Problem A. AAAAAAdenilação

Input file:	`entrada padrão`
Output file:	`saída padrão`
Time limit:	1 segundo
Memory limit:	256 megabytes

A poliadenilação é a adição de uma cauda poli(A) ao RNA, geralmente o RNA mensageiro (mRNA). A cauda poli(A) é composta de uma sequência de bases adeninas e fica na extremidade que é transcrita por último, a extremidade 3'. Ela é importante para a estabilidade do mRNA pois permite que o RNA mensageiro sobreviva mais tempo sem ser digerido acidentalmente pela própria célula.

O aperfeiçoamento no entendimento desses e outros processos biológicos nos permitiu hoje a criação de vacinas que utilizam impressoras de DNA que criam fitas de DNA que são transcritas para fitas mRNA que ultimamente são encapsuladas em um frasco. Assim, uma das otimizações que se foi buscada é o tamanho ideal da cauda poli(A) para permitir que a vacina fosse mais eficaz.

Ariel teve interesse em descobrir qual o tamanho do cauda poli(A) de alguns códigos RNA, e acabou descobrindo que é possível colocar códigos ligadores no meio do código do poli(A), e que podem existir outras caudas, como o poli(C) após a cauda de poli(A). Então para contabilizar o tamanho da cauda corretamente, Ariel decidiu que a cauda poli(A) é a maior subsequência que começa com 7 `'A'`s, e termina com 7 `'A'`s, independente de onde esteja.

No código da vacina para SARS-CoV-2 da BioNTech/Pfizer por exemplo, é possível encontrar um ligador de tamanho 10 AGCAUAUGACU dentro do poli(A), que deixa a junção dos pedaços de DNA sintético impressos seja mais fácil e também faz com que o mRNA resultante tenha um comprimento mais uniforme.

Ajude Ariel então a encontrar o tamanho da cauda poli(A) dado o código RNA.

Input

A primeira linha contém um inteiro N ($1 \leq N \leq 10^6$), o tamanho do código de RNA.

Segue uma linha com N caracteres `'G'`, `'A'`, `'U'` ou `'C'`, representando o código RNA.

Output

Imprima o tamanho da cauda poli(A), ou 0 se não há cauda poli(A).

Examples

entrada padrão
75 GAGAUUGCUAGCAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGCAUAUGACUAAAAAAAAAAAAAAAAAAAAAA
saída padrão
63

entrada padrão
75 GAGCAGAUUAAUAAAAAAAAAAAAAAAAAAAAAAAAAAUGCAUCCCCCCCCCCCCCCCCCCCAAAGGCUGCACCAGAAUU
saída padrão
23

entrada padrão
60 AUGGAUUCUAACACUGUGUCAAGUUUUCAGGUAGAUUGCUUCCUUUGGCAUGUCCGAAAA
saída padrão
0

Problem B. Chattes

Input file:	`entrada padrão`
Output file:	`saída padrão`
Time limit:	1 segundo
Memory limit:	256 megabytes

O Chattes é uma plataforma acadêmica onde pesquisadores podem compartilhar o seu currículo de forma unificada com outros pesquisadores. Lá pode ser encontrado o histórico acadêmico, os trabalhos científicos e as citações de um pesquisador. Porém em um sábado qualquer, os servidores principais do site tiveram problemas em seus discos rígidos, o que resultou no site ficar fora do ar.

Por sorte, um dia antes, becapes haviam sido feitos no dia anterior por máquinas externas ao centro de processamento de dados do Chattes. Só que como os becapes nunca haviam sido testados, houve preocupação sobre o que podia ter sido perdido no processo. Foi descoberto que todas as informações estavam presentes, mas de forma totalmente desorganizada.

Foram encontradas todas as publicações de todos os pesquisadores da plataforma distribuídas em vários arquivos em formato de citação. Os autores são encontrados na primeira parte da citação que é delimitada por pontos, e são separados por ponto e vírgula (sempre com um espaço após o ponto e vírgula). Na segunda parte da citação há o nome do arquivo, e ao final, o ano de publicação do documento e um ponto sempre termina a citação.

Para remontar o site, porém, os programadores do Chattes precisam descobrir o número de publicações de cada autor da plataforma, e precisam da sua ajuda para fazer isso da forma mais eficiente possível, pois são mais de 1 milhão de usuários e uma quantidade mais absurda ainda de publicações registradas.

