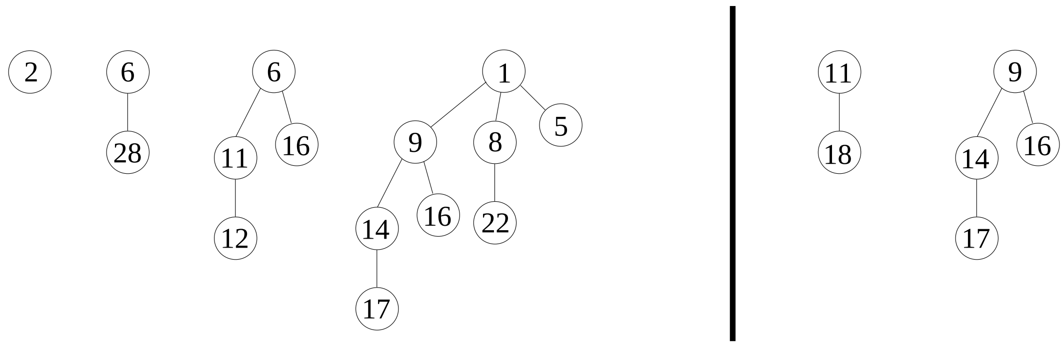
Examen Datastructuren en Algoritmen II

Naam : ....

Lees elke oefening zorgvuldig en helemaal voordat je met de oefening begint!

1. **(2.5 pt)**

   - Merge de volgende twee binomiale prioriteitswachtlijnen:

   Verwijder tenslotte het kleinste element. Toon voldoende tussenstappen om te zien wat er gebeurt.

- Voor welke bewerkingen is de bovengrens voor de tijd beter als je een binomiale prioriteitswachtlijn gebruikt in plaats van een leftist heap? Wat zijn de bovengrensen? (De $O()$-notatie is voldoende.)

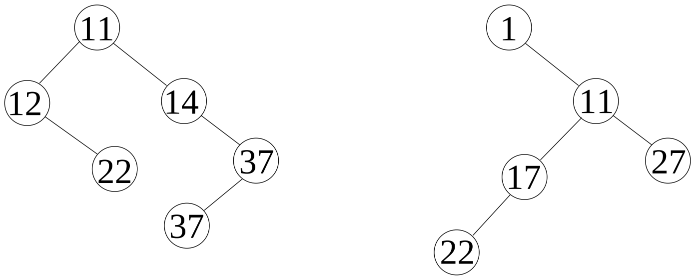
  Voor welke bewerkingen is de bovengrens voor de tijd beter als je een leftist heap gebruikt in plaats van een binomiale prioriteitswachtlijn? Wat zijn de bovengrensen? (De $O()$-notatie is voldoende.)

- Voor binomiale prioriteitswachtlijnen hebben wij expliciet toegelaten dat dezelfde sleutel er meerdere keren mag inzitten. Geef uitleg waarom dat een belangrijke eis is.

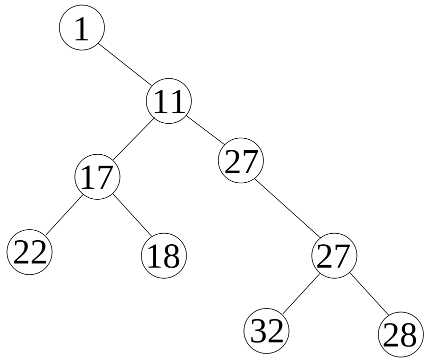
- Bewijs expliciet dat de wortel van een binomiale boom met hoogte k graad k heeft.

2. **(2.5 pt)**

- Merge de volgende twee heaps op een skew manier. Toon voldoende tussenstappen om te kunnen zien wat er gebeurt.



- Geef een reeks van sleutels zodat als de sleutels in deze volgorde aan een initieel lege skew-heap toegevoegd worden de volgende heap het resultaat is – of bewijs dat een dergelijke volgorde niet bestaat.



