MỨC 1.

1.1. Viết các chương trình con:

a. Xét 1 số có phải là số nguyên tố hay không ?

b. In trên màn hình n số nguyên tố đầu tiên, kể từ số 2

1.2. Chúng ta đều biết định lý Pitago nổi tiếng về tam giác vuông. Bài toán đặt ra là cho trước độ dài 3 cạnh . Hãy xác định xem đó có phải tam giác vuông hay không.

Input

- Gồm nhiều bộ test. Mỗi bộ test viết trên một dòng 3 số nguyên dương không quá 30000, lần lượt là độ dài ba cạnh.
- Input kết thúc với 3 số 0

Output

- Với mỗi bộ test, in ra màn hình, trên một dòng, chữ "right" nếu đó là một tam giác vuông, "wrong" nếu ngược lại.

Ví dụ

INPUT	OUTPUT
6 8 10	right
25 52 60	wrong
5 12 13	right
0 0 0	

1.3. Sở giao thông TP. Hồ Chí Minh quyết định bán đấu giá các biển số xe đẹp để lấy tiền ủng hộ đồng bào nghèo. Một biển số xe được gọi là đẹp nếu nó thỏa mãn các điều kiện sau :

- Là một số nguyên dương T mà $A \leq T \leq B$, trong đó A, B là 2 số nguyên dương cho trước ;
- T là một số nguyên tố ;
- T là một số thuận nghịch (đọc T từ trái qua phải cũng thu được kết quả giống như đọc T từ phải qua trái).

Yêu cầu : Cho hai số nguyên dương A, B thỏa $10^4 \leq A < B \leq 10^5$, hãy tìm số lượng các biển số xe đẹp.

Input : file text **Dulieu.in** gồm 1 dòng chứa 2 số nguyên dương A và B.

Output : file **KQ.out** ghi một số nguyên là số lượng biển số xe đẹp tìm được.

Ví dụ : Dulieu.in KQ.out

11111 22222 23

1.4. Viết chương trình đoán số : người chơi sẽ đoán 1 số trong phạm vi từ 1 đến 100, tối đa 5 lần. Chương trình kiểm tra kết quả và xuất thông báo

hướng dẫn :

- Số bạn đoán lớn hơn
- Số bạn đoán nhỏ hơn
- Bạn đoán đúng
- Bạn đoán sai.

1.5. a/ Viết hàm đệ qui, không đệ qui tính hàm mũ x^n, với n nguyên.
b/ Viết hàm đệ qui, không đệ qui tính phần tử thứ n của hàm Fibonacci

1.6. Hiện đang lưu hành các tờ giấy bạc 500000 , 200000, 100000, 50000đ, 20000đ, 10000đ, 5000đ, 2000đ, 1000đ, 500đ. Nếu có x đồng, hỏi rằng nên chọn các tờ giấy bạc nào để số lượng các tờ giấy bạc là ít nhất. Nếu không đổi được , thỡ in ra màn hình 'Không đổi được'

1.7. Nhập họ tên, tách hoten ra làm 2 phần họ và tên riêng.

1.8. Viết chương trình con đổi các ký tự đầu của các từ trong 1 chuỗi ra chữ in, các ký tự còn lại ra chữ thường.

1.9. Viết chương trình con cho phép ta kiểm tra một password là đúng hay sai (nhập tối đa password 3 lần).

1.10. Cho chuỗi s, viết chương trình di chuyển chuỗi từ bên trái qua bên phải của màn hình tại dòng i, $1 \le i \le 20$. Quá trình di chuyển dừng lại khi ta ấn phím ESC.

1.11. Một buổi họp mặt đại gia đình nhân dịp cụ già Ted tròn 100 tuổi, người ta muốn sắp xếp con cháu của cụ theo thứ tự từ tuổi cao xuống thấp. Giả sử ta có thông tin về giấy khai sinh của từng người đó. Mỗi giấy khai sinh chỉ viết ba thông tin đơn giản gồm: *Tên người cha, Tên người con, Tuổi của người cha lúc sinh con*. Hãy giúp đại gia đình trên tính ra tuổi của từng người con cháu cụ Ted và viết ra danh sách theo thứ tự từ tuổi cao xuống thấp.