Input

A primeira linha contém um inteiro N ($0 \leq N \leq 10^4$), a quantidade de publicações encontradas. Não há publicações repetidas.

Seguem N linhas, cada uma com uma publicação. Uma publicação é composta pelo inteiro M ($8 \leq M \leq 10^5$) um espaço e a publicação em formato de citação de tamanho M. Em uma citação aparecem apenas caracteres ASCII imprimíveis.

A soma de todos os M não excede 10^5.

Output

A primeira linha deve conter o inteiro P, o número de pesquisadores.

Em seguida devem seguir P linhas, que devem conter o nome do pesquisador, um espaço e o número de publicações daquele pesquisador.

Example

entrada padrão
7 58 Dijkstra, Edgar. Go To Statement Considered Harmful. 1968. 73 Liskov, Barbara; Wing, Jeannette. A Behavioral Notion of Subtyping. 1994. 31 Tzu, Sun. The Art of War. -500. 75 Dijkstra, Edgar. On the Cruelty of Really Teaching Computing Science. 1988. 72 Knuth, Donald. On the translation of languages from left to right. 1965. 57 Knuth, Donald. Semantics of Context-Free Languages. 1968. 48 Grossman, Marvin. Canudos Nao Tem Buracos. 2021.
saída padrão
6 Dijkstra, Edgar 2 Grossman, Marvin 1 Knuth, Donald 2 Liskov, Barbara 1 Tzu, Sun 1 Wing, Jeannette 1

Problem C. Floresta em Perigo

Input file:	`entrada padrão`
Output file:	`saída padrão`
Time limit:	1 segundo
Memory limit:	256 megabytes

O Amazonia 1 é o primeiro satélite de observação da Terra completamente projetado, integrado, testado e operado pelo Brasil. Ele foi lançado em 28 de fevereiro de 2021, e tem o propósito a observação do território nacional. Ele, e outros satélites que o Instituto Nacional de Pesquisas Espaciais (INPE) opera fornecem informações vitais sobre o desmatamento nos biomas do Brasil. No ano de 2020 se atingiu um novo patamar: O maior índice de desmatamento já encontrado na Amazônia, um recorde no número de queimadas no Pantanal e um aumento inédito nos incêndios no Pampa.

Para fazer estas medidas, os satélites fazem fotografia no espectro infravermelho, que permite ver o fogo mesmo que ele esteja "invisível"ao espectro visível. Dado o alcance desses satélites, o começo de uma queimada pode ser composto de apenas um pixel diferente na imagem. O algoritmo é extremamente refinado para impedir que reflexões na água, variações na atmosfera, exposição do solo e outras medições incorretas não sejam identificadas como focos de queimadas.

Você foi encarregado de fazer um algoritmo que estima o estrago provocado por uma queimada. Usando as imagens do espectro visível e análise do clima, além dos identificados pontos de queimadas você também conta com uma estimativa da aridez de cada um dos pixels da imagem, que influencia no comportamento do espalhamento da queimada.

- Pixels com aridez 0 estão queimando.
- Pixels com aridez 1 estão completamente secos, mas não estão queimados. Geralmente é resultado da prática comum na Amazônia de corte da floresta, seguido de um período para secar a vegetação, e então o uso do fogo para limpar a área desmatada. Frequentemente o local é então utilizado para pasto.
- Pixels com aridez 2 estão parcialmente secos e com grande potencial para queimar. Geralmente é resultado da agricultura de coivara, que envolve em preparar o terreno de antemão para queimá-lo, e posteriormente usar o local para plantio, utilizando das cinzas para fertilização.
- Pixels com aridez 3 e 4 fazem parte da floresta intocada, com umidades diferentes devido ao clima e localização.
- Pixels com aridez 5 são de água ou de material não inflamável.

O espalhamento das chamas então é determinado pelo seguinte comportamento:

- Pixels 1 tendo 1 ou mais vizinhos 0 na sua 8-vizinhança viram 0 na próxima unidade de tempo.
- Pixels 2 tendo 1 ou mais vizinhos 0 na sua 4-vizinhança viram 0 na próxima unidade de tempo.
- Pixels 3 tendo 2 ou mais vizinhos 0 na sua 4-vizinhança viram 0 na próxima unidade de tempo.
- Pixels 4 tendo 3 ou mais vizinhos 0 na sua 4-vizinhança viram 0 na próxima unidade de tempo.

