Hertentamen Algoritmiek
Maandag 10 juli 2017, 14.00 – 17.00 uur

Als er om uitleg, toelichting of motivatie gevraagd wordt bij een opgave, is het belangrijk om die ook te geven.
Als je het antwoord op een onderdeel niet weet, en je hebt dat antwoord nodig bij een later onderdeel, dan kun je het antwoord 'kopen' bij de docent.
Globale puntenverdeling: 1: 27 pt; 2: 24 pt; 3: 21 pt; 4: 28 pt. **Veel succes!**

---

1. Gegeven een array $A = A[0], A[1], \ldots, A[n-1]$ dat $n(\geq 1)$ reële getallen bevat. We willen weten wat het maximum getal en wat het minimum getal in A is. We gaan dit probleem op drie manieren oplossen: met brute force, met decrease-and-conquer en met divide-and-conquer. In alle drie gevallen moet het gevonden minimum worden geretourneerd in de parameter `min` en het gevonden maximum in de parameter `max`.

   (a) Geef een eenvoudig (brute force) iteratief algoritme dat het minimum getal en het maximum getal in array A bepaalt. Schrijf hiervoor een C++-functie `void bepaalminmax1 (double A[ ], int n, double &min, double &max)`.

   (b) Geef een voorbeeld van een slechtste geval voor $n = 8$, voor je algoritme uit het vorige onderdeel. Dat wil zeggen: een rij getallen $A[0], A[1], \ldots, A[7]$ waarvoor het algoritme zoveel mogelijk vergelijkingen tussen reële getallen uitvoert.

   Hoeveel vergelijkingen tussen reële getallen worden er voor dit voorbeeld uitgevoerd? Motiveer dit laatste antwoord.

   (c) Geef nu een decrease-by-one algoritme voor bovenstaand probleem. Schrijf daartoe een *recursieve* C++-functie `void bepaalminmax2 (double A[ ], int i, double &min, double &max)`, die het probleem oplost voor het (deel)array $A[0], A[1], ..., A[i-1]$ ($i \geq 1$). De aanroep `bepaalminmax2 (A, n, min, max);` geeft dan uiteraard het antwoord voor het hele array.

   (d) Neem aan dat n een 2-macht is en minstens 2. Geef nu een divide-and-conquer algoritme voor het probleem. Het array dient hiervoor in twee even grote delen te worden verdeeld. Schrijf een *recursieve* C++-functie `void bepaalminmax3 (double A[ ], int links, int rechts, double &min, double &max)` die het probleem oplost voor het deelarray A[links],. . . ,A[rechts] ter lengte een 2-macht (ook nu: minstens 2). De aanroep `bepaalminmax3 (A, 0, n-1, min, max);` geeft dan uiteraard het antwoord voor het hele array.

   (e) Geef een voorbeeld van een slechtste geval voor $n = 8$, voor je algoritme uit het vorige onderdeel. Dat wil zeggen: een rij getallen $A[0], A[1], \ldots, A[7]$ waarvoor het algoritme zoveel mogelijk vergelijkingen tussen reële getallen uitvoert.

   Hoeveel vergelijkingen tussen reële getallen worden er voor dit voorbeeld uitgevoerd? Motiveer dit laatste antwoord.

---

2. Bij het afscheid van zijn vrienden heeft Rick een puzzelboekje gekregen waar hij helemaal gek van wordt. Het lijkt zo simpel: elke puzzel bestaat uit twee woorden, die elk bestaan uit karakters 0 en/of 1. Het eerste woord P (van patroon) is M karakters lang. Het tweede woord T (van tekst) is N karakters lang. Er geldt dat $N \geq M \geq 1$ en de vraag is hoe vaak het woord P voorkomt als 'gecombineerd deelwoord' van T. Dat wil zeggen: op hoeveel manieren kun je precies het woord P krijgen door M letters te kiezen uit T en die in de oorspronkelijke volgorde achter elkaar te zetten. Bijvoorbeeld: het woord $P = 01$ komt vier keer voor in $T = 01101$:

   $\underline{0}\ \underline{1}\ 1\ 0\ 1$
   $\underline{0}\ 1\ \underline{1}\ 0\ 1$
   $\underline{0}\ 1\ 1\ 0\ \underline{1}$
   $0\ 1\ 1\ \underline{0}\ \underline{1}$

   (a) Geef alle voorkomens van $P = 0110$ in $T = 01001101$. Gebruik daarbij een notatie als in het voorbeeld hierboven, of iets vergelijkbaars.