Input

Dòng đầu ghi số bộ test (không quá 100). Với mỗi bộ test:

 Dòng đầu tiên ghi số X (0<X<100) là số người con cháu cần sắp xếp.

 Tiếp theo là X dòng, mỗi dòng ghi thông tin về một giấy khai sinh của từng người (thứ tự ngẫu nhiên) gồm 3 thành phần, mỗi thành phần cách nhau một khoảng trống:

- Tên người cha: không quá 20 ký tự và không chứa khoảng trống
- Tên người con: không quá 20 ký tự và không chứa khoảng trống
- Tuổi của người cha khi sinh con: 1 số nguyên dương, không quá 100.

Output

 Với mỗi bộ test, in ra màn hình thứ tự bộ test (xem thêm trong bộ test ví dụ), sau đó lần lượt là từng người trong danh sách tuổi từ cao xuống thấp (không tính cụ Ted). Mỗi người viết ra hai thông tin: tên, một khoảng trống rồi đến tuổi của người đó.

 Nếu hai người có cùng tuổi thì xếp theo thứ tự từ điển.

Ví dụ

INPUT	OUTPUT
2 1 Ted Bill 25 4 Ray James 40 James Beelzebub 17 Ray Mark 50 Ted Ray 20	DATASET 1 Bill 75 DATASET 2 Ray 80 James 40 Beelzebub 23 Mark 30

1.12. Viết các chương trình con trả về thời gian thực thi (tính theo giây) của giải thuật Bubble Sort và Quick Sort trên dãy số có số phần tử khá lớn (số phần tử của dãy : 50000 số, 100000 số, 150000 số….)

1.13. Để xác định xem trong dãy A với n số nguyên có giá trị x hay không, ta thường dùng giải thuật sau:

```
int Search(int A[], int n, int x) {
   int i=0;
   while (A[i] != x) i++;
   return i==n?-1:i ;
}
```

T(n) = 1 + 4n + 2
=4n+3

Hãy cải tiến giải thuật trên sao cho việc tìm kiếm nhanh hơn .

1.14. Cho 1 số nguyên dương n. Viết các chương trình con :

a. Trả về tổng các chữ số của số nguyên n. Ví dụ: n=32045 thì tổng sẽ là 14

b. Trả về số m với các ký số là các ký số đảo ngược của số n. Ví dụ: n=32045 thì m sẽ là 54023

1.15. Digital root

"Digital root" của một số nguyên dương được tính toán bằng cách cộng tất cả các chữ số của số đó. Nếu kết quả là một chữ số thì đó là "Digital root". Nếu không, toàn bộ quá trình được lặp lại cho đến khi kết quả là một chữ số.

Ví dụ:

Số 24: 2+4 = 6, 6 là "Digital root" của 24

Số 39: 3+9 = 12, 1+2 = 3, 3 là "Digital root" của 39

Viết chương trình con trả về số Digital root của 1 số nguyên dương có độ dài không quá 100 chữ số.

1.16. Viết chương trình tìm ước số nguyên tố lớn nhất của một số nguyên (không quá 9 chữ số)

Input:

- Dòng 1 ghi số bộ test
- Mỗi bộ test gồm ghi trên một dòng số nguyên n

Output: với mỗi bộ test ghi ra ước số nguyên tố lớn nhất của số đó.

