đồng và số mảnh đất mua được là ít nhất. Nếu có nhiều lựa chọn thì chọn phương án chứa mảnh đất có kích thước lớn nhất.

Chẳng hạn: Với n = 60 có hai phương án $60 = 5^2 + 5^2 + 3^2 + 1^2$ và $60 = 7^2 + 3^2 + 1^2 + 1^2$ thì chọn $60 = 7^2 + 3^2 + 1^2 + 1^2$

Dữ liệu vào từ file: **BDS.INP**

- Dòng duy nhất ghi số nguyên dương n $(n \leq 60000)$

Kết quả ra file: **BDS.OUT**

- Dòng duy nhất ghi một dãy số nguyên dương là kích thước cạnh các mảnh đất (đơn vị tính: mét), xếp theo thứ tự giảm dần.

Ví dụ:

BDS.INP	**BDS.OUT**
30	5 2 1
60	7 3 1 1

Phân tích:

Code tham khảo:

```
#Đọc dữ liệu từ file INP
import math
fi=open('BDS.INP')
n=int(fi.readline())
fi.close()
#Thuật toán
a=[i*i for i in range(1,int(math.sqrt(n))+1) if i*i <=n]#bảng bình phương
<=n
a.sort(reverse=True)#sắp xếp giảm dần
c=[]#bảng phương án khả thi
chon=n #chọn phương án tốt nhất (số mảnh đất bé nhất)
for i in a:
    b=[i]
    [b.append(x) for x in a if sum(b)+x<=n]#QHĐ đơn giản bằng List
Comprehension
    if sum(b)==n:
        c.append(b) #ghi nhận b vào bảng phương án khả thi
        if chon>len(b): chon=len(b) #đổi lại phương án tốt nhất
#Ghi dữ liệu file OUT
fo=open('BDS.OUT','w')
s=[0]#chốt giá trị cuối cùng
for i in c:
  if len(i)==chon:
```

```
24      if max(i)>max(s):
25          s=[int(math.sqrt(x)) for x in i] #lấy các phương án khả thi nhất
26  s.sort(reverse=True)
27  fo.write(' '.join(map(str,s)))
28  fo.close()
```