**Descrição do Problema e da Solução**

Para este primeiro projeto da cadeira de ASA foram-nos colocados dois problemas. O primeiro prendia-se com o cálculo do tamanho da maior subsequência estritamente crescente bem como o número de subsequências estritamente crescentes de tamanho máximo, dada sequência de inteiros, x. O segundo problema relacionava-se com o cálculo, dadas duas sequências de inteiros, x e y, do tamanho da maior subsequência comum estritamente crescente entre x e y.   
Para o primeiro problema baseámo-nos no ***patience sorting***. Este algoritmo utiliza uma estrutura de pilhas de cartas, usualmente aplicado para ordenação normal de números, sendo que aqui adaptámo-lo para o fim pretendido - o número de pilhas formadas corresponde ao tamanho da subsequência crescente de tamanho máximo. Dessa forma, o objetivo passará por criar pilhas com os elementos da sequência fornecida tal que: vamos verificar qual é a pilha mais à esquerda onde podemos colocar o elemento, ou seja, a primeira pilha em que o elemento do topo é maior que o elemento que estamos a considerar inserir. Caso não existam pilhas ou caso não exista nenhuma pilha onde podemos colocar o elemento, criamos uma nova pilha. Para obter a quantidade de subsequências crescentes de tamanho máximo, tivemos de recorrer à noção de **somas cumulativas**. Sempre que adicionamos uma carta a uma pilha, realizamos a diferença entre a soma guardada no topo da pilha à sua esquerda e o primeiro elemento dessa mesma pilha que é maior que ele próprio, adicionando esse valor ao topo da pilha onde será inserido. A soma guardada no topo da pilha mais à direita corresponderá, então, à quantidade de subsequências crescentes de tamanho máximo.  
Para o segundo problema, tivemos por base o algoritmo de construção de matrizes para encontrar o tamanho da maior subsequência comum dadas duas sequências. Contudo, para otimizarmos o desempenho, consideramos apenas um vetor, lenghts, com tamanho igual ao número de elementos da segunda sequência. Para calcularmos o tamanho da maior subsequência comum fazemos o seguinte: para cada elemento i da primeira sequência iremos percorrer toda a segunda sequência, elemento a elemento. Designando por j o elemento da segunda sequência realizamos uma de duas operações: caso os elementos que estamos a considerar sejam iguais, podemos ou não aumentar o tamanho da maior subsequência comum, caso o elemento i da primeira sequência seja maior que o elemento j da segunda sequência, como estamos apenas a considerar as subsequências que terminam no elemento de índice j da segunda sequência, então a que acaba nesse elemento também irá acabar no elemento que estamos a considerar pelo que atualizamos o tamanho da maior subsequência. No fim, depois de vistos todos os elementos iremos procurar o maior elemento no vetor lenghts que corresponde ao tamanho da maior subsequência comum.

**Análise Teórica**

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

Inserir aqui um pseudo código de muito alto nível a indicar a complexidade de cada etapa.

Exemplo:

* Leitura dos dados de entrada: simples leitura do input, com ciclo(s) a depender de linearmente/quadraticamente/… de V/E/V+E/… Logo, Θ(V)
* Processamento da instância para fazer alguma coisa. Logo, O(??)
* Aplicação do algoritmo X para fazer algo. Logo, O(?X?X)
* Transformação dos dados com uma dada finalidade. O(?Y?Y?)
* Apresentação dos dados. O(???)

Complexidade global da solução: O(!??!)

**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.