Ví dụ:

INPUT	OUTPUT
3	
15	5
1024	2
997	997

LEVEL 2

Câu 2.1. Cho dãy gồm n số tự nhiên phân biệt $a_1, a_2, .., a_n$ và số tự nhiên B. Hãy viết chương trình liệt kê tất cả các phần tử của tập $D = \left\{ (x_1, x_2, \cdots, x_n) : \sum_{i=1}^{n} a_i x_i = B, \quad x_i \in \{0,1\}, i = 1,2,...,n \right\}$;

Dữ liệu vào cho bởi file **data2.in** theo khuôn dạng như sau:

- Dòng đầu tiên ghi lại hai số tự nhiên n và B. Hai số được viết cách nhau bởi một vài khoảng trống.
- Dòng kế tiếp ghi lại n số nguyên dương $a_1, a_2,..,a_n$. Hai số khác nhau được viết cách nhau bởi một vài kí tự trống.o

Kết quả ra ghi lại trong file **ketqua2.out** theo khuôn dạng sau:

- Dòng đầu tiên ghi lại số tự nhiên k là số phần tử của tập D.
- k dòng tiếp theo mỗi dòng ghi lại một vector nhị phân $x = (x_1, x_2 ,.., x_n)$ là phần tử của D. Hai thành phần khác nhau của vector x được viết cách nhau bởi một vài khoảng trống.

Ví dụ với $n = 7, B = 25, \{ a_1, a_2, a_3, a_4, a_5, a_6, a_7\} = \{5, 10, 15, 20, 25, 30, 35\}$ trong file **Data2.in** sẽ cho ta *3* phần tử của tập D tương ứng với *3* vector nhị phân độ dài n trong file **ketqua2.out** dưới đây:

Data2.in

```
7     25
5     10     15    20    25    30     35
```

Ketqua2.Out

```
3
0  0  0  0  1  0  0
0  1  1  0  0  0  0
1  0  0  1  0  0  0
```

0

2.2. Cho trước một số nguyên, người ta sẽ làm tròn số này theo quy tắc sau:

- Nếu số đó lớn hơn 10 thì sẽ được làm tròn đến số hàng chục gần nhất
- Sau đó nếu kết quả lớn hơn 100 thì làm tròn đến số hàng trăm gần nhất
- Sau đó nếu kết quả lớn hơn 1000 thì làm tròn đến số hàng nghìn gần nhất
- …cứ tiếp tục như vậy …

Chú ý: giá trị 5 được làm tròn lên.

Hãy viết chương trình làm tròn số theo quy tắc trên.

Input

- Dòng đầu tiên chứa số n là số bộ test (không quá 100).
- n dòng tiếp theo, mỗi dòng ghi một số nguyên x với 0 <= x <= 99999999

Output

 Với mỗi bộ test, in ra màn hình trên một dòng kết quả của phép làm tròn

Ví dụ

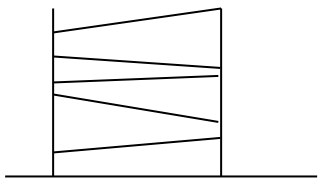
INPUT	OUTPUT
9	
15	20
14	10
4	4
5	5
99	100
12345678	10000000
44444445	50000000
1445	2000
446	500

2.3. Để tính biểu thức E = 1/1! + 3/2! + 5/3! + ... + (2n-1)/n! , ta có hàm sau:

```
float Tinh_E (int n)
{
        float E=0;
        long gt;
        for (int i=1; i<n; i++)
        {
          gt=1;
          for (int j=1; j<=n;   )  gt= gt *j;
          E= E + (2*i-1)/gt;
        }
}
```

a. Hãy tìm các lỗi trong hàm trên
b. Cải tiến thuật toán trên sao cho tối ưu về thời gian thực thi.

2.4. Một mảng chứa N cặp số thực (a_i, b_i) tương ứng với N đường thẳng $y_i = a_i x + b_i$. Các đường được sắp xếp theo đoạn [0,1] trên trục x với ý nghĩa sau: $y_i < y_{i+1}$ với mọi giá trị i giữa 0 và N-2, và với mọi giá trị của x trên [0,1]:



0 1

Làm thế nào để xác định nhanh 2 đường gần nhất "bao quanh" điểm P(x,y) , với P là điểm cho trước và $0 \le x_P \le 1$.

