1e deeltentamen Algoritmiek 2016 / 2017

Je hebt drie uur voor dit deeltentamen.

Schrijf op elk van de ingeleverde bladen je naam en studentnummer, en op het eerste blad het aantal ingeleverde bladen.

Geef duidelijke antwoorden in helder en correct Nederlands of Engels. Wanneer niet expliciet anders gevraagd, mag je resultaten uit het college gebruiken.

Als een algoritme wordt gevraagd is het een goed idee om dit algoritme zowel te geven in pseudocode, als een toelichting in woorden te geven.

Schrijf netjes, en lever overzichtelijk werk in.

Deel je tijd goed in: sommige opgaven kunnen moeilijker zijn dan andere.

1. String matching (1 punt)

Leg in eigen woorden duidelijk uit hoe het Rabin-Karp algoritme voor string matching werkt.

(Geef de essentie duidelijk weer. Pseudocode volstaat hier niet: je moet in woorden e.e.a. uitleggen. Probeer niet meer dan 15 regels te gebruiken.)

2. Master theorem (1.5 punt)

Los de volgende betrekking op met behulp van de master theorem:

$$T(n) = 6 \cdot T(n/4) + O(n \log n)$$

Je antwoord moet zowel bestaan uit de oplossing van de betrekking, als een duidelijke uitleg hoe je tot dit antwoord gekomen bent.

3. Algoritme ontwerp: de bergtop (1.5 = 1.25+0.25 punten)

Gegeven is een array met n integers $A[1 \cdots n]$, waarbij we weten dat de rij getallen een 'bergvorm' heeft: tot een bepaald punt stijgt de rij, en na dat punt daalt de rij. Dat wil zeggen: er is een i met $1 < i < n$ zodat voor alle j met $1 \leq j < i$: $A[j] < A[j+1]$, en voor alle k met $i \leq k < n$: $A[k] > A[k+1]$. Een voorbeeld van een rij getallen met de bergvorm is:

$$3, 5, 7, 8, 6, 5, 3, 1$$

(i) **Geef een zo efficient (in O-notatie) mogelijk algoritme dat het grootste getal in A vindt.** *(Geef een duidelijke beschrijving van je algoritme.)*

(ii) **Hoeveel tijd gebruikt je algoritme (in O-notatie?)** Leg dit heel kort uit.

4. Het Pieterpad: Greedy algoritmen en Dynamisch Programmeren I. 3 = 0.25 + 1.25 + 1.25 + 0.25 punten

Twee vrienden, Legolas en Aragorn, willen het Pieterpad lopen (van Noord naar Zuid.) Ze beginnen natuurlijk bij 0 kilometer. De totale afstand van het Pieterpad is a_n kilometer.

Op verschillende punten op de route kan er overnacht worden, stel op punten p_0, p_1, p_2, $\ldots$, p_{n-1}, p_n. p_0 is het begin van de route en p_n het eind. Vanwege de treinreis ervoor en erna zullen Legolas en Aragorn altijd overnachten op p_0 en op p_n. We nemen aan dat de route eerst langs punt p_1 komt, dan langs p_2, enz.

Punt p_i komt na a_i kilometer op de route; we hebben dus $0 < a_1 < a_2 < \cdots < a_{n-1} < a_n$; de afstand van p_i naar p_{i+1} is $a_{i+1} - a_i$ kilometer.

Per dag kunnen de vrienden maximaal D kilometer afleggen. Ze willen nu beslissen op welke punten ze overnachten; tussen twee opeenvolgende overnachtingspunten mag niet meer dan D kilometer zitten.

We weten verder dat er nooit meer dan D afstand is tussen twee opvolgende herbergen, d.w.z., voor alle i, $1 \leq i < n$ hebben we $p_{i+1} - p_i \leq D$.

(i) Legolas wil het Pieterpad in zo min mogelijk dagen afleggen. Hij stelt het volgende greedy algoritme voor: elke dag wordt gelopen naar het verst mogelijke overnachtingspunt dat bereikt kan worden van de plek waar die dag begonnen wordt.

Dus, als we overnachten na a_i kilometer, dan is de volgende overnachting op a_j kilometer, voor die waarde van j waarvoor geldt dat $a_j \leq a_i + D$ en $a_{j+1} > a_i + D$, of $j = n$.

Geeft het algoritme van Legolas het correcte antwoord, d.w.z., wordt de route op deze manier in zo min mogelijk dagen afgelegd? Schrijf op: JA of NEE.

(ii) Zo ja, **geef een *duidelijk* bewijs dat dit greedy algoritme inderdaad optimaal is (zo min mogelijk dagen geeft)**. Zo nee, leg duidelijk uit waarom niet.

