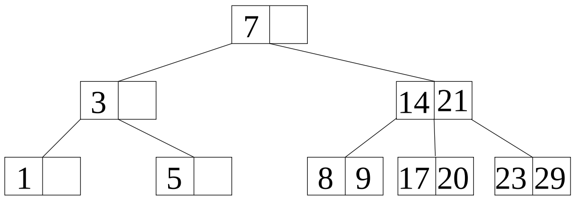
Examen Datastructuren en Algoritmen II

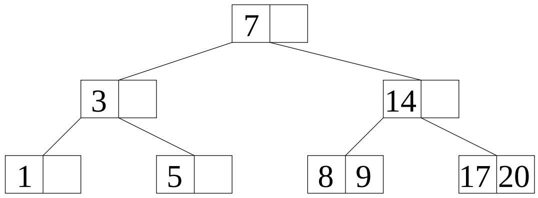
Naam : ....

1. **(3 pt)**

   - Voeg sleutel 18 toe aan de volgende 2-3 boom. Schrijf voor de tussenstappen welke deelboom je door welke andere deelboom vervangt.



- Verwijder sleutel 5 uit de volgende 2-3 boom. Schrijf voor de tussenstappen welke deelboom je door welke andere deelboom vervangt.



- Is de volgende uitspraak juist? Geef een bewijs of een tegenvoorbeeld:

  Als een rood-zwart boom en een 2-3 boom dezelfde sleutels bevatten en dezelfde nieuwe sleutel wordt toegevoegd aan beide bomen dan is het aantal bezochte toppen in de 2-3 boom altijd ten hoogste even groot als in de rood-zwart boom.

- Vergelijk semi-splay bomen en rood-zwart bomen. Wat zijn de voordelen en de nadelen? Voor welke toepassingen zou je eerder een rood-zwart boom kiezen en voor welke eerder een semi-splay boom?

2. **3 pt**

Tip: als je modulo m moet rekenen, reken soms met negatieve en soms met positieve getallen – kies gewoon het getal waarvan de absolute waarde kleiner is. Dat is **geen** eis voor deze oefening maar het helpt misschien om stomme rekenfouten te vermijden.

Gegeven een getal n waarvoor wij willen testen of het priem is en een toevallig gekozen getal b.

- Beschrijf de twee gevallen wanneer de Miller-Rabin test zegt dat een getal geen priemgetal is. Waarop zijn deze beslissingen gebaseerd?

- Welke van de twee antwoorden *is priem* en *is niet priem* is gegarandeerd juist? Geef uitleg.

- Pas de Miller-Rabin test toe op $n = 30$ en $b = 7$.

- Pas de Miller-Rabin test toe op $n = 28$ en $b = 9$.

3. **(3 pt)**

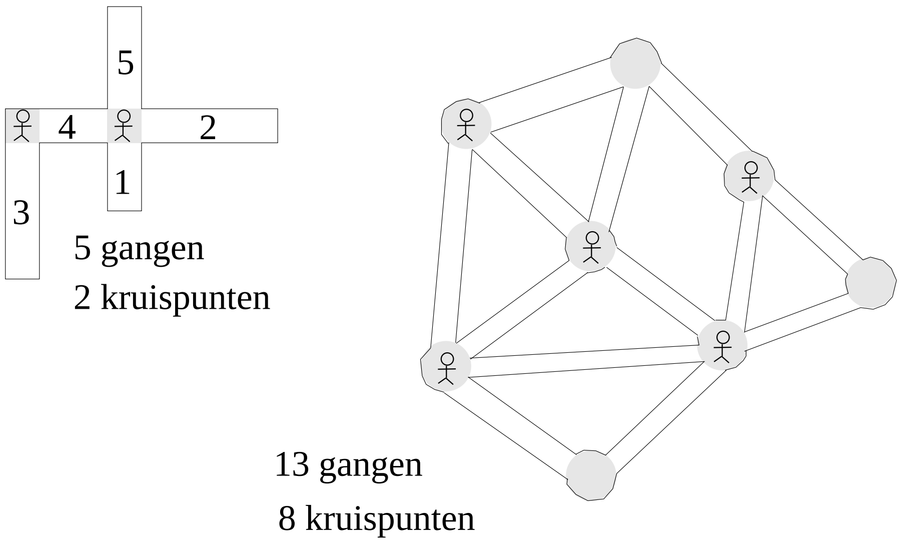
- In de les hebben wij de toevoegoperatie voor een binomiale prioriteitswachtlijn vergeleken met het optellen van een binaire teller en daardoor bewezen dat een reeks van n toevoegoperaties op een initieel lege hoop een geamortiseerde kost van ten hoogste $O(n)$ heeft. Bewijs dat nu direct door ofwel de potentiaalmethode ofwel de accountingmethode toe te passen. Natuurlijk mag jouw bewijs op dat voor de binaire teller lijken…