2.5. Cho 1 file text chứa các ký tự, trong đó có các chữ cái A .. Z. Viết chương trình con liệt kê tần suất xuất hiện của các ký tự chữ cái trong file text, biết rằng ta xem chữ cái in và chữ cái thường là như nhau.

2.6. Giả sử ta có danh sách liên kết First chứa n phần tử (n khá lớn và chưa xác định), mỗi phần tử chứa thông tin có kiểu Info_Type. Cho chương trình con sau **:**

```
void Ham (NODEPTR First)
{       NODEPTR p=First;
        int S, I ;
        for (S=0 ; p != NULL ; S++  ,   p= p->next ) ;
        Info_Type    *A = new  Info_Type [S];
        for (p=First  ,   I = 0  ; p != NULL ; A[I] = p->info , I++)
                p= p->next  ;
        for (p=First, I = S-1  ; p != NULL ; p->info = A[I]  , I--)
                p= p->next  ;
}
```

a. Hãy cho biết hàm trên thực hiện công việc gì ?

b. Tìm thuật toán thực hiện công việc trên sao cho tối ưu về thời gian thực thi, bộ nhớ.

2.7. Cho dãy số A có n số nguyên. Viết các chương trình con thực hiện các công việc sau:

a. Tách dãy thành 2 phần : số không âm nằm ở đầu dãy, số âm ở cuối dãy.
b. Sắp xếp từng phần không âm, âm theo thứ tự tăng dần.

2.8. Giả sử ta có 1 dãy A chứa 200000 số nguyên đang có thứ tự tăng dần, mỗi số nguyên trong dãy số có giá trị nằm trong khoảng từ 1 đến 100. Anh (chị) hãy khai báo dãy B mới, và viết chương trình con để ta có thể lưu trữ dãy số A trên vào dãy B sao cho tiết kiệm bộ nhớ nhất.

2.9. Một bãi đỗ xe nhận trông xe trong vòng một tháng. Mỗi xe sẽ được gắn một số hiệu là một số nguyên dương T ($10102010 \le T \le 10109999$). Hai xe khác nhau sẽ được gắn hai số hiệu khác nhau. Một xe có thể ra vào bãi đỗ xe nhiều lần, mỗi lần vào bãi đỗ xe, người trông xe sẽ ghi vào sổ sách số hiệu của chiếc xe đó.

Cuối tháng dựa vào sổ ghi chép, người trông xe làm thống kê về số lần vào bãi đỗ xe của từng chiếc xe để tiến hành thu phí. Nếu một chiếc xe vào bãi đỗ xe *p* lần, cuối tháng chủ xe phải trả một lượng phí được tính như sau:

$$C = \begin{cases} 100 \text{ , nếu } p <= 5 \\ 100 + (p-5) \text{ , nếu } p > 5 \end{cases}$$

Yêu cầu: Tính tổng số phí người trông xe thu được vào cuối tháng.
Dữ liệu: Vào từ file văn bản PARK.INP có dạng:

- Dòng đầu chứa 1 số nguyên dương k ($0 < k < 10^6$)
- k dòng tiếp theo, mỗi dòng chứa số hiệu 1 chiếc xe.

Kết quả: Đưa ra file văn bản PARK.OUT một số nguyên là tổng số phí thu được.
Ví dụ:

PARK.INP	**PARK.OUT**
7 10102010 10108888 10102010 10102010 10102010 10102010 10102010	201

2.10. Cho dãy gồm *N* số phân biệt $A_N = \{a_1, a_2, .., a_N\}$ và số tự nhiên *K* ($K \le N \le 100$). Ta gọi một *dãy con tăng dần bậc K* của dãy số A_N là một dãy các số gồm *K* phần tử trong dãy đó thỏa mãn tính chất tăng dần. Bài toán được đặt ra là hãy tìm số các *dãy con tăng dần bậc K* của dãy số A_N.