Determine então depois de T unidades de tempo o quanto o fogo se espalhou pelo mapa.

Input

A primeira linha contém três inteiros, as dimensões do mapa de aridez N e M ($1 \leq N, M \leq 10^3$) e a quantidade de unidades de tempo T ($0 \leq T \leq 10^6$).

Seguem então N linhas, cada uma com M inteiros, sendo cada inteiro correspondente a um nível de aridez conforme o enunciado.

Output

Determine quantos pixels estão queimando depois de T unidades de tempo.

Examples

entrada padrão	saída padrão
8 8 3 1 0 1 1 1 5 5 1 1 1 1 1 1 5 5 5 1 1 1 1 1 5 5 5 2 2 2 2 5 5 5 1 3 3 3 5 5 5 1 1 4 4 5 5 5 0 2 1 5 5 5 5 5 2 1 3 5 5 5 5 3 4 4 4	28
5 5 7 0 3 4 2 5 3 1 2 3 5 2 5 2 1 3 5 5 5 5 1 2 2 2 2 2	16

Problem D. Gatinhos Criptográficos

Input file:	`entrada padrão`
Output file:	`saída padrão`
Time limit:	0.5 segundos
Memory limit:	256 megabytes

Os NFTs (Non-fungible tokens) são *tokens* criptográficos que representam algo único. Diferente de uma cédula de dinheiro que pode ser trocada por outra do mesmo valor, um NFT não é fungível, e é equivalente a uma obra de arte em termos práticos: É único, insubstituível e tem valor inerente à sua unicidade. A implementação computacional de um NFT geralmente envolve o uso da *blockchain* que é um mecanismo que permite que exista um registro público de transações único e confiável de forma totalmente descentralizada.

O primeiro uso mais popular das NFTs pode ser atribuído aos *CryptoPunks*, um projeto de 2017 que criou 10 000 cartoons únicos na *blockchain* Ethereum. Naquele mesmo ano, inspirado por esse projeto foi criado outro projeto, o *CryptoKitties* que introduzia gatinhos únicos que podiam ser adotados e trocados no Ethereum. O sucesso dos gatinhos foi tão imprescindível que em dezembro de 2017, o jogo congestionou a rede Ethereum causando um enorme lentidão no seu sistema de transações. O mundo gastou na casa de milhões de dólares comprando esses gatinhos criptográficos que eternamente estarão gravados nos registros da *blockchain* Ethereum.

Os gatinhos do *CryptoKitties* tem um número único e um genoma de DNA de 256 bits que definem suas características. O jogo tem por volta de 12 "gatributos", que incluem cores, formato dos olhos, padrão, formato da boca, entre outros. O jogo conta atualmente com mais de 1,5 milhão de gatos únicos que podem ser cruzados para obter novas características de forma teoricamente infinita.

Mas você quer ir além. A população de gatos no mundo é estimada em 600 milhões, então seu objetivo é colocar esses gatos em existência no mundo virtual imediatamente. A sua proposta é distribuir gatos em intervalos, e permitir a compra a venda também em intervalos. Com o sistema de transações pronto, você agora quer saber quantos gatos cada dono possui dado o histórico de transações.

Input

A primeira linha contém um inteiro N ($1 \leq N \leq 3 \cdot 10^4$), o número de transações. Seguem N linhas, cada uma com uma transação. Elas podem ser dos seguintes tipos:

- Um inteiro 0, os inteiros A e B ($0 \leq A \leq B \leq 10^9$) e o identificador do dono D ($0 \leq D \leq 10^5$) representando o nascimento de um intervalo de gatos $[A, B]$ de dono D. É garantido que o intervalo $[A, B]$ não pertence a um dono.
- Um inteiro 1, um intervalo A e B ($0 \leq A \leq B \leq 10^9$) e dois identificadores de dono V e C ($0 \leq V, C \leq 10^5, V \neq C$) representando uma transferência dos gatos do intervalo $[A, B]$ do dono V para o dono C. É garantido que o dono V possui os gatos do intervalo.

