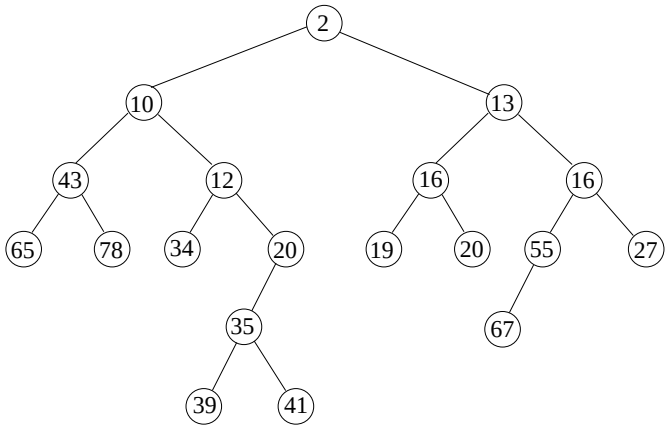
Examen Datastructuren en Algoritmen II

Naam : .....

1. **(4 pt)**

   - Bouw een leftist heap L_1 door de elementen $3, 19, 8, 1, 12,\ 16, 15, 7, 10, 9$ (in deze volgorde) toe te voegen aan een initieel lege heap. Toon voldoende tussenstappen om te kunnen zien wat je doet.

- Verwijder het kleinste element uit de volgende leftist heap. Toon voldoende tussenstappen om te kunnen zien wat je doet.



- Geef een lijst van vier verschillende heaps die je tijdens jouw studies gezien hebt.
  - Schrijf voor elke van de heaps hoe duur een reeks van n toevoegbewerkingen op een initieel lege heap is.

  - Kies drie van de vier heaps en geef voor elk van de zes paren (a, b) van heaps a en b één voordeel van heap a ten opzichte van heap b.

2. **(3 pt)**

   - Voeg de sleutels 7, 12, 22, 35, 15, 13 in deze volgorde toe aan een rood-zwart boom. Toon voldoende tussenstappen om te kunnen zien wat er gedaan wordt.

- Geef een benedengrens en een bovengrens voor het aantal **rode** toppen op het pad van de wortel naar een NULL-pointer in een rood-zwart boom met n sleutels. Geef uitleg en schets voorbeelden die tonen dat het goede grenzen zijn. De grenzen moeten natuurlijk een functie van n zijn. Stellingen uit de les die gebruikt worden, moeten genoemd maar niet opnieuw bewezen worden.

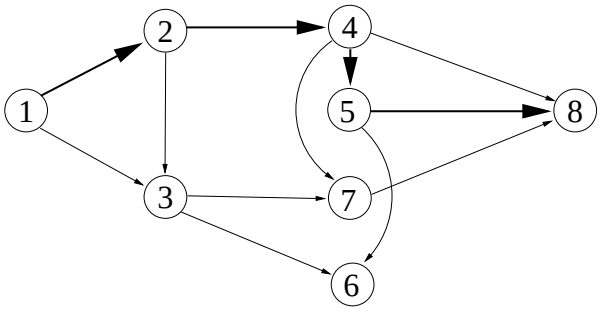
- Bedenk zelf een definitie van een 2-3-4 boom die de definitie van een 2-3 boom simuleert maar ook 4 kinderen per top toelaat. Zijn er overeenkomsten van deze 2-3-4 boom met rood-zwart bomen? (Tip: kijk naar de deelboom die door een zwarte top en zijn rode kinderen gevormd wordt.)

3. **(2 pt)**

Een spel op een graaf:

Gegeven een graaf met n toppen. De nummers van de toppen zijn $1, \ldots, n$. X en O kiezen afwisselend toppen. X begint met top 1 en elke keer dat een top werd gekozen, moet de andere speler een top kiezen die een buur van de net gekozen top is en een groter nummer heeft. De speler die het eerst niet meer kan kiezen, is verloren, de andere gewonnen. De winst is de som van de labels van de toppen en moet door de verliezer aan de winner betaald worden.

Een voorbeeld:



In dit voorbeeld win je als je top 6 of top 8 kan kiezen – deze toppen hebben geen grotere buren.

