1 Organizador de Vagões

 (++)

Na estação de trem você ainda pode encontrar o último dos "*organizadores de vagões*". Um organizador de vagões é um empregado cujo trabalho é apenas o de *reordenar* os vagões do trem, trocando-os de posição. Uma vez que os vagões são organizados em uma ordem considerada ótima, o condutor pode desconectar cada vagão e colocá-los na estação. O título "*organizador de vagões*" é dado à pessoa que realiza esta tarefa, cuja estação fica perto de uma ponte. Ao invés da ponte poder subir ou descer, ela roda sobre um pilar que fica no centro do rio. Após rodar 90^{o}, os barcos podem passar na esquerda ou direita dela.
O primeiro "*organizador de vagões*" descobriu que girando a ponte em 180^{o} graus com dois vagões em cima dela, é possível a troca de lugar entre os dois vagões. Obviamente a ponte pode operar no máximo com dois vagões sobre ela.
Agora que quase todos os "*organizadores de vagões*" já faleceram, a estação gostaria de automatizar esta operação. Parte do programa a ser desenvolvido é uma rotina que decide, para um dado trem com um determinado número de vagões, o número de trocas entre trens adjacentes que são necessárias para que o trem fique ordenado.
Você já sabe: sua tarefa é criar tal rotina.

Entrada

A entrada contém na primeira linha o número de caso de testes $n \in \mathbb{N}^*$. Cada par de linhas seguintes representa um caso de teste, sendo que a primeira linha do par contém um inteiro L, determinando o tamanho do trem $(1 \leq L \leq 1000)$, ou seja, o número de vagões nele presente. A segunda linha do par contém uma permutação qualquer dos naturais de 1 até L, indicando a ordem atual dos vagões.
Os vagões devem ser ordenados de forma que o vagão 1 venha por primeiro, depois o vagão 2, e assim sucessivamente, até que o último vagão seja aquele com o número L.

Saída

Para cada caso de teste imprima a sentença (numa linha):
'Número ideal de trocas é de S troca(s).' onde S é um número natural.

Exemplo

Entrada	Saída
3 3 1 3 2 4 4 3 2 1 2 2 1	Número ideal de trocas é de 1 troca(s). Número ideal de trocas é de 6 troca(s). Número ideal de trocas é de 1 troca(s).

1 Separando Pares de Ímpares

 (++)

Tales, um menino muito levado, pegou na escola uma caixa repleta de números naturais impressos em cartelas de EVA (*Espuma Vinílica Acetinada*) e derrubo-os sobre o chão da sala. Por estranho que pareça, ao cairem os números formaram uma *fila indiana* de tal maneira que ficaram com seus valores distribuídos aleatoriamente nesta fila.
Sabe-se que na caixa havia $n \in \mathbb{N}^*$ números, mas seus valores são desconhecidos.
Você deverá ordenar esta fila, segundo as seguintes regras:

- primeiro devem vir todos os números pares, em ordem crescente;
- depois devem vir os números ímpares, em ordem decrescente.

Entrada

A primeira linha de entrada contém um único inteiro positivo n, com $(1 < n < 100)$, indicando a quantidade de números existente na caixa que Tales derrubou.
As próximas n linhas conterão, cada uma delas, um número natural.

Saída

Apresente todos os valores lidos na entrada segundo a ordem descrita anteriormente. Cada número deve ser impresso em uma linha, conforme exemplo abaixo.

Exemplo

Entrada	Saída
10 4 32 34 543 3456 654 567 87 6789 98	4 32 34 98 654 3456 6789 567 543 87

1 Altura

 (+)

Sempre cheio de "*boas ideias*", agora o governo brasileiro resolveu criar a "bolsa altura", subsidiado na intenção de que as futuras gerações de brasileiros sejam mais altas, em média, que a atual. Desta forma, você, depois de ser aprovado num árduo concurso, você faz parte da equipe que realizará o levantamento da altura da população de todas as cidades brasileiras.
Você foi designado para realizar a tarefa num conjunto de cidades com 10000 ou menos habitantes, sabendo-se que você terá que coletar a altura de *todos* os habitantes da cidade e que ninguém, segundo o IBGE, tem mais do que 230cm de altura nestas cidades.