Output

A saída deve consistir do inteiro M, a quantidade de donos de gatos com mais de 0 gatos.

Em seguida, seguem M descrições de donos de gato que devem ser ordenadas pelo identificador do dono. A primeira linha de cada descrição deve consistir do inteiro D, o identificador do dono e um inteiro I, a quantidade de intervalos de gatos que esse dono possui. Em seguida, seguem I linhas com a descrição dos intervalos ordenadas pelo início do intervalo e de forma que não exista uma descrição de intervalos menor.

Cada descrição de intervalo é composta por dois inteiros A e B que descrevem um intervalo de gatos $[A, B]$ que o dono D possui.

Example

entrada padrão	saída padrão
5 0 0 5 0 0 6 10 1 0 11 20 2 1 7 8 1 2 1 0 5 0 1	2 1 2 0 6 9 10 2 2 7 8 11 20

Problem E. Impressão nas Terças-Feiras

Input file:	`entrada padrão`
Output file:	`saída padrão`
Time limit:	1 segundo
Memory limit:	256 megabytes

As eleições na Nlogônia estão em chegando! E nessa nova eleição, será implantado um novíssimo sistema que visa garantir a segurança do voto, que envolve além do registro eletrônico do voto, a impressão de um comprovante que é verificado pelo eleitor e colocado e depois colocado em uma urna selada. Todo o sistema foi homologado, testado e estava a princípio pronto para as eleições. Porém, de uma hora para outra, a data da eleição teve que ser alterada. Um furacão de enormes proporções iria passar pela Nlogônia naquele dia, então foi necessário escolher outra data para ser realizada a eleição.

É claro que os engenheiros responsáveis pelo sistema de votação foram os últimos a serem avisados dessa mudança no cronograma. E sem ser consultados, eles viram que a nova data da eleição seria numa terça-feira. Oh não. O desespero foi compartilhado por todos que sabiam que o firmware das impressoras tinha um bug esquisito que impedia que se a data informada fosse terça-feira, nada imprimiria. Mas agora, eles tinham um problema gigantesco nas mãos, afinal não havia tempo de fazer modificações tanto no software da urna quanto no da impressora, pois envolveria mais uma rodada de homologação e testagem.

A solução dos engenheiros, é claro, é extremamente simples. Foi proposto a inclusão de um chip microcontrolador na placa da urna de forma que se na comunicação entre a urna e a impressora fosse detectado a data de impressão como terça-feira, o chip modificaria a saída para que fosse a próxima quarta-feira. Fazer funcionar para um dia específico seria fácil, mas como eles queriam usar essa solução nas próximas urnas, eles querem que funcionem para qualquer terça-feira dentro da vida útil da urna, e descobriram que fazer isso não era uma tarefa fácil.

Então eles estão pedindo a sua ajuda. Eles entregaram nas suas mãos o protocolo da impressora:

- Toda transmissão inicia com dois bytes 255. Isso serve para diferenciar qualquer ruído no cabo de uma mensagem legítima. Ao invés de transmitir ruído, transmita 0.
- Segue o ano (últimos dois dígitos), mês e dia da impressão em três bytes A ($21 \leq A \leq 29$), M ($1 \leq M \leq 12$) e D ($1 \leq D \leq$ dias no mês M).
- Então há a data, hora e segundo da impressão em 3 bytes h ($0 \leq h \leq 23$), m ($0 \leq m \leq 59$) e s ($0 \leq s \leq 59$).
- Em seguida seguem dois bytes que descrevem B ($1 \leq B \leq 10^4$), a quantidade de bytes a se imprimir em *big-endian* (bytes mais significativos vem primeiro).
- Então seguem B bytes referentes a impressão.
- Por fim, um byte C é uma soma de verificação que é obtida fazendo xor em todos os bytes da mensagem (excluindo o próprio C) para certificar a integridade dos dados referentes a impressão (os dados do cabeçalho da mensagem são sempre consistentes).

Seu trabalho então é pegar as mensagens vindas da rede e fazer a modificação pedida (modificando inclusive mensagens inválidas, mas certifique que ela continuará inválida na saída).

Input

Uma sequência de bytes X ($0 \leq X \leq 255$) separados por espaço em uma linha. É garantido que não há mensagens parciais e que a quantidade de bytes é inferior a 10^5.