Als in het voorbeeld de vette lijnen de volgorde van keuzes aanduiden dan zou de rij van keuzes dus 1 (door X), 2 (door O), 4 (door X), 5 (door O) en ten slotte 8 (door X) zijn. X zou dus gewonnen zijn. De winst van X is $1+2+4+5+8=20$ en de *winst* van O is -20.

- Teken de spelboom voor de voorbeeldgraaf en bepaal de waarde van het spel.

- Beschrijf een **polynomiaal** algoritme dat voor een arbitraire graaf met toppen $1, \ldots, n$ de waarde van het spel op deze graaf bepaalt (polynomiaal in n). Geef de complexiteit van dit algoritme (de notatie $O(f(n))$ voor een geschikte functie $f()$ is natuurlijk voldoende) en leg uit waarom dat een bovengrens voor de complexiteit is. Tot welke klasse van algoritmen behoort jouw algoritme?

4. **(3 pt)**

   Bepaal voor de volgende twee datastructuren de geamortiseerde complexiteit van een reeks van n toevoegbewerkingen op een initieel lege datastructuur. De $O()$-notatie is voldoende. Je mag kiezen welke technieken je gebruikt – het moet alleen een juist bewijs zijn. Schets bovendien een voorbeeld van een reeks van sleutels dat aantoont dat jouw geamortiseerde complexiteit inderdaad een goede bovengrens is. Stellingen uit de les mogen natuurlijk gebruikt worden!

   - Een rood-zwart boom. Neem het aantal tijdens het toevoegen bezochte toppen als de kost van een toevoegbewerking.

- Een array. Je begint met een leeg array met maar tien elementen en elke keer dat een array met m elementen volzit, vervang je het door een array met $\lceil \frac{3m}{2} \rceil$ elementen. De kost van een toevoegbewerking is het aantal gekopieerde elementen plus één .

5. **(2.5 pt)**

- Lees de volgende uitspraken heel zorgvuldig. Het zijn de kleine details in de uitspraken die het verschil maken met de gelijkaardige stelling uit de les en de oefening die jullie hebben gemaakt! Maar de technieken die jullie daar hebben gezien, zullen ook hier heel nuttig zijn…!

- Toon aan of geef een tegenvoorbeeld: voor elk online inpakalgoritme $\mathcal{A}$ en elke $n \in \mathbb{N}$ bestaat er een inputreeks met ten minste n gewichten zodat $\mathcal{A}$ ten minste $\frac{3}{2}$ keer het optimale aantal vrachtwagens gebruikt.

- Toon aan of geef een tegenvoorbeeld: voor elke inputreeks $g_1, \ldots, g_n$ met $n \geq 2$ en $g_i \leq 1, 1 \leq i \leq n$ bestaat er een online inpakalgoritme $\mathcal{A}$ zodat $\mathcal{A}$ ten minste $\frac{3}{2}$ keer het optimale aantal vrachtwagens gebruikt.

- Geef een voorbeeld van een inputreeks waarvoor het online algoritme *first fit* beter presteert (minder vrachtwagens gebruikt) dan de offline heuristiek *first fit dalend* of toon aan dat een dergelijk voorbeeld niet bestaat.

6. **(1.5 pt)**

Door een grote school voor permanente vorming worden $l = 50$ verschillende lessen met betrekking tot informatica aangeboden. Er zijn $s = 1000$ studenten en elke student volgt ten minste $a = 8$ lessen. De bedoeling is dat aan de ene kant in elke les maar één besturingssysteem wordt gebruikt maar aan de andere kant de studenten in alle lessen die ze volgen niet altijd hetzelfde besturingssysteem zien maar tenminste twee van de drie belangrijkste besturingssystemen Gnu-Linux, MAC-OS en Windows. Het probleem is nu te beslissen in welke les welk besturingssysteem gebruikt moet worden.

- Beschrijf een Las Vegas algoritme voor dit probleem dat na één poging met kans ten minste $1/2$ een oplossing voor deze school heeft gevonden. Eén poging moet voor variabele l, s, a (dus niet de vaste waarden voor de gegeven school) in tijd $O(l + s * a)$ uitgevoerd kunnen worden.

- Bewijs dat jouw algoritme aan de voorwaarden voldoet.

NOG NIET OMDRAAIEN !