Entrada

A entrada contém vários casos de teste.
A primeira linha de entrada contém um inteiro η_C $(\eta_C < 100)$ que indica a quantidade de casos de teste, ou seja, o número de cidades.
Cada caso de teste é formado por duas linhas: a primeira linha conterá um inteiro $n\left(1 \leq n \leq 10^4\right)$, que indica a quantidade de pessoas daquela cidade. A segunda linha apresenta a altura de cada uma destas pessoas, em centímetros, representado pela letra $h\,(20 \leq h \leq 230)$ e separados por um espaço em branco.

Saída

Deve-se imprimir, por caso de teste, uma linha contendo os valores das alturas de todos os moradores da cidade (em cm), em ordem crescente de altura, separados por um espaço em branco.

Exemplo

Entrada	Saída
6 10 65 31 37 37 72 76 61 35 57 37 12 45 186 185 55 51 51 22 78 64 26 49 21 10 20 93 203 67 64 225 112 81 58 180 8 169 189 220 228 68 32 214 180 6 133 55 67 166 112 41 4 39 38 120 55	31 35 37 37 37 57 61 65 72 76 21 22 26 45 49 51 51 55 64 78 185 186 20 58 64 67 81 93 112 180 203 225 32 68 169 180 189 214 220 228 41 55 67 112 133 166 38 39 55 120

1 PLACAR - Quem vai ser reprovado?

 (+++)

O professor Nala Gnirut aplicou, para seus alunos, uma *Lista de Exercícios* contendo um conjunto de dez "*problemas*" e concedeu o prazo de um mês para que eles os resolvessem.
No final do mês os alunos mandaram o número de problemas resolvidos corretamente. A promessa do brilhante didata era reprovar, sumariamente, o último colocado desta competição, onde os alunos seriam ordenados conforme o número de problemas resolvidos, com empates resolvidos de acordo com a ordem alfabética dos nomes[1].
Este padrão fez com que alunos com nomes iniciados nas últimas letras do alfabeto se esforçassem muito nas tarefas, e não compartilhassem suas soluções com colegas (especialmente aqueles cujos nomes começassem com letras anteriores às deles).
Você é o "*estudante monitor*" do professor Nala e sua tarefa, nesta questão, é escrever um programa de computador que leia os resultados dos alunos do professor e imprima a classificação dos alunos.

Entrada

A primeira linha de cada instância consiste num número natural $n\,(1 \leq n \leq 100)$ que indica a quantidade de alunos na competiação.
Cada uma das n linhas seguintes contém o nome do aluno – sem espaço entre as suas palavras formadoras – e o número de problemas resolvidos por ele(a). O número de problemas resolvidos está, obviamente, entre 0 e 10, inclusive extremos.

Saída

O programa de computador criado por você deverá imprimir o nome do(a) aluno e o número de exercícios feitos pelo aluno, ordenadamente.

[1]Suponha, para simplificação, que não há homônimos na turma e que todos os nomes dos estudantes são escritos utilizando uma única palavra, sem espaços em branco entre elas, e que terá, no máximo, 20 símbolos. Por exemplo: AnaLuiza, Guilherme, LuizFelipe, Marcelo, PedroAugusto, etc.

Exemplo

Entrada	Saída
4 cardonha 9 infelizreprovado 3 marcel 9 infelizaprovado 3	cardonha 9 marcel 9 infelizaprovado 3 infelizreprovado 3

1 Insertion - Selection

 (+)

Escreva um programa de computador que a partir de um vetor de números naturais fornecido como entrada calcule a diferença entre o número de trocas realizadas pelos algoritmos `insertionSort` e `selectionSort`, nesta ordem.
Cada movimentação efetiva de um número no vetor deve ser contabilizada. O `selectionSort` deve ser implementado com a otimização que realiza o menor número de trocas possível.

