Tentamen Algoritmiek
Dinsdag 31 mei 2011, 10.00 – 13.00 uur

Geef een **duidelijke toelichting** bij al je antwoorden.

Puntenverdeling: 1: 20; 2: 20; 3: 20; 4: 25; 5: 15

Opgave 1. We bekijken het volgende tweepersoonsspel. Het wordt gespeeld met $m * n$ damstenen, die in een m bij n rechthoek (m rijen en n kolommen) worden gelegd. Er zijn twee spelers, V (Verticaal) en H (Horizontaal), die om de beurt een zet doen. We spreken af dat V begint.
Als speler V een zet doet neemt deze steeds een hele kolom weg, speler H een hele rij. Door het wegnemen van een rij of kolom uit een rechthoek zal in het algemeen die rechthoek in twee kleinere rechthoeken worden verdeeld. Indien van de rand wordt weggehaald, resteert één kleinere rechthoek. Bij aanvang van het spel is er slechts één rechthoek, maar tijdens het spel worden dit er meer. Als een speler aan de beurt is neemt deze een hele kolom (V) of rij (H) weg uit precies één van die rechthoeken. De speler die de laatste steen/stenen weghaalt heeft gewonnen.

Een mogelijk spelverloop, met $m = 3$ en $n = 4$ (waarin de X-en de stenen aangeven en de verschillende rechthoeken door komma's gescheiden naast elkaar staan):

```
X X X X    V     X  X X    H     X              V     X           H    X
X X X X   --->   X, X X   --->   X, X X, X X   --->   X, X, X X  --->  X, X
X X X X          X  X X          X                    X                X
  (*)                                                                  (**)
```

In toestand (**) is V aan de beurt. Als deze in de volgende zet de kolom van 3 stuks weghaalt verliest hij. Veronderstel dus dat hij de losse steen weghaalt. Dan blijft er een toestand over die winnend is voor H. (Zij neemt namelijk de middelste rij (één steen) en haalt in haar volgende zet de laatste steen weg.) Stand (**) is derhalve verliezend voor V.

a. Wat zijn voor dit spel toestanden en acties (voor algemene m en n)?
b. Teken de toestand-actie-ruimte voor het geval $m = 2$ en $n = 4$, uitgaande van de beginsituatie waarin V begint. Geef bij elke toestand aan of deze winnend is voor V of voor H. Bepaal zo of het spel winnend is voor V of H en geef aan hoe diegene kan winnen. Toestanden die in één zet te winnen zijn hoef je niet verder uit te werken. Verder hoef je toestanden die speltechnisch hetzelfde zijn maar één keer uit te werken. Geef wel aan waarom zo'n stand dan winnend is voor V/H.

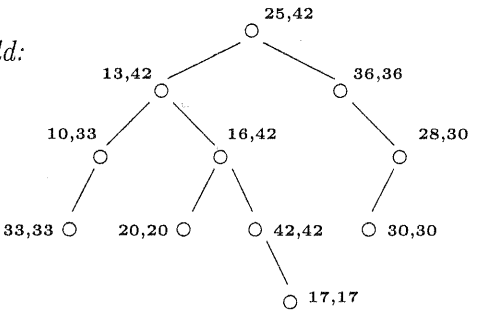
c. We bekijken het speciale geval waarbij $m = 1$, dus standen bestaande uit één enkele rij met n stenen. We veronderstellen weer dat V begint.
Toon aan: voor n oneven is zo'n stand winnend voor V. Geef hierbij een winnende strategie voor V. *Hint*: laat zien dat het geval 1 bij 3 winnend is voor V, en breid dit uit naar algemene oneven n.

Bonus (5 punten). Bewijs dat als n even is zo'n stand juist verliezend is voor V (dus winnend voor H). Geef een winnende strategie voor H.

Opgave 2. Gegeven een binaire boom met ingang `wortel`. Een knoop van de boom ziet er uit zoals hieronder links is aangegeven. De `info`-velden bevatten verschillende gehele getallen. De `reus`-velden zijn bij aanvang van deze opgave nog niet geïnitialiseerd.

```
struct knoop {
   knoop* links;
   knoop* rechts;
   int info;
   int reus;
}; // knoop
```