Output

Uma sequência de bytes do mesmo tamanho da entrada e no mesmo formato com as modificações.

Mensagens inválidas na entrada devem continuar inválidas na saída, invalidando-se a soma de verificação. Qualquer soma de verificação inválida serve.

Examples

entrada padrão
4 0 255 0 255 255 21 8 31 13 54 54 0 11 72 101 108 108 111 32 87 111 114 108 100 36 23 255
saída padrão
0 0 0 0 255 255 21 9 1 13 54 54 0 11 72 101 108 108 111 32 87 111 114 108 100 59 0 0

entrada padrão
92 255 255 21 1 1 7 10 20 0 7 99 117 114 105 111 115 111 0
saída padrão
0 255 255 21 1 1 7 10 20 0 7 99 117 114 105 111 115 111 69

entrada padrão
255 255 23 2 28 16 20 3 0 1 65 78 0 1 255 255 24 12 31 11 51 23 0 1 66 82
saída padrão
255 255 23 3 1 16 20 3 0 1 65 82 0 0 255 255 25 1 1 11 51 23 0 1 66 69

Problem F. Jogo Ponto-a-Ponto

Input file:	`entrada padrão`
Output file:	`saída padrão`
Time limit:	2 segundos
Memory limit:	400 megabytes

Usando redes ponto a ponto, é possível compartilhar informações entre computadores na Internet de forma direta, sendo necessário um servidor apenas para fazer a conexão inicial entre os dois computadores. Buscando reduzir custos de servidor, a Entendo, uma empresa famosa de jogos, decidiu implementar o modo multijogador via rede ponto-a-ponto no seu mais novo jogo, Sr. Vídeo Construtor.

O modo multijogador do Sr. Vídeo Construtor tem um limite de três jogadores simultâneos, e dentro do jogo é possível bloquear usuários indesejados, que obrigatoriamente não podem participar das mesmas partidas. Dado essas complicações, a Entendo acabou fazendo um sistema que não garante que dado a fila de jogadores, as partidas que tenham as melhores conexões sejam selecionadas, acabando por prejudicar muito a experiência dos jogadores que reclamam do lag constante. Mas para o próximo jogo, o Sr. Vídeo Construtor 2, eles garantem que todos esses problemas serão resolvidos.

Porém, isso é só *marketing*, porque realmente eles não sabem como resolver. A dificuldade é em pegar uma fila de jogadores e descobrir quais são os possíveis K melhores trios para se criar uma partida. Eles definiram que um trio é melhor que outro se a distância entre os jogadores for menor do que no outro trio, ou em caso de empate, o melhor trio é aquele que tem o jogador de menor posição na fila que o outro trio não possui.

Para descobrir a distância entre os jogadores, no processo de entrada na fila, o jogo faz *pings* periódicos testando a conexão entre os jogadores. Para economizar banda, o jogo não faz testes de conexão com todos os outros jogadores da fila, mas como os testes formam uma rede de jogadores conectada, é possível extrapolar a distância entre quaisquer jogadores traçando um caminho entre os dois jogadores com a menor quantidade de pulos (e em caso de empate, a menor distância).

Usando essas informações, ajude os engenheiros da Entendo a melhorar o modo multijogador do seu novo jogo!

Input

A primeira linha contém quatro inteiros. O primeiro deles é N ($0 \leq N \leq 400$), a quantidade de jogadores esperando na fila. O segundo é M ($0 \leq M \leq \binom{N}{2}$), a quantidade de testes de conexão entre pares diferentes de jogadores. O terceiro é B ($0 \leq B \leq \binom{N}{2}$), a quantidade de pares diferentes de jogadores bloqueados. E por último, K ($1 \leq K \leq 10^5$), a quantidade de melhores trios requisitados.

Em seguida seguem M linhas, cada uma descrevendo um teste de conexão. Um teste de conexão é composto por dois inteiros U e V ($1 \leq U < V \leq N$), as posições dos jogadores na fila e um inteiro P ($1 \leq P \leq 10^4$), a distância estimada entre os dois jogadores em milisegundos.

Em seguida seguem B linhas, cada uma descrevendo um par de jogadores bloqueados que não podem jogar juntos. Cada descrição tem dois inteiros U e V ($1 \leq U < V \leq N$), representando a posição dos dois jogadores na fila.

