Estrutura de Dados e Algoritmos
Projeto 02: João e os Games
Ciência da Computação

Prof. Daniel Saad Nogueira Nunes

1 Contextualização

João gosta muito de comprar jogos de videogame, tanto é que gasta toda a sua mesada com isso. Para "zerar ou platinar" um jogo, ele demora exatamente 2 semanas. Como ele recebe a sua mesada religiosamente, ele quer comprar dois jogos por mês , assim, ele demorará um mês para *zerar* os dois jogos que comprar.

Além disto, João quer gastar toda a sua mesada com jogos, e nada mais. Então ele deve tentar achar dois jogos cuja soma dos preços seja exatamente o valor da sua mesada. No caso de múltiplos jogos que satisfaçam esse critério, ele sempre opta por escolher aqueles com menor diferença de preço.

Como a biblioteca de jogos existentes é muito grande, João pediu a sua ajuda para ajudá-lo a encontrar esses jogos.

2 Especificação

O projeto deverá ser executado através da linguagem `C`.

A entrada deve ser lida da entrada padrão (`stdin`), enquanto a saída deverá ser impressa na saída padrão (`stdout`).

O programa deverá obedecer rigorosamente o formato de saída especificada, pois parte da correção será automatizada.

2.1 Entrada

A primeira linha da entrada possui dois inteiros separados por um espaço: N ($2 \leq n \leq 10^5$), indicando o número de jogos disponíveis na biblioteca que joão nunca *zerou* e M ($1 \leq n \leq 2 \cdot 10^9$), indicando o valor da mesada de João.

As próximas N linhas descrevem cada um dos jogos. A descrição de um jogo é composta por: seu nome (até 30 caracteres maiúsculos de A a Z) e seu preço, que está no intervalo inteiro $[1, 10^9]$. O nome do jogo é separado de seu preço por um espaço em branco.

É garantido que o nome dos jogos são distintos.

2.2 Saída

Seu programa deverá fornecer como saída uma linha com o nome de dois jogos, separados por um espaço, cuja soma seja igual ao valor da mesada de João. Os Em caso de múltiplas soluções, deverá ser impressa qualquer uma que minimize a diferença entre os jogos.

Caso não existam jogos que satisfaçam o critério de João, seu programa deverá imprimir uma linha com a mensagem "`impossivel`".

3 Exemplos

- Exemplo :

```
5 10
CYBERPUNKLXXII 10
DOOM 2
TETRIS 6
THEWITCHER 8
PACMAN 4
```

- Saída:

```
PACMAN TETRIS
```

- Entrada:

```
2 5
TNMTIV 2
SUPERMARIO 2
```

- Saída:

```
impossivel
```

- Entrada:

```
5 20
FINALFANTASYVII 10
HALO 10
PES 10
DOTA 10
METALGEAR 10
```

- Saída:

```
FINALFANTASYVII HALO
```

3.1 Limites de Tempo e Memória

Para cada caso de teste, será permitido a execução do programa por apenas 1 segundo com utilização máxima de 256 MB de memória. Caso o programa leve mais tempo ou memória do que isso, considerar-se-á que o algoritmo empregado foi ineficiente.

3.2 Documentação

Junto do(s) código(s) necessário(s) para resolver o problema, deverá ser disponibilizado um arquivo README, identificando o autor do trabalho e especificando as instruções para compilação e execução do(s) código(s).

3.3 Critérios de Correção

Fazem partes dos critérios de correção:

- Eficiência do programa.
- Utilização de estruturas de dados adequadas.
- Documentação: além do arquivo README, o código deve estar bem documentado.
- Legibilidade.

3.4 Ambiente de Correção

Os projetos serão corrigidos em uma máquina com sistema `GNU/Linux` e compilador `gcc 10.2.0`.

Trabalhos que não compilarem não serão avaliados.

4 Considerações

- Este projeto deve ser executado individualmente.
- Os trabalhos que incidirem plágio serão avaliados automaticamente com nota 0 para os envolvidos. Medidas disciplinares também serão tomadas.
- O trabalho deve ser entregue dentro de uma pasta zipada com a devida identificação do(s) aluno(s) através da sala de aula virtual da disciplina na data estipulada no ambiente.