Input: Dòng đầu tiên ghi số bộ test, không lớn hơn 100. Mỗi bộ test được xây dựng theo khuôn dạng sau:

- Dòng đầu tiên ghi lại hai số *N* và *K* tương ứng với số phần tử của dãy số và bậc của dãy con.
- Dòng kế tiếp ghi lại *N* số của dãy số A_N, các số trong dãy không lớn hơn 100.

Output: Với mỗi bộ test, in ra màn hình số các dãy con tăng dần tự nhiên bậc *K* của dãy số A_N.
Ví dụ cho Input và Output:

INPUT (G.txt)	OUTPUT
2 5 3 2 5 15 10 20 5 3 2 20 10 15 5	7 0

2.11. Búp bê truyền thống của Nga được làm bằng gỗ với nhiều kích thước khác nhau. Búp bê con có thể được cho vừa vào trong búp bê mẹ. Búp bê (có chiều cao h1 và chiều rộng w1) được cho vừa vào búp bê khác (có chiều cao h2 và chiều rộng w2) khi và chỉ khi h1 < h2 và w1 < w2.

Yêu cầu: Cho n búp bê, hãy nhét các búp bê vào nhau sao cho số búp bê còn lại là ít nhất

Dữ liệu vào: Dòng đầu tiên ghi số n là số bộ dữ liệu (0 < n < 20) Mỗi bộ dữ liệu bắt đầu bằng số m (0 < m < 20000) được ghi trên 1 dòng. Dòng tiếp theo ghi 2m số nguyên w1, h1, w2, h2,… ,wm, hm trong đó wi là chiều rộng và hi là chiều cao của búp bê thứ i (0<wi, hi<10000) với mọi i.

Dữ liệu ra: Với mỗi bộ dữ liệu in ra số lượng búp bê còn lại ít nhất. Mỗi kết quả in ra trên 1 dòng.

Dữ liệu vào	Dữ liệu ra
4 3 20 30 40 50 30 40 4 20 30 10 10 30 20 40 50 3 10 30 20 20 30 10 4 10 10 20 30 40 50 39 51	 1 2 3 2

2.12. Trong bài toán này, bạn sẽ được cung cấp một danh sách bao gồm cả chữ và số. Mục tiêu là làm thế nào để sắp xếp danh sách đó sao cho các chữ theo thứ tự alphabet và các số theo thứ tự tăng dần. Một yêu cầu nữa là nếu thành phần tại vị trí n là một số thì nó vẫn phải là số sau khi sắp xếp và ngược lại một chữ vẫn phải là chữ.

Input:

Input là 1 danh sách, mỗi danh sách là một dòng (không quá 100 ký tự). Mỗi thành phần của danh sách sẽ được tách bởi một dấu phẩy và một dấu cách. Danh sách sẽ kết thúc bằng một dấu chấm.

Output:

Với danh sách trong Input, in ra danh sách sau khi đã sắp xếp, tách giữa các thành phần là một dấu phẩy và một dấu cách

Input

banana, strawberry, OrAnGe.
Banana, StRaWbErRy, orange.
10, 8, 6, 4, 2, 0.
x, 30, -20, z, 1000, 1, Y.
50, 7, kitten, puppy, 2, orangutan, 52, -100, bird, worm, 7, beetle.

Output

banana, OrAnGe, strawberry.
Banana, orange, StRaWbErRy.
0, 2, 4, 6, 8, 10.
x, -20, 1, Y, 30, 1000, z.
-100, 2, beetle, bird, 7, kitten, 7, 50, orangutan, puppy, 52, worm.

2.13. Ngày nay, việc sử dụng bàn phím điện thoại di động để nhấn các số đã trở thành một việc rất quen thuộc với các bạn sinh viên. Ai cũng biết các phím số trên điện thoại cũng là các phím dùng để nhấn các chữ cái:
`2: ABC, 3: DEF, 4: GHI, 5: JKL, 6: MNO, 7: PQRS, 8: TUV, 9: WXYZ`
Nam viết ra giấy một dãy ký tự và đố Bình xác định đó là dãy số nào theo cách nhấn số trên điện thoại (chỉ xem xét sự tương ứng giữa số và ký tự chứ không xem xét phải nhấn bao nhiêu lần phím đó, ví dụ cả A, B, C đều là một số 2).
Bình rất nhanh chóng xác định được kết quả, không những thế Bình còn muốn xác định nhanh xem số đó có phải là số dạng thuận nghịch hay không. Một số là thuận nghịch nếu viết theo thứ tự ngược lại cũng là chính nó. Hãy viết chương trình giúp Bình thực hiện công việc trên.