Entrada

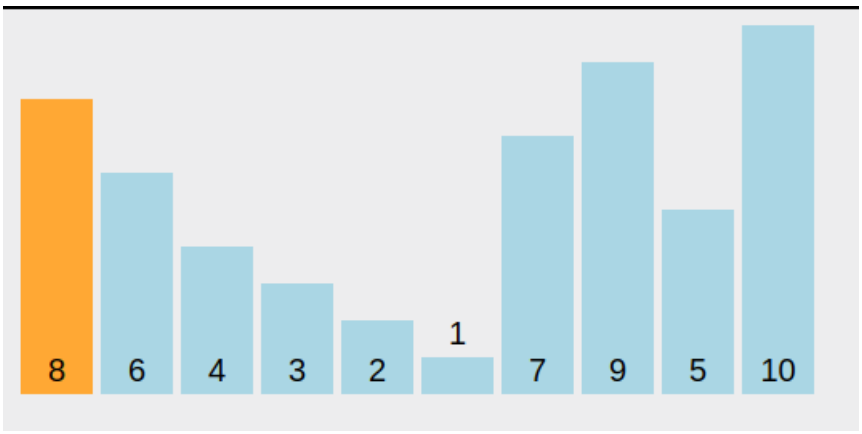
A primeira entrada é um número natural n, $(1 \leq n \leq 100)$, que representa o tamanho do vetor de entrada. As próximas n linhas são os elementos do vetor, sempre fornecidos da primeira posição até a última.

Saída

A saída consiste em um número inteiro que corresponde à diferença entre o número de trocas realizadas pelo `insertionSort` e pelo `selectionSort`, nesta ordem.

Exemplo

Entrada	Saída
10 8 6 4 3 2 1 7 9 5 10	23



1 Ordenação

 (+)

Escreva um programa, empregando a linguagem $\mathbb{C}$, para a partir de dois números naturais dados, ℓ e k, com $0 < \ell < k \leq n$, imprima a soma do ℓ-ésimo menor elemento do vetor com o k-ésimo menor elemento do vetor, considerando-se que o referido vetor possui n elementos.

Entrada

A primeira linha da entrada contém o número natural n, que representa o tamanho do vetor.
A segunda linha da entrada contém os n elementos do vetor, sempre apresentados a partir da primeira posição até a última posição, ou seja, das posições $1, 2, 3, \ldots, n$, nesta ordem.
Por fim, a terceira linha da entrada contém os valores de ℓ e k.

Saída

Seu programa deve imprimir, numa única linha, o valor a soma do ℓ-ésimo menor elemento do vetor com o k-ésimo menor elemento do vetor.

Exemplo

Entrada	Saída
9 9 8 7 6 5 4 3 2 1 3 4	7

Entrada	Saída
6 1 2 3 6 4 4 5 2	6

Alice In Numberland

Prime Numbers and Arrays

 (++)

Consider a array of natural numbers v, with size expressed by $n \in \mathbb{N}^*$. The natural numbers are distributed in it totally randomly, that is, there is no apparent order.
Each position of the vector v corresponds to the *order* of a particular prime number, where 2 is the prime number of order 1, 3 is the prime number of order 2 and so on.
A computer program must be developed: it will be able to print the prime numbers corresponding to the elements of v, showing them in ascending order.

Input

The first line of the entry will contain the value of n $(1 \leq n \leq 1000)$.
The second line will contain the values of the elements of v, always displayed from the first element towards the last element.

Output

The output should display, in just one line, the prime numbers corresponding to the elements of v, in ascending order.

