Lista de Exerc´ıcios de PAA - M´odulo 3 - 2021.1

Prof. Vinicius Fernandes dos Santos

1. Considere um vetor de *n ≥* 2 posi¸c˜oes, cada uma preenchida com um d´ıgito no intervalo 0-9. Queremos decompor este vetor em subvetores, ou seja, em trechos consecutivos. Al´em disso, todo subvetor deve ser par, isto ´e, o n´umero formado pelos d´ıgitos do subvetor deve ser par. Por exemplo, caso o vetor de entrada seja 805438452, este pode ser particionado como “8054, 38, 452”, com 3 subvetores, ou como “8, 0, 54384, 52”, com 4 subvetores, dentre outras possibilidades. Escreva um algoritmo que determine o maior n´umero de subvetores em que o vetor pode ser decomposto. Determine a complexidade de seu algoritmo. Vocˆe pode assumir que o ´ultimo d´ıgito do vetor ´e par.

2. Dizemos que uma sequˆencia ´e alternante se n˜ao existe trˆes elementos consecutivos *ai*, *ai*+1, *ai*+2 tal que *ai < ai*+1 *< ai*+2 ou que *ai > ai*+1 *> ai*+2. Em outras palavras, para todo *i*, os sinais de *ai − ai*+1 e *ai*+1 *− ai*+2 s˜ao distintos. Dado um vetor de n´umero inteiros, vocˆe deve determinar o tamanho da maior subsequˆencia alternante. Para o vetor 4*,* 3*,* 2*,* 5*,* 8*,* 7*,* 1*,* 10*,* 11, uma poss´ıvel subsequˆencia alternante seria 4*,* 2*,* 8*,* 7*,* 11. Forne¸ca um algoritmo para resolver este problema e determine sua complexidade. Vocˆe pode assumir que todos os elementos do vetor de entrada s˜ao distintos.

3. Esta quest˜ao novamente trata de sequˆencias alternantes. Desta vez, diferentemente da quest˜ao anterior, n˜ao estamos interessados em uma subsequˆencia, mas sim em um subvetor. Ou seja, queremos um conjunto de posi¸c˜oes consecutivas que seja alternante. Para o mesmo vetor de exemplo da quest˜ao anterior, 3*,* 2*,* 5 e 7*,* 1*,* 10 seriam subvetores alternantes. Estamos interessados em determinar o tamanho do maior subvetor alternante. Note que quaisquer dois elementos distintos consecutivos formam um subvetor alternante. Forne¸ca um algoritmo para resolver este problema e determine sua complexidade. Vocˆe pode assumir que todos os elementos do vetor de entrada s˜ao distintos.

4. Dada uma string bin´aria (ou seja, uma string de zeros e uns), descreva uma estrat´egia para determinar o tamanho da maior sequˆencia de zeros consecutivos desta string. Vocˆe deve utilizar um dos paradig mas visto durante o curso para resolver este problema. Descreva qual paradigma foi utilizado e explique porque, de fato, seu algoritmo pode ser interpretado como um exemplo de uso de tal paradigma.

5. Considere um algoritmo de programa¸c˜ao dinˆamica que resolva o problema da mochila. Considere que h´a *n* elementos, a mochila tem capacidade *C* e dois vetores s˜ao fornecidos, com *pi* e *vi* correspondendo ao peso e ao valor do *i*-´esimo elemento, respectivamente.

Considere que o resultado do algoritmo foi armazenado em uma tabela, de forma que pd[i][j] armazena o maior lucro que pode ser obtido podendo-se utilizar quaisquer elementos do conjunto *{*1*, . . . , i}* em uma mochila de capacidade *j*.

Mostre como reconstruir a lista de objetos que deve ser selecionada em uma solu¸c˜ao ´otima, fornecendo um algoritmo para isso. Para isso, escolha uma das seguintes abordagens.

(a) Descrever a solu¸c˜ao apenas a partir da tabela pd[][], supondo que ela j´a foi computada anterior mente. Ou seja, todas as decis˜oes a serem tomadas se baseiam nos ´ındices e nos valores armazenados na tabela. Se escolhida (e resolvida corretamente) esta abordagem pode lhe conferir pontos extras.

(b) Descrever a solu¸c˜ao completa, fornecendo o algoritmo para computar os valores de pd[][] e quaisquer outras estruturas auxiliares que julgar necess´arias.

6. Um grande esquema de corrup¸c˜ao assola a Federa¸c˜ao dos Planetas Unidos. Os ´org˜aos competentes inicia ram uma investiga¸c˜ao, para entender como a quadrilha opera. Disfar¸cados de pessoas bem intencionadas, esta quadrilha ´e estruturada de forma extremamente hier´arquica, com um ´unico l´ıder. Com exce¸c˜ao do l´ıder, cada membro da quadrilha possui um chefe imediato. Desta forma, a cadeia de comando ´e muito bem definida.

As autoridades resolveram tomar uma providˆencia, ap´os o surgimento de algumas provas contra a quadri lha. Entretanto, as provas n˜ao s˜ao muito contundentes: o ideal seria a obten¸c˜ao de “dela¸c˜oes premiadas” para validarem as suspeitas e para possibilitar a obten¸c˜ao de provas inquestion´aveis. Como o “prˆemio” de tais dela¸c˜oes consiste de redu¸c˜ao de pena, de certa forma, quanto mais delatores, menos criminosos ser˜ao punidos. Desta forma, deseja-se que o n´umero de dela¸c˜oes seja o menor poss´ıvel. Por outro lado, para condenar todos os criminosos, para cada membro da quadrilha ´e necess´ario que ele ou pelo menos um de seus associados (ou seu chefe imediato, ou seus subordinados imediatos) fa¸ca uma dela¸c˜ao.