Input

 Dòng đầu tiên chứa số n là số bộ test (không quá 1000).
 Mỗi bộ test viết trên một dòng một dãy ký tự gồm các chữ cái có thể là chữ hoa hoặc chữ thường, dài không quá 20 ký tự, không có khoảng trống.

Output

 Với mỗi bộ test, in ra màn hình, trên một dòng, chữ "YES" nếu đó tương ứng là số thuận nghịch, chữ "NO" nếu ngược lại.

Ví dụ

INPUT	OUTPUT

```
2                   YES
ANBOBNA             NO
iAmACoolCompany
```

2.14. Cho đồ thị *vô hướng* $G = <V, E>$ được biểu diễn dưới dạng ma trận kề trong tệp ***dothi.in*** theo khuôn dạng :

- Dòng đầu tiên ghi số tự nhiên N ($N \leq 100$) là số đỉnh của đồ thị ;
- N dòng kế tiếp ghi ma trận kề của đồ thị G, hai phần tử khác nhau của ma trận kề được ghi cách nhau một vài khoảng trống.

Hãy viết chương trình kiểm tra đồ thị G có liên thông hay không? Nếu G không liên thông thì ghi vào tệp ***ketqua.out*** thông báo " ***NO".*** Nếu G liên thông thì tìm tất cả các đỉnh trụ của đồ thị. Ghi lại các đỉnh trụ tìm được vào tệp ***ketqua.out*** theo khuôn dạng:

- Dòng đầu tiên ghi lại số nguyên **m** là số các đỉnh trụ tìm được ;
- Trong trường hợp m > 0, **m** dòng tiếp theo mỗi dòng ghi lại một đỉnh trụ.

Ví dụ với tệp ***dothi.in*** dưới đây, chương trình sẽ cho ta hai đỉnh trụ trong tệp ***ketqua.out***.

dothi.in

```
5
0 1 1 0 0
1 0 1 0 1
1 1 0 1 0
0 0 1 0 0
0 1 0 0 0
```

ketqua.out

```
2
2
3
```

2.15. Viết các CTC sau:

- Tìm số nguyên tố nhỏ nhất lớn hơn mọi giá trị trong mảng nguyên
- Hãy tìm ước chung lớn nhất của tất cả ptử trong mảng nguyên
- Hãy tìm bội số chung nhỏ nhất trong mảng nguyên

2.16. Cho **k** số nguyên dương : **a1, a2, a3, ...ak (0<k<50)** và một số nguyên dương **N**. Điền vào phép toán cộng **(+)** hoặc trừ **(-)** thích hợp vào dấu **(?)** cho biểu thức sau (nếu có lời giải) : **a1 (?) a2 (?) a3 (?) ... (?) ak = N**.

Ví dụ: **7 (?) 2 (?) 5 (?) 10 = 10 => 7 - 2 - 5 + 10 = 10**

LEVEL 3

3.1. Để chọn ra M số nguyên ngẫu nhiên không trùng nhau từ 1..N với M < N, ta có đoạn mã giả sau:

Khởi động dãy A rỗng
$Size_A$ = 0
while $Size_A$ < M
{
T=RandInt(1,N); // lấy số nguyên ngẫu nhiên từ 1 đến N
if (T ∉ A) A[$Size_A$++] = T;
}

a/ Cài đặt chương trình con TaoDayNgauNhien (int M, int N) theo thuật toán trên , trong đó A, $Size_A$ là 2 biến toàn cục. Tính T(N)

b/ Cải tiến lại thuật toán trên sao cho tối ưu về thời gian khi M,N khá lớn và M gần bằng N

3.2. Trong trường hợp ta muốn tạo dãy số 20000 số nguyên phân biệt khác nhau trong miền giá trị [1..30000], sau đó sắp xếp dóy số theo thứ tự tăng dần thì giải thuật tối ưu về mặt không gian và thời gian sẽ thực hiện như thế nào? Cho biết thời gian thực thi của giải thuật Quick Sort và giải thuật bạn cài đặt.

