

## 1 Problema

Alfred deseja planejar o que cozinhar nos próximos dias. Ele pode cozinhar vários pratos. Para cada prato, o custo dos ingredientes e o lucro final é conhecido. Se um prato é cozinhado duas vezes seguidas, o valor do lucro na segunda vez é 50 por cento do lucro na primeira vez. Se ele é preparado uma terceira vez ou mais em seguida, o valor do lucro é zero. Por exemplo, se um prato, que gera um lucro  $v$ , é cozinhado três vezes em seguida, o lucro final desses três dias é  $1.5v$ . Ajude-o a construir o cardápio que maximiza o lucro sob a restrição de que seu orçamento não seja excedido.

## 2 Entrada

A entrada consiste em vários casos de teste. Cada caso de teste começa com 3 inteiros em uma linha: O número de dias  $k$  ( $1 \leq k \leq 21$ ) que Alfred quer planejar, o número de pratos  $n$  ( $1 \leq n \leq 50$ ) que ele pode cozinhar e seu orçamento  $m$  ( $0 \leq m \leq 100$ ). As  $n$  próximas linhas descrevem os pratos que Alfred pode cozinhar. A  $i$ -ésima linha contém dois inteiros: o custo  $c$  ( $1 \leq c \leq 50$ ) e o lucro  $v$  ( $1 \leq v \leq 10000$ ) do  $i$ -ésimo prato. O final da entrada é definido pelo caso de teste com  $k = n = m = 0$ . Não é necessário processar esse caso de teste. A entrada não deve ser lida de qualquer arquivo. Ela deve vir da entrada padrão.

## 3 Saída

Para cada saída, imprima o valor máximo do lucro alcançável, com 1 dígito após o ponto decimal. Imprima então  $k$  inteiros com o  $i$ -ésimo inteiro sendo o número do prato a ser cozinhado no dia  $i$ . Pratos são numerados de 1 a  $n$ . Imprima pelo menos um espaço ou caractere de nova linha após cada inteiro. Se existirem vários cardápios possíveis alcançando o lucro máximo, selecione aquele com menor custo. Se existirem dois ou mais com o mesmo custo mínimo, você pode imprimir qualquer um deles.

Se todo cardápio exceder o orçamento, imprima apenas o valor 0 como lucro (veja o caso um no exemplo). A saída não deve ser escrita em nenhum arquivo. Ela deve ser escrita na saída padrão.

## 4 Exemplo

Entrada:

```
2 1 5
3 5
3 5 20
2 5
18 6
```

1 1  
3 3  
2 3  
0 0 0

Saída:

0.0  
  
13.0  
1 5 1

## 5 Entrega

O aluno deve entregar três arquivos apenas. O primeiro arquivo contendo uma solução do problema utilizando uma **abordagem gulosa**; um outro arquivo com a solução utilizando **programação dinâmica**. Por fim um arquivo README com uma breve descrição da solução utilizada. A abordagem gulosa não precisa, necessariamente, dar a solução ótima.

### 5.1 Perguntas e Problemas para o README

O arquivo README deve possuir respostas para as seguintes perguntas:

1. Como esse problema pode ser modelado para o paradigma guloso?
2. Seu algoritmo guloso apresenta a solução ótima? Por quê?
3. Como esse problema pode ser modelado para o paradigma de programação dinâmica?
4. Discuta a subestrutura ótima e a sobreposição dos problemas.
5. Se algum algoritmo clássico foi adaptado para resolver o problema, qual foi ele?

## 6 Dicas!

Teste seu programa (opcional) <http://www.spoj.pl/problems/MENU/>