Forne¸ca um algoritmo para determinar o menor n´umero de dela¸c˜oes necess´arias para condenar toda a quadrilha. Vocˆe pode assumir que os membros da quadrilha estejam numerados de 1 a *n*, onde o n´umero 1 corresponde ao l´ıder, e que existe um vetor chefe, onde o valor chefe[i] corresponde ao chefe imediato de *i*. Al´em disso, vocˆe pode assumir que chefe[1] = 0. Analise a complexidade de seu algoritmo.

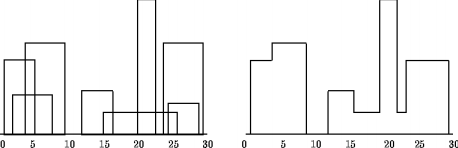


Figura 1: Ilustra¸c˜ao do exemplo de entrada acima. Note, no lado direito, o *skyline* formado pelos retˆangulos fornecidos na entrada. Figura original do UVa Online Judge (http://uva.onlinejudge.org).

7. O problema do *skyline* consiste em determinar, a partir de uma lista de pr´edios, qual o visual formado quando olhamos de longe. Neste problema, vamos assumir que uma pessoa observa uma cidade de um ponto suficientemente distante, de forma que os pr´edios s˜ao vistos como retˆangulos em um plano. Tais pr´edios ser˜ao representados por triplas (*ai, bi, ci*), correspondendo a um pr´edio de altura *bi*, se es tendendo da coordenada *ai* at´e a coordenada *ci*, no eixo *x*. A entrada do problema ´e uma sequˆencia (*a*1*, b*1*, c*1)*, . . . ,*(*an, bn, cn*). A sa´ıda deve ser uma sequˆencia de n´umeros inteiros (*x*1*, y*1*, . . . , xn, yn, xn*+1), de forma que a altura do *skyline* entre os pontos *xi* e *xi*+1 ´e *yi*.

Nesta quest˜ao vocˆe deve fornecer um algoritmo de tempo *O*(*n* log *n*) para este problema. Vocˆe pode assumir que as triplas j´a est˜ao armazenadas em alguma estrutura, ou que s˜ao passada como parˆametro. N˜ao ´e necess´aria sua leitura.

Dica: use divis˜ao e conquista.

Um exemplo de entrada seria o conjunto de triplas (1,11,5), (2,6,7), (3,13,9), (12,7,16), (14,3,25), (19,18,22), (23,13,29), (24,4,28). Para esta entrada, a sa´ıda esperada seria (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0). Note que para uma entrdaa com um ´unico pr´edio, a sa´ıda corresponde exatamente `a entrada.

8. A Nlogˆonia ´e uma pr´ospera na¸c˜ao, que preza pela eficiˆencia de seus processos. Para aumentar a eficiˆencia de seu sistema monet´ario, seu sistema de c´edulas est´a sendo modificado: agora *n* tipos de c´edulas ser˜ao utilizados, de valores *c*1*, . . . , cn*. Para garantir que todos os valores inteiros positivos podem ser formados, foi definido que *c*1 = 1. Tal mudan¸ca implicar´a na reprograma¸c˜ao dos caixas eletrˆonicos do pa´ıs que, prezando pela eficiˆencia, sempre fornece o menor n´umero de c´edulas poss´ıvel quando um nlogoniano saca uma quantia *Q*. Alguns testes preliminares est˜ao sendo feitos com determinadas sequˆencias *c*1*, . . . , cn* e valores *Q*. Nesta quest˜ao, vamos ajudar os nlogonianos com os testes para decidir o sistema ideal a ser utilizado.

(a) Forne¸ca um exemplo de sequˆencia *c*1*, . . . , cn* e um valor *Q* tal que um algoritmo guloso que sempre tente utilizar o maior valor de c´edula poss´ıvel n˜ao produz a menor quantidade de c´edulas poss´ıvel. Explique qual seria a solu¸c˜ao encontrada pelo algoritmo e qual seria a solu¸c˜ao ´otima.

(b) Forne¸ca uma rela¸c˜ao de recorrˆecia para determinar o valor *m*(*i, j*) correspondendo ao menor n´umero de c´edulas para se fazer a quantia *j*, podendo-se utilizar os *i* primeiros tipos de c´edulas. N˜ao se esque¸ca dos casos base!

(c) Escreva um algoritmo correspondente que determina o valor de *m*(*n, Q*). Para que tal algoritmo seja eficiente, recomenda-se utilizar programa¸c˜ao dinˆamica. Analise sua complexidade de tempo. Para praticar, vocˆe pode fazer duas vers˜oes: uma top-down e outra bottom-up.

9. Dada uma sequˆencia de n´umeros inteiros positivos *s* = *s*1*, . . . , sn*, escreva um algoritmo de tempo linear que retorne o menor inteiro positivo *k* que n˜ao esteja em *s*. Pode-se interpretar o seu algoritmo como aplica¸c˜ao de algum dos paradigmas visto em sala? Justifique.