Output

A primeira linha da saída deve ser T ($0 \leq T \leq K$), a quantidade de melhores trios encontrados. T deve ser o máximo possível.

Em seguida, seguem T linhas, cada uma contendo um trio dado por três inteiros U, V e W ($1 \leq U < V < W \leq N$), referentes a posição dos jogadores na fila, ordenados do melhor trio para o pior.

Example

entrada padrão	saída padrão
7 10 1 8 1 5 2 3 5 3 1 3 5 1 2 2 2 3 2 3 4 3 2 4 4 2 6 8 6 7 2 4 7 9 1 5	8 1 2 3 2 3 4 2 3 5 1 2 4 3 4 5 1 3 4 2 4 5 1 2 6

Problem G. Kandalarrai

Input file:	`entrada padrão`
Output file:	`saída padrão`
Time limit:	2 segundos
Memory limit:	256 megabytes

Vinícius tem uma história que aconteceu com ele mas que mais cedo ou mais tarde pode acontecer contigo. Ele foi convidado pra uma festa de BCC (Bacharelado em Complexidade Computacional), e estava tudo muito bacana. Mas para a surpresa dele, começaram a falar em língua estranha, gritavam O(Kandalarrai)! O(Kandalarrai)! Confuso, Vinícius suspeitava que se tratava de algum tipo de complexidade que ainda não viu no curso. Em seguida, começaram a cantar:

```
Crux Sacra Sihi mihi lux;
non draco sihi mihi dux;
vade retro satana;
nunquan suad mihi vana;
sunt mala quae libas;
ipse venena bibas.
```

Daí Vinícius ficou perdido de vez. Decidiu que ia deixar pra entender tudo aquilo só no dia seguinte.

No dia seguinte, Vinícius entendeu tudo. Ele abriu o seu computador, pesquisou "Kandalarrai" no PatoPatoVai e lá estava o artigo da Wikipédia para salvar a pátria:

"Na teoria da computabilidade, a **Função de Kandalarrai**, nomeada por Wilhelm Kandalarrai, é uma função computável comumente representada pela letra grega κ. A função $\kappa_n(i, j)$ é definida da seguinte maneira para os inteiros não negativos i, j e n:

$$\kappa_n(i,j) = \begin{cases} j+1 \pmod{n} & \text{se } i = 0 \\ \kappa_n(i-1,1) & \text{se } i > 0 \text{ e } j = 0 \\ \kappa_n(i-1,\kappa_n(i,j-1)) & \text{se } i > 0 \text{ e } j > 0 \end{cases}$$

Uma versão mais restrita da Função de Kandalarrai tem muitas aplicações na área da complexidade computacional, e envolve a retirada da restrição do módulo no seu caso base, e geralmente é denotada k_∞ ou apenas A. A função inversa denotada por k_∞^{-1} ou σ por exemplo, é presente em análises de complexidade de estruturas de união-busca."

Agora Vinícius entendeu tudo o que tinha acontecido. E aquele canto em latim? Era só a Oração de São Bento, o que no contexto de uma universidade católica, faz todo sentido. Mas Vinícius então ficou curioso para calcular a Função de Kandalarrai, você pode ajudá-lo?

Input

A primeira linha contém um inteiro T ($0 \leq T \leq 20$), a quantidade de casos de teste.

Seguem T linhas, cada uma com três inteiros I, J ($0 \leq I, J \leq 10^6$) e N ($0 \leq N \leq 10^5$).

Output

Imprima na mesma ordem da entrada o resultado de $\kappa_N(I, J)$ para cada caso de teste.

Examples

entrada padrão	saída padrão
12 0 0 13 1 0 13 1 1 13 3 8 13 4 1 13 12 12839 182 3 4 100 18 123 1412 3 4 200 8 9 10000 12 182 100000 14 8414 100000	1 2 3 4 5 13 25 725 125 8733 48733 48733
9 1493 9303 1969 2200 8210 1969 3130 7001 1969 4020 6023 1969 5221 5101 1969 6031 4032 1969 7231 3404 1969 8383 2400 1969 9803 1501 1969	1698 0 0 0 1698 0 0 0 1698