Examples

Entrada	Saída
10 10 2 7 8 5 4 3 1 6 9	2 3 5 7 11 13 17 19 23 29

Entrada	Saída
7 35 140 18 150 28 59 15	47 61 107 149 277 809 863

1 Máquina de Café

 (+++)

Brasilino, dono de uma sofisticada cafeteria na centro da cidade, no dia da Independência do Brasil, decidiu distribuir cafés grátis como campanha de *marketing* durante todas as 24h deste dia.
Ele decidiu instalar p máquinas de café na sua cafeteria, cada uma das máquinas pode servir uma pessoa por minuto. Ele recebeu a lista de n pessoas que irão visitar sua cafeteria. O momento da chegada de cada pessoa é denotado por dois números naturais (h,m), onde h é a hora da visita e m é o minuto da visita naquela hora[1].
Devido à tecnologia, no século XXI somos todos *impacientes*, e, assim, se uma pessoa tiver que esperar para obter o seu café após sua chegada na cafeteria, a pessoa fica **brava**. Brasilino está preocupado com o custo desta campanha de *marketing* e quer configurar o número mínimo de máquinas de café para que nenhuma pessoa fique brava durante sua estada na cafeteria, sabendo-se que uma máquina leva exatamente um minuto para servir o café[2].
Você, como exímio programador de computadores, recebeu a tarefa de ajudá-lo a encontrar o número mínimo de máquinas de café necessárias para que todos os participantes da campanha de *marketing* fiquem felizes (ou seja, não **bravos**).

Entrada

A primeira linha da entrada contém $t \in \mathbb{N}^*$, o número de casos de testes, com $1 \leq t \leq 100$.
Cada grupo de $(n+1)$ linhas seguintes corresponde a um caso de teste, onde n representa o número de pessoas a serem servidas naquele dia. A primeira linha de cada caso de teste contém o valor de n, com $n \in \mathbb{N}^*$ e $n < 10.000$. As n linhas seguintes contém dois números naturais, h e m, separados por um espaço em branco, representando cada cliente específico.

[1] O horário definido por h:m representa o exato momento em que a pessoa entra na cafeteria e se coloca à frente de uma das máquinas de café disponíveis, onde $0 \leq h \leq 23$ e $0 \leq m \leq 59$.

[2] Suponha, para simplificação, de que não há a menor possibilidade de que qualquer uma das máquinas instaladas apresente defeito durante todo o dia.

Saída

Para cada caso de teste fornecido, o programa deve imprimir, numa única linha, o número mínimo de máquinas de café necessários para atender ao problema anteriormente definido.

Restrições

Lembre-se: Todas as restrições a seguir devem ser satisfeitas.

Exemplos

Entrada	Saída
1 7 10 20 5 40 10 20 23 11 5 50 10 30 17 12	2

Entrada	Saída
2 8 8 0 9 10 9 11 8 0 8 0 16 40 8 0 16 41 10 13 0 13 1 13 2 13 3 13 0 13 1 13 0 13 0 13 0 13 4	4 5

1 Olimpíadas

 (+++)

O Comitê Olímpico Internacional (COI) está visitando as cidades candidatas a sediar as Olimpíadas de 2032 e, desta vez, Goiânia é uma das cidades concorrentes, mas a competição entre as cidades candidatas está muito acirrada.

O COI tem um conjunto de exigências que devem ser obedecidas pelas cidades candidatas, como boas *arenas* para os jogos (ginásios, campos de futebol, pistas de atletismo, parque aquático,...), bons alojamentos, um plano para o tráfego de veículos durante os jogos, etc.

Durante sua visita à Goiânia, o COI colocou ainda mais uma exigência: a demonstração da qualidade dos sistemas de informática. Especificamente, o COI quer que a organização local demonstre a sua capacidade em informática produzindo um programa de computador que gere a classificação final dos países, considerando o número de medalhas recebidas pelos atletas de cada país.

Para ser justo, o COI abriu um concurso que permite que equipes de estudantes de quaisquer faculdades, centros universitários e universidades apresentem seus *programas de computador* para cumprir a tarefa. O programa que melhor resultado obtiver numa *bancada de testes* a ser proposto pelo COI, em momento oportuno, será o escolhido.

