

# Índice de Arquivos

Arquivo de entrada: entrada padrão  
Arquivo de saída: saída padrão  
Tempo limite: 1 segundo  
Limite de memória: 256 megabytes

O sistema de arquivos Fila de Arquivos no Topo (FAT) possui uma lista ligada de arquivos no topo. Arquivos e as entradas de pasta são guardados em alguma localização do disco e para controle do sistema de arquivos, é criado um nó ao final da lista ligada no topo que representa aquele arquivo ou pasta.

Em cada nó dessa lista, é encontrado os índices do começo dos fragmentos do arquivo no disco, o tamanho desses fragmentos, opcionalmente alguns metadados, e o índice do próximo nó dessa lista ligada.

		Fragmento 1		Fragmento 2		Fragmento 3	
Tamanho	Próximo	Tamanho	Índice	Tamanho	Índice	Tamanho	Índice
8	11	4	1038	192	1937	42	2505
0	1	2	3	4	5	6	7

Fragmento 1							
Tamanho		Próximo	Tamanho	Índice	Metadados		
			5	16	4	1038	u=x
8	9	10	11	12	13	14	15

Este sistema de arquivos tem esse nome porque permite que possíveis becares do sistema sejam facilmente criados. Qualquer modificação no conteúdo dos arquivos ou nos seus metadados implica a deleção daquela entrada, e ela é inserida novamente no fim da fila. Os criadores do sistema de arquivos sabiam que os espaços deixados iam ser problemáticos, e implementaram um algoritmo simples que detectava se tinha espaços no arredor do final da fila e moviam algumas entradas pra trás.

Esse algoritmo funciona bem para o usuário comum que não cria e deleta uma quantidade excessivas de arquivos. Mas os usuários ferrenhos de computadores que se vangloriam com seus *uptimes* gloriosos, começaram a ver que isso é um problema sério, pois o espaço para guardar arquivos no disco ficava cada vez menor, afinal não se pode guardar arquivos no meio dessa fila de arquivos no topo.

No mercado, já existia uma ferramenta similar, o *Desfragmentator 3000*, que organiza a posição dos arquivos de forma a melhorar o desempenho da máquina, isto é, diminuir os espaços que estão entre os arquivos apontados pela fila de arquivos no topo e unificar entradas. Porém, isso não resolve o nosso problema, pois queremos otimizar os espaços encontrados nessa fila, ao invés dos arquivos apontados pela fila.

Então, como usuário ferrenho, você decidiu criar uma solução para desfragmentar a fila de arquivos no topo. O seu programa funcionou muito bem, mas ele é excessivamente lento, porque ele move todos as entradas da fila umas próximas das outras, e no seu sistema de arquivos com mais de 120 000 arquivos, isso demora um tempo considerável.

Então, você teve uma ideia: Limitar o programa para uma quantidade de tempo. O tempo para mover uma entrada da fila de arquivos é proporcional ao tamanho das entradas a serem movidas. Então, bole um algoritmo que libere a maior quantidade de espaço no disco (lembrando que o espaço no disco começa

no final da fila) dado um tempo para sua execução. Lembre-se, porém, de manter a propriedade temporal do sistema de arquivos.

## Entrada

Na primeira linha, se encontra um inteiro  $N$  ( $1 \leq N \leq 10^6$ ), a quantidade de arquivos ou pastas do sistema.

Em seguida, seguem  $N$  linhas, cada uma descrevendo uma entrada da lista ligada. As linhas contêm  $M$  ( $4 \leq M \leq 10^8$ ), o tamanho da entrada e  $P$  ( $0 \leq P \leq 10^{10}$ ), a posição do próximo item da lista no disco, sendo que o último aponta para o primeiro.

Os itens são dados em ordem de posição no disco que começa na posição 0, onde  $P$  é estritamente crescente e sempre aponta para a entrada na próxima linha (exceto pelo último). As inconsistências na lista já foram detectadas e arrumadas nesta etapa do processo, então é garantido que o sistema de arquivos é consistente.

Por fim, é dado em uma linha o tempo  $T$  ( $0 \leq T \leq 10^8$ ), que é o tempo disponível para mover os arquivos, medido em quantidade de posições de memória que podem ser movidas.

## Saída

Imprima um inteiro  $K$ , o máximo de espaço que é possível liberar em tempo  $T$ .

## Exemplos

entrada padrão	saída padrão
3 8 11 5 16 6 0 11	3
3 5 6 4 11 6 0 9	1