**Descrição do Problema e da Solução**

Pequena descrição da solução proposta e mapeamento com o problema (1 ou 2 parágrafos).

Para ambos os problemas para descobrir o tamanho da maior subsequência possível, a ideia é ter um vetor (lensList) em que na mesma posição de um certo elemento teria o tamanho máximo de uma subsequência crescente que acaba nesse elemento, para o problema 1 teremos uma lista adicional para quantas subsequências crescentes existem com o tamanho dito na lensList.

(É expressamente proibido utilizar fontes externas de código !!)

**Análise Teórica**

Análise teórica da complexidade total e das várias etapas da solução proposta.

* Leitura dos dados de entrada: simples leitura do input, com ciclo(s) a depender linearmente O(n)
* Processamento da instância para fazer alguma coisa. Logo, O(1)
* Aplicação do algoritmo 1 para encontrar a numero e o tamanho das maiores subsequências crescentes numa lista com n elementos. Logo, O(n^2)
* Aplicação do algoritmo 2 para encontrar tamanho da maior subsequência comum crescente entre uma lista com n elementos e outra com m elementos. Logo, O(n\*m)
* Calculo do numero de listas com o maior tamanho possivel. O(n^2)
* Apresentação dos dados. O(1)

Complexidade da solução do 1º algoritmo: O(n^2)

Complexidade da solução do 2º algoritmo: O(n\*m)

**Avaliação Experimental dos Resultados**

Descrição do tipo experiências feitas e gráfico demonstrativo da avaliação de tempos associados.

Gerar pelo menos 10 instâncias (e indicar quais) de tamanho incremental e cálculo dos tempos para cada instância.

Gerar o gráfico do tempo (eixo do YYs) em função do tamanho da instância de entrada (eixo dos XXs) como exemplificado abaixo. Indicar a informação dos eixos.



Concluir se o gráfico gerado está concordante com a análise teórica prevista.