Examen Datastructuren en Algoritmen II

Naam : .....

Lees de hele oefening zorgvuldig voordat je begint ze op te lossen!
Als je niet goed verstaat wat de vraag of taak is, vraag het aan de lesgever!
Voor oefeningen die fout verstaan zijn, kunnen geen punten gegeven worden!

Schrijf leesbaar. Oplossingen die niet leesbaar zijn, kunnen ook niet beoordeeld worden.

Als technieken toegepast moeten worden, toon altijd voldoende tussenstappen om te kunnen zien wat er gebeurt en dat de technieken goed verstaan zijn.

Stellingen uit de les mogen natuurlijk altijd gebruikt worden zonder dat het bewijs opnieuw gegeven moet worden (behalve in gevallen waar het expliciet anders staat)!

1. **Verzamelingen 4 pt**

   - Het universum is de verzameling $\{1,\ldots,7\}$. Gebruik een union-find datastructuur met union by size en path compression. In gevallen waar niet uniek bepaald is wat de nieuwe wortel is, neem het element met de kleinere index.

     Pas de bewerkingen `union(1,2)`, `union(3,4)`, `union(5,3)`, `union(1,3)`, `union(2,4)`, `union(6,7)`, en `union(6,5)` in deze volgorde toe.

- Beschrijf een bewerking voor verzamelingen die in veel toepassingen gebruikt wordt maar waarvoor bitvectoren niet zo ideaal geschikt zijn als b.v. voor de berekening van de vereniging.

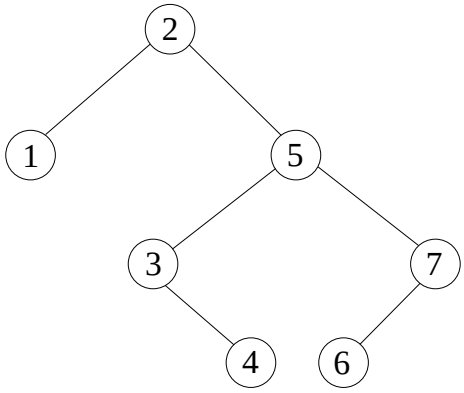
- Ons universum is de verzameling $\{0,\ldots,n\}$ en wij werken met een computer waar de lengte van een woord 64 bit is. Je mag (zoals in C) veronderstellen dat alle 64 bit voor de voorstelling van de verzamelingen bruikbaar zijn. M en M' zijn twee verzamelingen die als bitvectoren voorgesteld zijn (het zijn dus arrays $M[]$ en $M'[]$). Beschrijf (zoals in de les) zo efficiënt mogelijke uitdrukkingen die het volgende betekenen:
  - $i \in M$ en $i \in M'$
  - i is een element van precies één van de verzamelingen M, M'
  - er bestaat een element dat in beide verzamelingen M, M' zit
  - niet elk element $0 \leq j < 64$ komt in M of M' voor.

2. **Zoekbomen 4 pt**

- Geef een voordeel van semi-splay bomen in vergelijking met rood-zwart bomen en een voordeel van rood-zwart bomen in vergelijking met semi-splay bomen. Geef elke keer ook een toepassing waar dit voordeel goed te pas komt.

- Voeg de sleutels 7, 2, 5, 1, 4, 3 en 6 in deze volgorde toe aan een semi-splay boom.

- Verwijder sleutel 5 uit de volgende semi-splay boom.



- Geef een reeks van n bewerkingen op een semi-splay boom waarvan ten minste één bewerking kost $\Theta(n)$ heeft. Geef uitleg.

3. **Geamortiseerde complexiteitsanalyse 2 pt**

   Je hebt een binaire teller met n bits die $m \geq n$ keer geïncrementeerd wordt. Maar anders dan in de les staat de teller in het begin niet noodzakelijk op 0. Toon aan dat de geamortiseerde complexiteit per bewerking nog altijd constant is.

   Je kan zelf kiezen welke methode je gebruikt om dat aan te tonen, maar volgens mij gaat het met de potentiaalmethode het gemakkelijkst.

4. **Dynamisch programmeren 2.5 pt**

   Je kan in deze oefening kiezen of je een gemakkelijkere versie van de oefening wil oplossen. Deze versie is gekenmerkt als (a) – maar een goede oplossing van deze gemakkelijkere versie zal je ten hoogste 1.5 punten opleveren. Toch is een goede oplossing van (a) beter dan een slechte van de moeilijkere versie (b)...

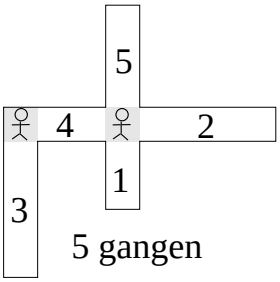
   Je moet een bedrag van n Eurocent aan iemand geven. De bedoeling is hem dit bedrag met zo weinig mogelijk munten te geven. De munten die je ter beschikking hebt zijn 1, 20 en 50 Eurocent.

   (a) Geef een dynamisch programmeren algoritme in pseudocode dat het minimale aantal munten bepaald.

   (b) Je hebt maar b_1 munten van 1 Cent, b_{20} munten van 20 Cent en b_{50} munten van 50 cent. Geef een dynamisch programmeren algoritme in pseudocode dat het minimaal **mogelijke** aantal munten bepaald – rekening houdend met het aantal munten van elke soort dat je hebt.

