Problema

Balanceamento de carga é muito importante em ambientes Cloud. Estamos sempre tentando minimizar os custos para que possamos manter o número de servidores o menor possível. Em contrapartida a capacidade e performance aumenta quando adicionamos mais servidores.

Em nosso ambiente de simulação, em cada `tick` (unidade básica de tempo da simulação), os usuários conectam aos servidores disponíveis e executam uma tarefa. Cada tarefa leva um número de ticks para ser finalizada (o número de ticks de uma tarefa é representado por `ttask`), e após isso o usuário se desconecta automaticamente.

Os servidores são máquinas virtuais que se auto criam para acomodar novos usuários. Cada servidor custa R$ 1,00 por `tick` e suporta no máximo `umax` usuários simultaneamente. Você deve finalizar servidores que não estão sendo mais usados.

O desafio é fazer um programa em Python que recebe usuários e os aloca nos servidores tentando manter o menor custo possível.

Input

Um **arquivo** onde:

- a primeira linha possui o valor de `ttask`;
- a segunda linha possui o valor de `umax`;
- as demais linhas contém o número de novos usuários para cada `tick`.

Output

Um **arquivo** onde cada linha contém uma lista de servidores disponíveis no final de cada `tick`, representado pelo número de usuários em cada servidor separados por vírgula e, ao final, o custo total por utilização dos servidores

Limites

1 ≤ ttask ≤ 10
1 ≤ umax ≤ 10

Exemplo

input.txt

```
4
2
1
3
0
1
0
1
```

output.txt

```
1
2,2
2,2
2,2,1
1,2,1
2
2
1
1
0
15
```

Detalhamento do exemplo

- ttask = 4 (valor da primeira linha do input.txt)
- umax = 2 (valor da segundo linha do input.txt)

Tick	Input	Output	Explicação
1	1	1	1 servidor para 1 usuário. (1 servidor criado)
2	3	2,2	2 servidores para 4 usuários. (1 servidor criado)
3	0	2,2	2 servidores para 4 usuários. (nenhum servidor criado ou removido)
4	1	2,2,1	3 servidores para 5 usuários. (1 servidor criado)
5	0	1,2,1	3 servidores para 4 usuários. (nenhum servidor criado ou removido)
6	1	2	1 servidor para 2 usuários. (2 servidores removidos)
7		2	1 servidor para 2 usuários. (nenhum servidor criado ou removido)
8		1	1 servidor para 1 usuário. (nenhum servidor criado ou removido)
9		1	1 servidor para 1 usuário. (nenhum servidor criado ou removido)
10		0	nenhum servidor e nenhum usuário. (1 servidor removido)
		15	Custo Total: R$1 x 5 ticks (primeira VM) + R$1 x 4 ticks (segunda VM) + R$1 x 6 ticks (terceira VM) = R$15

Critérios de avaliação

- Funcionamento
- Testes de unidade
- Cobertura de testes
- Complexidade de código
- Padronização de código (PEP-8, PEP-257)