- Stel nu dat de wachtlijn niet leeg is maar dat er al n elementen door vroegere toevoegoperaties in de wachtlijn geplaatst werden. Is dan de geamortiseerde kost van de volgende n bewerkingen nog altijd $O(n)$? Bewijs dat jouw antwoord juist is.

- Bepaal de geamortiseerde complexiteit van een reeks van n toevoeg- of verwijderbewerkingen op een initieel lege leftist heap. Stellingen uit de les mogen natuurlijk gebruikt worden.

4. **(2 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.

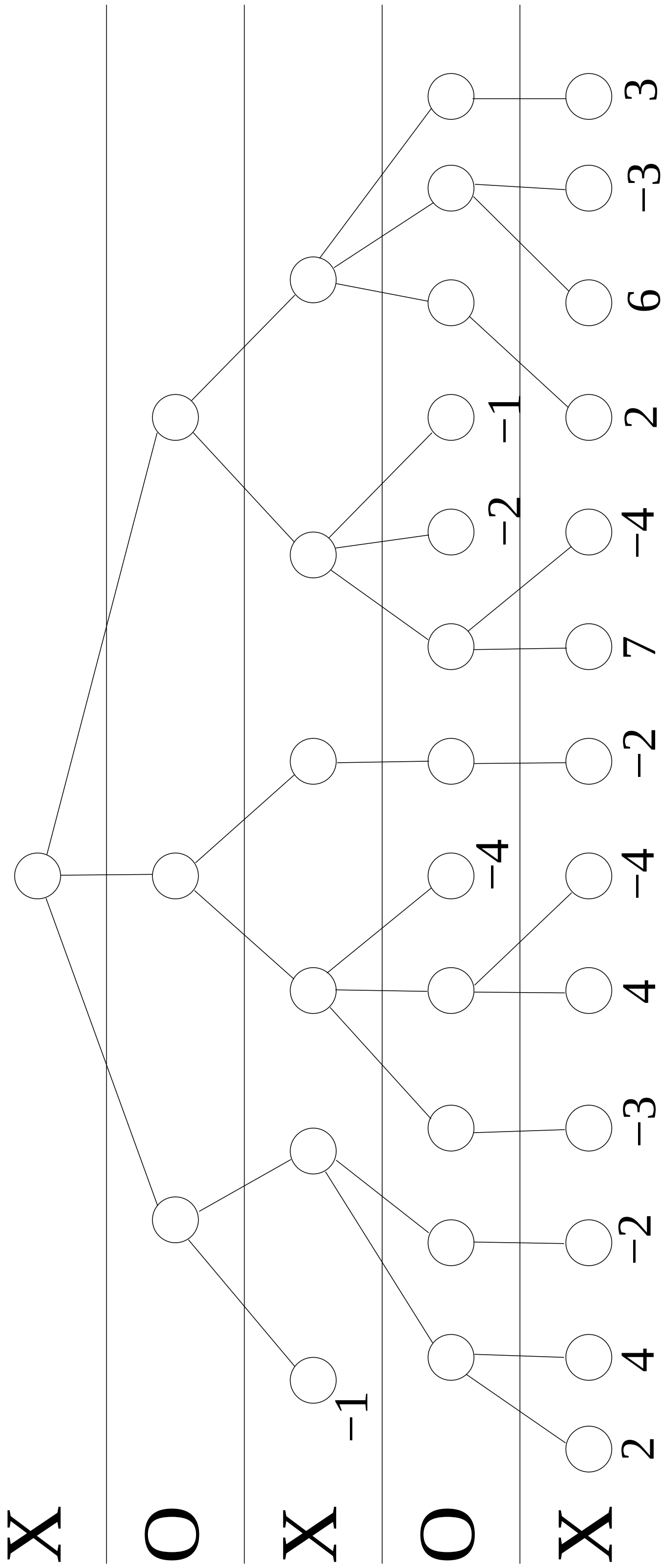


De taak is nu een manier te vinden om bewakers op de kruispunten te plaatsen zodat het aantal bewakers zo klein mogelijk is, maar elke gang door ten minste één bewaker gezien kan worden.

- Geef een branch and bound algoritme voor dit probleem in pseudocode. Ten minste één van de bounding criteria moet niet triviaal zijn – dus niet alleen testen of de deeloplossing al niet meer aan de eisen voldoet, maar een echte look-ahead mogelijk maken.

5. **(1 pt)**

Bereken de waarde van dit spel. Pas α-β-snoeien toe. Schrijf volgnummers bij de bogen om te kunnen zien in welke volgorde je de boom hebt doorlopen en markeer de toppen waar je α-β-snoeien hebt toegepast. Geef ook de tijdelijke grenzen voor de waarden van de toppen.



6. **(2 pt)**

Gegeven is een tekst van n woorden $w_1, \ldots, w_n$ die gescheiden zijn door een enkele spatie en die (nog) geen regeleinde-tekens bevat. De taak is nu voor een gegeven maximaal aantal tekens l (zonder regeleinde-teken) in een regel spaties te vervangen door regeleinde-tekens zodat het totale aantal regels minimaal is maar er nooit te veel tekens op een regel staan.