   (b) Voor de puzzels in zijn puzzelboekje komt Rick zelf geen enkele keer uit op het antwoord dat op de laatste pagina van het boekje staat. Hij twijfelt derhalve of de antwoorden wel kloppen. Kun jij hem helpen?

   Schrijf een *recursieve* C++-functie `void zoekpatroon (char P[], int M, char T[], int N, int i, int j, int &aantal)` die met behulp van backtracking alle voorkomens van P in T opbouwt. Het patroon staat op posities 0 t/m $M-1$ van het char-array P, de tekst op posities 0 t/m $N-1$ van het char-array T. De indices i en j geven aan dat we op zoek gaan naar het karakter op positie i van P, te beginnen op positie j van tekst T. Steeds als we $P[i]$ vinden op een bepaalde positie in de tekst, dan doen we een recursieve aanroep.

   De eerste aanroep zal van de vorm `zoekpatroon (P, M, T, N, 0, 0, aantal);` zijn, waarbij `aantal` is geïnitialiseerd op 0. Iedere keer als we heel P gevonden hebben, wordt de parameter `aantal` met 1 opgehoogd.

   (c) Beschrijf voor algemene M en algemene N met $N \geq M \geq 1$ een 'zo slecht mogelijk geval' voor dit probleem: een puzzel bestaande uit een patroon P van lengte M en een tekst T van lengte N, waarvoor P **zo vaak mogelijk** voorkomt in T.

   Hoe vaak komt P voor in T in dit geval? Wat zegt dit dus over de worst-case tijdcomplexiteit van de functie `zoekpatroon`? Motiveer je antwoord op deze laatste vraag.

   (d) Geef twee verschillen tussen brute force algoritmen en backtracking algoritmen (in het algemeen). De verschillen hoeven niet per se van toepassing te zijn op het puzzelprobleem van Rick.

---

3. Tijdens zijn vakantie in het zonnige zuiden heeft Terence de keuze uit een groot aantal leuke activiteiten i, die allemaal een zeker tijdsinterval $[b_i, e_i)$ duren. De activiteiten overlappen deels, dus Terence kan helaas niet alle activiteiten doen. Wel wil hij graag zoveel mogelijk activiteiten (gedurende hun volledige interval) doen. Een typisch geval van een activiteiten-selectie-probleem. Kun jij Terence helpen?

   Tijdens het college hebben we vier mogelijke gretige algoritmes besproken om dit probleem aan te pakken:

   (a1) selecteer in iedere stap de activiteit die overlapt met zo min mogelijk andere activiteiten

   (a2) selecteer in iedere stap de activiteit die het eerst begint

   (a3) selecteer in iedere stap de activiteit die het eerst eindigt

   (a4) selecteer in iedere stap de activiteit die het kortst duurt

   In alle gevallen worden natuurlijk alleen activiteiten geselecteerd die compatibel zijn met eerder geselecteerde activiteiten. Wellicht ten overvloede: als activiteit i bijvoorbeeld eindigt op tijdstip $e_i = 8$, dan is een activiteit j die op tijdstip $b_j = 8$ (of elk later tijdstip) begint daarmee compatibel.

   (a) Slechts één van deze vier algoritmes geeft Terence daadwerkelijk in alle gevallen een optimale oplossing. Welke van de vier algoritmes is dit?

   (b) Pas het gretige algoritme van het vorige onderdeel toe op het volgende voorbeeld met $n = 11$ activiteiten:

i	b_i	e_i
1	3	8
2	6	10
3	7	11
4	4	8
5	8	11
6	2	6

i	b_i	e_i
7	2	13
8	12	14
9	1	4
10	5	7
11	3	5

   Geef bij iedere stap van het algoritme aan welke activiteit je bekijkt, en waarom je die activiteit wel of niet aan de oplossing toevoegt. Wat is de resulterende optimale oplossing?