3.3. Giả sử ta có 1 vecto X chứa N số nguyên . Ta định nghĩa vecto con của X là vecto mà thành phần của nó là các thành phần liên tiếp trong X (vecto rỗng có thể xem là vecto con của X). Ta gọi tổng của 1 vecto là tổng của tất cả các thành phần của nó. Hãy cải tiến chương trình tìm ***tổng lớn nhất*** trong tất cả các tổng của các vecto con của X sao cho thực hiện nhanh hơn.

```
long  Max = -MAXINT , i, j, k, Sum;
for ( i=1; i<= N ; i++ )
{  for (j = i; j <= N ; j++)
        {   Sum = 0;
            for  (k=i ; k <= j ; k++)  Sum = Sum + X[k];
        }
 Max = (Max < Sum ? Sum : Max) ;
}
```

3.4. Cho 1 dãy số nguyên có n số (100000 ≤ n ≤ 1000000), viết chương trình con tìm nhanh 1 số nguyên x có trong dãy số đó hay không. Nếu có x trong dãy, chương trình con sẽ trả về vị trí của x trong dãy, ngược lại chương trình con sẽ trả về -1.

3.5. Một hệ thống phòng thủ của địch gồm N điểm (N≤100), giữa các điểm bất kỳ của hệ thống đều có thể đi lại trực tiếp hoặc gián tiếp với nhau thông qua hệ thống các đường hầm. Bài toán được đặt ra là cho trước một hệ thống phòng thủ, hãy giúp bộ đội sử dụng đúng 1 quả pháo bắn vào một trong N điểm sao cho hệ thống bị chia cắt thành nhiều mảnh nhất.

Input: Dòng đầu tiên ghi số bộ test, không lớn hơn 100. Mỗi bộ test được tổ chức theo khuôn dạng sau:

- Dòng đầu tiên ghi lại số tự nhiên N không lớn hơn 100 là số điểm của hệ thống phòng thủ.
- Những dòng kế tiếp ghi lại ma trận $G = (g_{ij})$ biểu diễn hệ thống phòng thủ, trong đó $g_{ij}=0$ được hiểu là không có đường hầm trực tiếp nối giữa điểm i và j, $g_{ij}=1$ được hiểu có đường hầm nối trực tiếp giữa điểm i và điểm j ($1 \le i, j \le N$).

Output: Với mỗi bộ test, in ra màn hình trên một dòng một số duy nhất là đỉnh bị phá hủy thỏa mãn yêu cầu bài toán (nếu có nhiều đỉnh cùng thỏa mãn yêu cầu thì in ra đỉnh có giá trị nhỏ nhất). Nếu không thể chia cắt được hệ thống, hãy in ra số 0.

Ví dụ cho Input và Output:

INPUT	OUTPUT
2 **5** **0 1 1 0 0** **1 0 1 0 0** **1 1 0 1 1** **0 0 1 0 0** **0 0 1 0 0** **5** **0 1 1 0 0** **1 0 1 0 1** **1 1 0 1 1** **0 0 1 0 1** **0 1 1 1 0**	3 0

3.6. Tìm hiểu thuật toán nén dữ liệu Huffman và cài đặt

3.7. Cho một xâu ký tự S gồm n chữ số 0, các ký tự trong xâu S được đánh số từ 1 tới n theo thứ tự từ trái qua phải. Xét lệnh Fill(i,j,c) : Trong đó i, j là các số nguyên dương, $1 \le i \le j \le n$, và c là một chữ số $\in \{0,1,2,\ldots,9\}$: Điền ký tự c vào xâu S bắt đầu từ vị trí i tới vị trí j. Các chữ số mới điền vào sẽ đè lên các chữ số đang có trong xâu S.