- Geef een gretig algoritme (in pseudocode) voor dit probleem.

- Bewijs dat jouw algoritme inderdaad een optimale oplossing vindt.

(2 pt)

Opnieuw is een tekst van n door spaties gescheiden woorden $w_1, \ldots, w_n$ zonder regeleinde-tekens gegeven. Ook hier willen wij spaties vervangen door regeleinde-tekens zodat geen regel meer dan een gegeven aantal l (zonder regeleinde-teken) van tekens bevat. Maar nu willen wij ook dat de tekst er zo gelijkmatig mogelijk uitziet. Daarom kennen wij aan elke regel r een fout $f(r)$ toe die het kwadraat van het aantal vrije tekens in de regel na het laatste woord is. De fout van de hele tekst is dan de som van de fouten van alle regels en wij willen regeleinde-tekens zo toekennen dat de fout van de tekst minimaal is.

Voorbeeld: De woorden zijn `dit_is_een_voorbeeld` en de maximale lengte van een lijn is 9. Dan heeft de oplossing

```
dit_is___
een______
voorbeeld
```

een fout van $3^2 + 6^2 + 0^2 = 45$.

- Toon aan dat jouw gretig algoritme uit de vorige oefening hier niet noodzakelijk een optimale oplossing vindt.

- Geef een dynamisch programmeren algoritme in pseudocode dat een optimale oplossing bepaalt.

  De volgende tip **mag** gebruikt worden maar **hoeft niet** gebruikt te worden:

  Als $F(i, j)$ de minimale fout van de tekst beschrijft die begint met woord i en eindigt met woord j en het eerste regeleinde-teken is na woord $k < n$, dan is $F(1,n) = F(1,k) + F(k+1,n)$.

NOG NIET OMDRAAIEN !