- Wat is in het slechtste geval voor skew-heaps en leftist heaps

  (a) de diepte van de heap

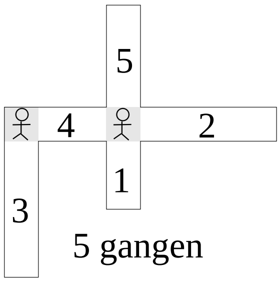
  (b) de lengte van een rechterpad

  (c) de lengte van een linkerpad

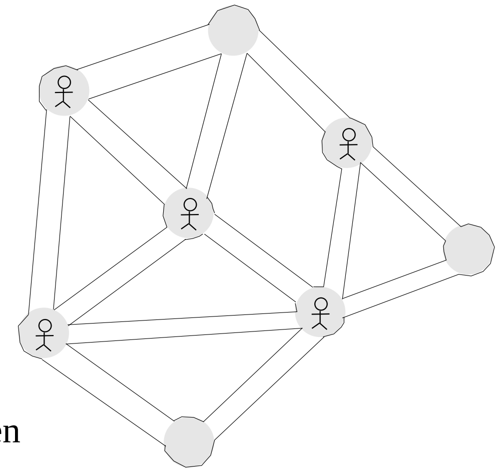
  als er n sleutels in de heap zitten? De $O()$-notatie is voldoende **maar** schrijf ook **expliciet** wat het betekent dat de diepte/lengte $O(f(n))$ is (dus de definitie van de $O()$-notatie). Je mag als $f(n)$ één van de van jou gekozen functies nemen om het uit te leggen.

3. **(3 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. Wij stellen dat er ten minste 2 gangen zijn en dat elke gang aan ten minste één kruispunt grenst.



5 gangen
2 kruispunten



13 gangen
8 kruispunten

De taak is nu een manier te vinden om bewakers op de kruispunten te plaatsen zodat het aantal bewakers zo klein mogelijk is maar elke gang door ten minste één bewaker gezien kan worden.

- Geef een gretig algoritme voor dit probleem in pseudocode. Het moet een algoritme zijn waarvan je in de praktijk zou verwachten dat het ten minste in de meeste gevallen een goede oplossing geeft!

- Toon aan dat jouw algoritme altijd het optimale aantal bewakers vindt of geef een voorbeeld waar de optimale oplossing door jouw algoritme niet gevonden kan worden.

- Stel nu dat niet de gangen maar de **kruispunten** bewaakt moeten worden. Het doel is dus bewakers zo op kruispunten te plaatsen dat elk kruispunt door ten minste één bewaker gezien kan worden. Er zijn k kruispunten en je mag bewakers op $\lceil k/5 \rceil$ kruispunten plaatsen. Je weet over het museum dat je van elk kruispunt ten minste d andere kruispunten kan zien en dat $k < 1.25^d$. Het doel is dus een manier te vinden om $\lceil k/5 \rceil$ bewakers zo te plaatsen dat elk kruispunt bewaakt is.

  Beschrijf een Las Vegas algoritme voor dit probleem en geef uitleg over jouw algoritme (op de manier die jullie in de les hebben gezien).

4. **(1.5 pt)**

   Bewijs de volgende stelling:

   Stelling:

   Voor elk online inpakalgoritme bestaat er een inputreeks zodat het algoritme ten minste $\frac{3}{2}m$ vrachtwagens gebruikt waarbij m het aantal vrachtwagens in een optimale oplossing is.

   Tip: Anders dan voor de stelling in de les zijn hier geen speciale eisen voor m. Daardoor kan een *gelijkaardig* bewijs gemakkelijker zijn en levert toch een betere grens op.

5. **(3.5 pt)**

- Toon aan dat $O(n \log n)$ een goede bovengrens voor een reeks van n bewerkingen op een initieel lege semi-splay boom is.

- Voeg de sleutels 1, 20, 7, 40, 33, 41 en 5 in deze volgorde toe aan een initieel lege semi-splay boom. Toon voldoende tussenstappen om te kunnen zien wat er gebeurt.