Voorbeeld:



a. Schrijf een *recursieve* C++-functie `void initialiseer(knoop* wortel)`, die van alle knopen in de boom met ingang `wortel` het `reus`-veld vult met de `info`-waarde uit die knoop. (Dus: kopieer de `info`-velden in de `reus`-velden.)
b. Neem aan dat de `reus`-velden gevuld zijn zoals in **a.** bedoeld. Schrijf nu een *recursieve* C++-functie `void reusachtig(knoop* wortel)`, die in elke knoop het `reus`-veld vult met het maximum van de `info`-waarden uit de (sub)boom met de betreffende knoop als wortel. In bovenstaand voorbeeld wordt bij elke knoop de inhoud van het `info`-veld en het aldus gevulde `reus`-veld gegeven.
c. We veronderstellen dat de `reus`-velden gevuld zijn zoals in **b.** bedoeld. Schrijf nu een *niet-recursieve* C++-functie `knoop* grootste(knoop* wortel)`, die een pointer oplevert naar de knoop die de grootste waarde uit de boom met ingang `wortel` bevat. Gebruik hierbij de `reus`-velden. Je mag aannemen dat de boom niet leeg is.

Opgave 3. Gegeven een array A dat n (≥ 2) gehele getallen $A[0], A[1], \ldots, A[n-1]$ bevat. Verder is gegeven dat op de even posities positieve getallen (> 0) staan, en op de oneven posities negatieve (< 0) getallen. Gevraagd wordt het aantal paren (i, j) met $i < j$ waarvoor $A[i] + A[j] = 0$.

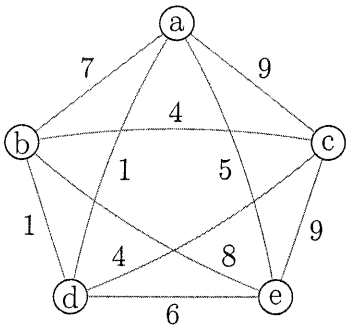
a. Van welke paren (i, j) weet je al zeker dat $A[i] + A[j] \neq 0$?

Om nodeloze vergelijkingen te vermijden mogen de bij **a.** genoemde paren bij **b.** en **c.** niet bekeken worden.

b. Geef een eenvoudig (brute force) algoritme in C++, dat het gevraagde aantal oplevert. Wat is de complexiteit van je algoritme (als functie van n)?
c. Geef een divide-and-conquer algoritme in C++ voor bovenstaand probleem. Verdeel hiertoe het array in twee gelijke delen. Neem aan dat n (≥ 2) een 2-macht is. Hier moet dus een recursieve C++-functie `int aantal2(A,links,rechts)` worden geschreven die het probleem oplost voor het deelarray A[`links`], ..., A[`rechts`] ter lengte een 2-macht.

Opgave 4. Tussen Leiden en Den Haag zal een sponsorloop gehouden worden die langs precies n locaties zal gaan. Je kunt vanuit elke locatie rechtstreeks in elke andere locatie komen. Elke deelnemer moet elke locatie precies één keer bezoeken en weer terugkeren in de startlocatie. Het stukje weg tussen locatie i en locatie j wordt voor een zeker bedrag (afhankelijk van i en j) gesponsord. Het doel is uiteraard om een route te vinden die elke locatie één keer aandoet, weer terugkeert in de startlocatie, en een zo hoog mogelijk sponsorbedrag oplevert. De mogelijke wegen en bijbehorende sponsorbedragen kunnen gemodelleerd worden met een complete, ongerichte graaf met n knopen en met gewichten (de sponsorbedragen) op de takken. We zoeken dan een Hamiltonkring met maximaal totaalgewicht.

a. Doe een voorstel voor een *gretig* algoritme voor dit probleem en pas het toe op onderstaand voorbeeld. Levert dit een Hamiltonkring met maximaal totaalgewicht op?



De kring abdec is een Hamiltonkring met totaalgewicht $7+1+6+9+9 = 32$. Dit is *niet* maximaal.

b. Leg uit hoe best-fit-first *branch and bound* werkt voor maximalisatieproblemen in het algemeen.
c. Beschrijf nu een best-fit-first branch and bound algoritme dat ons probleem oplost. Pas het algoritme toe op het voorbeeld en teken de bijbehorende state space tree. Geef daarin ook aan in welke volgorde de knopen bekeken worden. Wat voor afschatting van het te verwachten totaalgewicht gebruik je voor deeloplossingen?

Opgave 5. **a.** Wanneer noemen we een binaire boom een heap (hoopstructuur)?
b. Teken de met de rij 30 42 66 78 33 60 22 20 54 74 45 27 86 37 corresponderende complete binaire boom en breng deze m.b.v. *heapify* in hoopstructuur. Laat tussenstappen zien.
c. Leg uit hoe het sorteeralgoritme heapsort werkt. Voer ter illustratie de eerste twee herhaalstappen van heapsort uit op de hoopstructuur die in **b.** verkregen is.

Veel succes !