Ví dụ với n=6:

Fill(4,6,5) : 000000 → 000555

Fill(1,3,1) : 000555 → 111555

Cho biết trước m lệnh Fill và thứ tự thực hiện của chúng. Với một số nguyên dương k < n, hãy xóa đi k ký tự trong xâu S (sau m lệnh Fill đã cho) để được một xâu T gồm n-k ký tự là biểu diễn thập phân của một số lớn nhất có thể.

Dữ liệu vào:

- Dòng đầu tiên ghi số bộ dữ liệu
- Mỗi bộ dữ liệu gồm m+1 dòng trong đó
- Dòng 1 chứa ba số nguyên dương n, m, k (k < n)
- m dòng tiếp theo, dòng thứ p chứa ba số nguyên i_p, j_p,c_p cho biết lệnh Fill thứ p là Fill (i_p, j_p,c_p) ($1 \le i_p \le j_p \le n$; $0 \le c_p \le 9$)

Các số trên một dòng được ghi cách nhau ít nhất một dấu cách

Kết quả: Ghi ra xâu T tìm được

Ví dụ

Dữ liệu vào	Dữ liệu ra
1 **6 3 2** **4 6 5** **1 3 1** **3 4 9**	**9955**

3.8. Cho 2 mảng Lenhmua chứa các lệnh mua và Lenhban chứa các lệnh bán của cổ phiếu có mã là ABC. Mỗi lệnh có các thông tin sau : loại lệnh (ATO,LO), số lượng, giá đặt, giờ (giá đặt phải trong khoảng giá sàn=8000 và giá trần=10000). Viết chương trình xác định giá khớp tốt nhất khi có lệnh đặt vào hệ thống; giá khớp tốt nhất là giá mà có số lượng cổ phiếu khớp nhiều nhất. Nguyên tắc khớp lệnh:

- Ưu tiên về giá : giá bán thấp sẽ ưu tiên khớp trước;
  giá mua cao sẽ ưu tiên khớp trước;
  Lệnh mua ATO sẽ ưu tiên khớp trước lệnh mua theo giá trần.
  Lệnh bán ATO sẽ ưu tiên khớp trước lệnh bán theo giá sàn.
- Ưu tiên về thời gian: nếu lệnh cùng giá thì lệnh đặt trước sẽ ưu tiên khớp trước.

3.9. Ma trận con lớn nhất

Cho ma trận vuông (mảng 2 chiều) gồm các số nguyên (cả âm và dương). Một "ma trận con" trong ma trận đã cho là bất cứ mảng 2 chiều nào (kích thước 1X1 hoặc lớn hơn) nằm trong ma trận đó. Ma trận con được gọi là lớn nhất nếu tổng các phần tử trong ma trận con đó là lớn nhất so với tất cả các ma trận con khác.

Ví dụ: ma trận

```
 0 -2 -7  0
 9  2 -6  2
-4  1 -4  1
-1  8  0 -2
```

Có ma trận con lớn nhất là:

```
 9  2
-4  1
-1  8
```

Với tổng là 15.
Hãy viết chương trình xác định tổng ma trận con lớn nhất.
Input: Chỉ có 1 bộ test duy nhất.
Dòng đầu tiên ghi số n là cỡ của ma trận (không quá 100). Tiếp theo là n2 số nguyên, các số cách nhau các khoảng trống hoặc xuống dòng. Các số nguyên trong khoảng từ -127 đến 127.
Output:
Ghi ra giá trị tổng của ma trận con lớn nhất.
Ví dụ:

INPUT	**OUTPUT**
4 0 -2 -7 0 9 2 -6 2 -4 1 -4 1 -1 8 0 -2	15