Derde Serie Opgaven

1. Stel je krijgt drie rijtjes letters: $X = x_1, x_2, \ldots, x_m$, $Y = y_1 y_2, \ldots, y_n$ en $Z = z_1, z_2, \ldots, z_{m+n}$. Z is een *shuffle* van X en Y als Z gemaakt kan worden door om beurten een stukje van X en een stukje van Y te nemen, waarbij de letters in volgorde blijven staan. Bijvoorbeeld, *cchocohilaptes* is een shuffle van *chocolate* em en *chips*, maar *cchocolhilaptes* is dat niet.

   (a) Ontwerp een dynamic programming algoritme die X, Y, Z als input neemt en beslist of Z wel of niet een shuffle van X en Y is.

   (b) Wat is de tijdcomplexiteit van de algoritme?

   (c) Is *pnesuumonoulpertracmicralifroagilscopspicsilticicovoexplcaianocolidoniocioussis* een shuffle van *pneumonoultramicroscopicsilicovolcanoconiosis* en *supercalifragilisticexpialidocious*?

2. Gegeven zijn een stel dozen met dimensies: breedte $\times$ diepte $\times$ hoogte ($b_i \times d_i \times h_i$), voor $1 \leq i \leq n$. Een doos i mag bovenop een doos j staan als $b_i \leq b_j$ *en* $d_i \leq d_j$. Dozen mogen niet worden gedraaid om ze passend te krijgen. Beschrijf een dynamic programming algoritme die een zo hoog mogelijke stapel maakt van een deelverzameling van deze dozen. Wat is de complexiteit van uw algoritme? Hoe hoog is de stapel die uw algoritme kan maken van de set: $(7,7,7); (10,4,7); (4,4,1); (2,2,2); (3,4,5); (6,8,5); (6,1,9); (8,8,4); (1,1,9); (10,4,6); (6,2,5); (2,8,1); (5,1,3); (8,2,6); (1,8,9); (10,2,4); (2,1,1); (9,6,8); (1,4,6); (7,2,8)$?

3. Gegeven is een balk van lengte N die in stukken gezaagd moet worden. De zaagmaatschappij rekent €L om een balk van lengte L op een willekeurige plaats te zagen. Dus als een balk van lengte 12 op plaatsen 2, 5 en 8 gezaagd moet worden kan dat €12+10+7 kosten, of €12+5+7 afhankelijk van de volgorde (2,5,8, of 5,2,8). Een greedy algoritme om de zaagkosten te minimaliseren zou bijvoorbeeld kunnen zijn de balk altijd zo dicht mogelijk bij het midden te zagen.

   (a) Laat zien dat die greedy algoritme niet werkt (geef een voorbeeld met 3 sneden).

   (b) Geef een dynamic programming algoritme die wel werkt (kijk naar matrixvermenigvuldiging)

   (c) Wat kost het om een balk van lengte 100 te zagen op de plaatsen:
   $10, 18, 20, 28, 32, 33, 42, 52, 62, 63, 66, 74, 75, 80, 84, 86, 94, 97$?

4. Het is makkelijk te zien dat DFS en BFS dezelfde worst-case tijdcomplexiteit hebben, maar met geheugengebruik ligt dat anders.

   (a) Geef een graaf en een knoop v, zodat de stapel bij een DFS $\Omega(n)$ knopen bevat, terwijl de stapel bij BFS, gestart in dezelfde knoop v,altijd beperkt is tot 1 knoop.

   (b) Geef een graaf waarbij het omgekeerde het geval is, dus de BFS onderhoudt een queue met $\Omega(n)$ knopen, terwijl de recursiediepte bij DFS constant is.