- In de les hebben wij het volgende deelresultaat bewezen:

  Deelresultaat 3: Stel dat een deelboom in een boom T op een semi-splay manier wordt vervangen en het resultaat is T'. Wij noemen de laagste top in de deelboom van T die wordt vervangen v_l (l voor *laag*) en de top die de nieuwe wortel van de nieuwe deelboom van T' wordt v_w (w voor *wortel*). Natuurlijk kan dat dezelfde top zijn!
  Dan geldt:
  $\Phi(T') - \Phi(T) \leq 2L_{T'}(v_w) - 2L_T(v_l) - 2.$

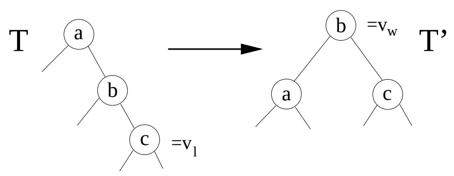
  Dit deelresultaat hebben wij bewezen voor de potentiaal $\Phi(T) = 0$ voor een lege boom T en anders

  $$\Phi(T) = \sum_{v \in T} L_T(v)$$

  met voor een top $v \in T$: $L_T(v) = \log A_T(v)$ en $A_T(v)$ het aantal toppen in de deelboom bestaande uit v en al zijn nakomelingen in T.
  Daarbij hebben wij de volgende opmerking gebruikt:

  Opmerking:
  Stel dat a, b, c natuurlijke getallen zijn en dat ze allemaal verschillen van 0. Als dan $a + b \leq c$ geldt dan is $\log a + \log b \leq 2\log c - 2$.

  Wij hebben in de les 2 gevallen van de semi-splay bewerking niet expliciet besproken, maar gewoon gezegd dat het *analoog met de andere gevallen* is. Eén van de niet gezien gevallen is de bewerking

  Bewijs nu expliciet Deelresultaat 3 voor deze bewerking.

6. **(1 pt)**

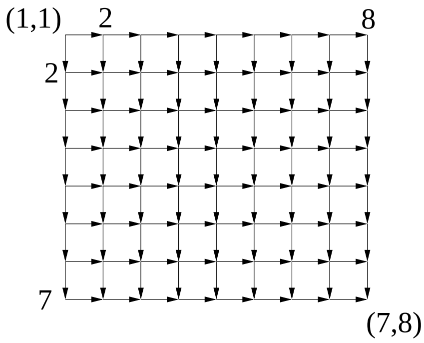
Stel dat T een boom met diepte k is die een verzameling voorstelt en dat T door union by size werd opgebouwd. en dat $m(k)$ het minimale aantal toppen is dat in een dergelijke boom aanwezig kan zijn.

- Geef een formule voor het minimale aantal toppen dat in een dergelijke boom T aanwezig kan zijn. (Geen uitleg vereist.)

- Bewijs of geef een tegenvoorbeeld (beide richtingen apart):
  Een dergelijke boom T bevat 2^k toppen als en slechts als T een binomiale boom is.

7. **(2 pt)**

In een rooster van $p \times q$ punten kan je altijd van een punt v met coördinaten (i,j) ($1 \leq i \leq p, 1 \leq j \leq q$) naar het punt met coördinaten $(i+1,j)$ gaan als $i+1 \leq p$ en naar het punt met coördinaten $(i,j+1)$ als $j+1 \leq q$. Het principe zie je in de volgende figuur.



Elke mogelijke stap van een punt v naar een punt w heeft een gewicht $g(v,w)$ met $0 \leq g(v,w)$. De kost van een pad in dit rooster is het **product** van de kosten van de gebruikte bogen. Gezocht is nu een goedkoopste pad van het punt met coördinaten $(1,1)$ naar het punt met coördinaten (p,q). Geef een algoritme met dynamisch programmeren in pseudocode en geef uitleg indien nodig.

NOG NIET OMDRAAIEN !