Tarefa

Você está na equipe que o INF/UFG designou para *vencer* este concurso. Sua tarefa é escrever um programa que, dada a informação dos países que receberam medalhas de ouro, prata e bronze em cada modalidade, gere a lista de classificação dos países na competição.

Nesta tarefa, os países serão identificados por números inteiros $(1, 2, 3, \ldots, n)$.

O melhor colocado deve ser o país que conseguiu o maior número de medalhas de ouro. Se houver empate nestas medalhas, o melhor colocado é o país que conseguiu o maior número de medalhas de prata. Se houver empate nestas medalhas, o melhor colocado é o país que recebeu o maior número de medalhas de bronze. Por fim, se ainda assim houver empate, o melhor classificado é o que tem o menor número de identificação.

Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (normalmente o teclado).

A primeira linha da entrada contém dois números naturais n e m, separados por um espaço em branco, e indicando, respectivamente, o número de países $(1 \leq n \leq 100)$ e número de modalidades esportivas envolvidas na competição $(1 \leq m \leq 50)$. Os países são identificados por numeros inteiros de 1 a n.

Cada uma das m linhas seguintes contém três números inteiros O, P e B, separados por um espaço em

branco, representando os identificadores dos países cujos atletas receberam, respectivamente, medalhas de ouro $(1 \leq O \leq n)$, prata $(1 \leq P \leq n)$ e bronze $(1 \leq B \leq n)$. Assim, se uma das m linhas contém os números 3 2 1, significa que nessa modalidade a medalha de outro foi ganha pelo país cuja identificação é 3, a de prata pelo país cuja identificação é 2 e a de bronze pelo país cuja identificação é 1.

Saída

Seu programa deve imprimir, na saída padrão, uma linha contendo n números, separados por um espaço em branco, representando os países na ordem decrescente de classificação (o primeiro número representa o país que é o primeiro colocado, o segundo número representa o país que é o segundo colocado, e assim por diante).

Exemplos

Entrada	Saída
2 2 2 1 2 1 2 2	2 1

Entrada	Saída
4 3 3 2 1 4 3 1 4 3 1	4 3 2 1

Entrada	Saída
3 3 3 1 2 2 3 1 1 2 3	1 2 3

1 Corrida Escolar

 (+++)

A escola de Joãozinho tradicionalmente organiza uma corrida ao redor do prédio principal.
Apesar, evidemente, de haver estudantes dos mais variados períodos, todos os alunos da escola são convidados a participar, sendo impossível que todos participem da mesma corrida.
Para contornar esse problema, os professores cronometram o tempo que cada aluno consome para dar cada uma das voltas ao redor do prédio principal, e depois comparam os tempos para descobrir a classificação final.

Tarefa

Sua tarefa é, sabendo o número de competidores, o número de voltas de que consistiu a corrida e os tempos de cada aluno competidor, descobrir quem foi o aluno vencedor, para que ele possa receber uma medalha comemorativa.

Entrada

A primeira linha da entrada contém dois naturais, n e m, representando o número de competidores e o número de voltas da corrida, respectivamente, sendo que $(1 \leq n \leq 50)$ e $(1 \leq m \leq 5)$.
Na sequência, cada linha representa um competidor[1], sendo que nela haverá m números naturais, cada um indicando o tempo consumido em cada uma das voltas dadas por aquele competidor. O tempo t é expresso em segundos e, por isso, garante-se que não houve dois competidores que gastaram o mesmo tempo para completar a corrida inteira, ou seja, o tempo total de todas as suas voltas[2]. Sabe-se também que $1 \leq t \leq 1000$.

[1] Haverá, portanto, n linhas representando os competidores.
[2] Considera-se, para simplificação, que nenhum dos competidores desistiu da corrida, ou seja, todos concluiram as m voltas da corrida da qual participou.

Saída

A saída consiste de um único natural, que corresponde ao número do competidor que ganhou a corrida.

Exemplos

Entrada	Saída
2 1 5 7	1

Entrada	Saída
3 3 3 5 6 1 2 3 1 1 1	3