5. **Online algoritmen 1.5 pt**

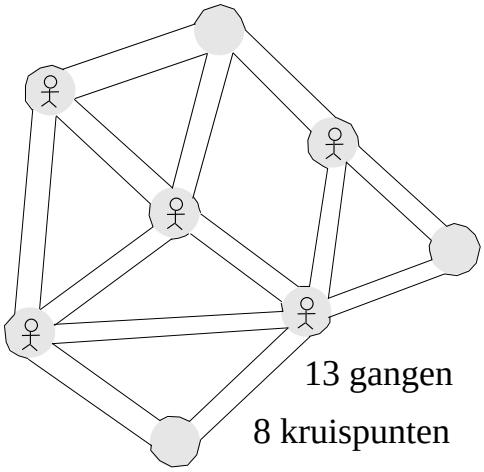
In een museum zijn er verschillende gangen die bewaakt moeten worden. De gangen zijn allemaal recht zodat iemand die op het einde van een gang staat de hele gang kan zien. Het eindpunt van een gang kan tegelijk het eindpunt van andere gangen zijn – dat noemen wij dan een kruispunt.



5 gangen

2 kruispunten

4 eindpunten die geen kruispunten zijn



13 gangen

8 kruispunten

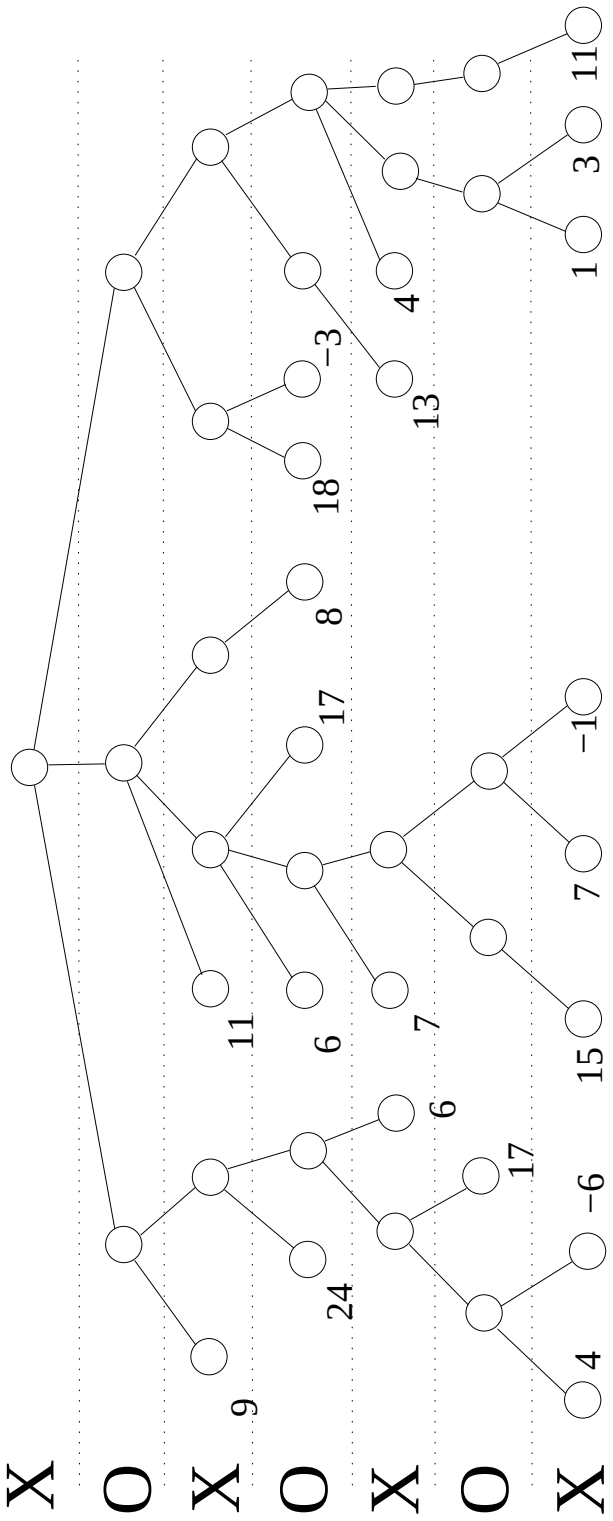
De taak is nu een manier te vinden om bewakers op de eindpunten of kruispunten van gangen te plaatsen zodat elke gang door ten minste één bewaker gezien kan worden – **maar:** je moet dit probleem als online probleem oplossen: je kent niet alle gangen en kruispunten op voorhand, maar je krijgt de gangen één bij één meegedeeld. Je moet elke keer dat je een gang krijgt meegedeeld veilig stellen dat die bewaakt wordt. Bewakers die geplaatst zijn mogen niet verplaatst worden. Maar **jij** moet geen algoritme geven. Jouw taak is de volgende:

Toon aan dat er voor elk online algoritme een inputreeks (dus een plan van een muzeum en een volgorde van gangen) zijn zodat dit algoritme 2 keer zoveel bewakers gebruikt als nodig.

Tip: Ook al verschilt dit probleem sterk van het online-inpak probleem uit de les is het basisidee van het bewijs toch bruikbaar. In feite is de toepassing van dit idee in dit geval zelfs duidelijk gemakkelijker. Je kan zelfs aantonen dat een dergelijke invoer voor elk optimaal aantal *b* van bewakers bestaat. Dit mag (natuurlijk) aangetoond worden maar moet niet.

6. **α–β–snoeien 1 pt**

   Bereken de waarde van dit spel. Evalueer altijd eerst de linkertak van een top en schrijf de grenzen aan de top. Pas (diep) α–β–snoeien toe. Maak duidelijk welke stappen je doet en beslissingen je neemt.



7. **Heaps 1 pt**

- Geef de (complete) definitie van een binomiale prioriteitswachtlijn.

- Voeg de volgende sleutels in deze volgorde toe aan een skew heap: 4, 2, 3, 5, 1 en 6

NOG NIET OMDRAAIEN !