   (c) De andere drie gretige algoritmes leveren Terence dus niet altijd een optimale oplossing op. Geef voor elk van deze algoritmes een voorbeeld (met plaatjes of getallen, maar in ieder geval duidelijk!) waarvoor dat algoritme niet (per se) een optimale oplossing oplevert.

   Geef bij elk voorbeeld ook aan,

   - welke oplossing het betreffende algoritme op zou (kunnen) leveren,
   - in welke volgorde daarbij activiteiten aan de oplossing zouden worden toegevoegd,
   - en wat een betere oplossing zou zijn.

   N.B.: bij één algoritme kan het best wel lastig zijn om zo'n voorbeeld te construeren. Als dat niet snel lukt, ga dan eerst verder met de rest van het tentamen.

---

4. Ronald bespreekt met zijn zaakwaarnemer de mogelijkheden voor contracten in de toekomst. Hij heeft een viertal aanbiedingen, van clubs die hem graag voor een aantal jaren als coach willen vastleggen. Voor zo'n periode (alle jaren bij elkaar) biedt elke club hem dan een bepaald totaal salaris. Het maakt de clubs niet uit of Ronald nu al bij hen komt, of pas over een aantal jaar: de aanbiedingen blijven gewoon staan. Als hij komt, moet hij de afgesproken periode echter wel volmaken. Ronald wil nog maximaal 15 jaar bij clubs werkzaam zijn. Daarna gaat hij gratis aan de slag bij het Nederlands elftal. Hij geeft zijn zaakwaarnemer de opdracht om van de beschikbare aanbiedingen die aanbiedingen te kiezen die hem het hoogste totaalsalaris opleveren. De zaakwaarnemer komt daar echter niet uit. Kun jij hem helpen?

   Een typisch geval van een knapzakprobleem. De aanbiedingen zijn de objecten, het aantal jaar van een aanbieding i is het gewicht w_i en het te verdienen salaris bij die aanbieding is de waarde v_i van het object. In dit geval is het maximale gewicht (capaciteit) W van de knapzak 15, want zoveel jaar wil Ronald maximaal nog bij clubs werken. De vier objecten/aanbiedingen zien er nu als volgt uit:

object i	w_i	v_i	v_i/w_i
1	5	45	9
2	4	32	8
3	8	56	7
4	6	36	6

   (a) Leg uit hoe best-first branch-and-bound werkt voor maximalisatieproblemen in het algemeen. Geef daarbij o.a. aan hoe (deel)oplossingen gegenereerd worden, wat met branch bedoeld wordt en wat met bound, wat best-first betekent, wanneer gesnoeid wordt, enz.

   (b) Tijdens het college hebben we een branch-and-bound algoritme voor het knapzakprobleem behandeld. Hierbij wordt gebruikt dat de objecten gesorteerd zijn op de gemiddelde-waarde-per-gewicht v_i/w_i, van groot naar klein. Iedere stap van het algoritme wordt een object wel of juist niet aan de knapzak toegevoegd, te beginnen bij object 1.

   Wat is de hierbij gebruikte bovengrens bij een deeloplossing? Geef zowel de bovengrens bij aanvang (als we nog geen object behandeld hebben), als de bovengrens na stap i (als we net object i wel of niet gekozen hebben). In dit laatste geval nemen we aan dat we nog niet alle objecten behandeld hebben. Geef antwoorden voor een algemeen knapzakprobleem met $n \geq 1$ objecten, dus niet specifiek voor het voorbeeld van de vier aanbiedingen voor Ronald.

   (c) Pas het branch-and-bound algoritme met de bovengrens uit het vorige onderdeel toe op het voorbeeld en teken de bijbehorende state-space-tree, met bij elke knoop (deeloplossing) de relevante informatie. Geef ook aan in welke volgorde de knopen zijn aangemaakt, welke knopen gesnoeid worden en waarom.

   Wat is dus de optimale oplossing voor Ronald?