(iii) Aragorn merkt op dat het greedy algoritme ook dagen heeft waarbij de afstand die die dag gelopen is erg kort is. Hij wil iets anders doen. De rest van deze opgave gaat over dynamisch programmeren. Aragorn wil een dagindeling van het Pieterpad

- waarbij elke dag minstens C kilometer gelopen wordt, en
- waarbij elke dag hooguit D kilometer gelopen wordt.

Aragorn houdt erg van wandelen, en hij vraagt zich af wat het *grootste* aantal dagen is van een dagindeling die aan deze eisen voldoet.

Dit probleem kan met dynamisch programmeren worden opgelost. In deze opgave gaan we alleen kijken naar de recurrente betrekking.

Schrijf, voor $0 \leq i \leq n$, $Z(i)$ als het maximum aantal dagen voor een correcte verdeling van het gedeelte van het Pieterpad vanaf p_0 tot en met p_i, waarbij een overnachting plaatsvindt op p_i; als er niet zo'n verdeling bestaat dan is $Z(i) = -\infty$.

Geef een recurrente betrekking voor $Z(i)$. Leg je antwoord kort uit.

Bij deze opgave wordt niet gevraagd om het bijbehorende DP algoritme te maken; een goede recurrente betrekking volstaat.)

(iv) Geef naast de recurrente betrekking bij deel (iii) hierboven ook de **basisgevallen met bijbehorende waardes of formules**.

5. Aantallen tellen: Dynamisch Programmeren II (1.5 punten)

Een backpacker moet beslissen welke voorwerpen hij mee neemt. Er zijn n types voorwerpen $v_1, \ldots, v_n$. Van voorwerp v_i heeft hij a_i exemplaren ($1 \leq i \leq n$). Voorwerp v_i weegt g_i ($1 \leq i \leq n$). De handelsreiziger kan een totaal gewicht van maximaal G meenemen. Hij wil bepalen hoeveel verschillende mogelijkheden hij heeft wat betreft hij mee kan nemen.

Dit gaan we oplossen met dynamisch programmeren. Zij voor $j \in \{0, 1, \ldots, n\}$, en $B \in \{0, 1, \ldots, G\}$: $X(j, B)$ is het aantal mogelijkheden om voorwerpen $v_1, \ldots, v_j$ mee te nemen zodat het totaalgewicht precies B is. Natuurlijk, met inachtneming dat we een voorwerp niet meer keer mee kunnen nemen dan het aantal exemplaren.

De volgende recurrente betrekking geldt. (Dit mag je als gegeven beschouwen; je hoeft het niet te bewijzen.)

- $X(0, 0) = 1$.
- Als $B > 0$, dan is $X(0, B) = 0$.
- Als $1 \leq j \leq n$, dan is

$$X(j, B) = \sum_{i=0}^{\min\{a_j, \lfloor B/g_j \rfloor\}} X(j-1, B - i * a_j)$$

(i) (1.25 pnt) **Geef pseudocode van een dynamisch programmeer algoritme** dat berekent hoeveel mogelijkheden er zijn om voorwerpen $v_1, \ldots, v_n$ met een totaal gewicht van hooguit G, waarbij we elk voorwerp v_i hooguit a_i keer kunnen meenemen. Uw algoritme kan gebaseerd zijn op bovenstaande recurrente betrekking, waarvan je de correctheid mag aannemen.

(ii) (0.25 pnt) **Hoeveel tijd gebruikt je algoritme in O-notatie?** Leg dit heel kort uit.

6. Algoritme ontwerp II. 1.5 punten

We hebben een verzameling van n getallen A, opgeslagen in een array $A[1 \cdots n]$. Merk op dat A niet gesorteerd is. Verder zijn gegeven een integer i, $1 \leq i \leq n$, en een integer B.

We willen een algoritme voor de vraag: heeft A een deelverzameling van precies i getallen, zodat de som van die getallen hooguit B is?

Een voorbeeld: $A = [21, 4, 3, 6, 100, 7]$, $i = 2$ en $B = 10$. Het antwoord is nu *ja*: we kunnen bijvoorbeeld $\{3, 6\}$ nemen. We weten dat alle getallen in A verschillend zijn.

(i) **Geef een zo efficient (in O-notatie) mogelijk algoritme voor dit probleem.**

(ii) **Leg kort uit waarom je algoritme correct is.**

(iii) **Hoeveel tijd gebruikt je algoritme in O-notatie?** (*Je hoeft je tijdgrens niet verder toe te lichten.*)