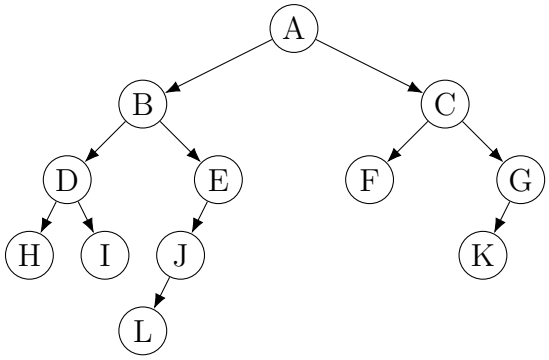
Het tentamen bevat vijf opgaven. Geef steeds voldoende uitleg. Succes.

1. a) Wat is een abstracte datastructuur (ADT)?

b) Beschrijf de ADT *priority queue*. Wat wordt opgeslagen en wat zijn de operaties?

c) Beschrijf de implementatie van een *binaire heap* als priority queue. Welke eigenschap heeft de binaire heap? Beschrijf de operaties.

2. a) Breng in de onderstaande boom symmetrische bedrading aan (alleen rechter draden).



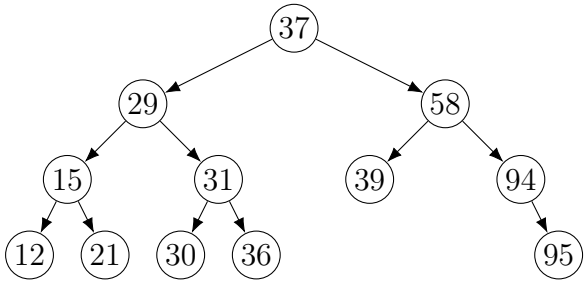
b) Beschrijf het maken van de inorde-wandeling in de bovenstaande bedrade boom. Geef duidelijk aan wanneer een draad wordt gevolgd en geef ook het resultaat van deze wandeling.

c) Welk voordeel heeft een symmetrisch bedrade boom op een onbedrade boom? Welk nadeel staat hier tegenover?

d) Geef de pseudo-code voor een functie die recursief de hoogte van alle knopen in een *binaire boom* berekent.

3. a) Een *Fibonacci boom* van hoogte h is een *AVL-boom* van hoogte h met zo weinig mogelijk knopen. Geef voor $h = 1, 2, 3, 4$ en 5 een Fibonacci boom van hoogte h. Het gaat alleen om de vorm van de boom; de waarden van de knopen hoeven niet te worden gegeven.

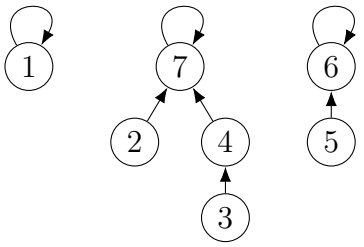
b) Voeg achtereenvolgens de getallen $34, 35$ en 96 toe aan de onderstaande AVL-boom. Uiteraard dient de boom na iedere toevoeging de AVL-eigenschap te bezitten, dus pas zonodig rotaties toe. Geef via tussenresultaten en een korte toelichting duidelijk aan hoe U aan uw antwoord komt.



c) Verwijder achtereenvolgens de getallen $39, 12$ en 29 uit de bovenstaande AVL-boom (de originele boom dus).

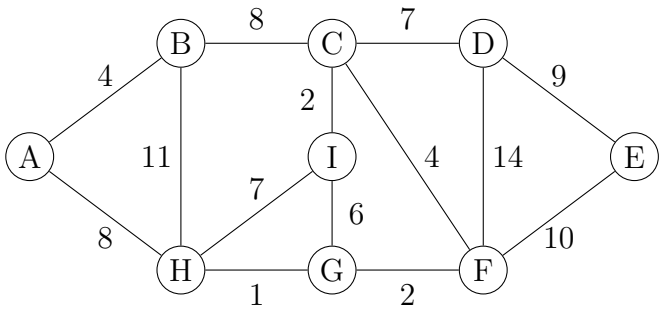
4. a) De ADT *union-find* houdt een aantal disjuncte verzamelingen bij. Geef een abstracte beschrijving van de ADT *union-find*, d.w.z. beschrijf het domein en de operaties.

b) Bij een mogelijke implementatie worden (omgekeerde) bomen gebruikt. Geef een *array*-implementatie van deze boom-representatie voor de onderstaande disjuncte verzamelingen.



c) Geef het algoritme van Kruskal in pseudo-code, waarbij gebruik wordt gemaakt van de ADT *union-find*.

d) Pas het algoritme van Kruskal toe op de onderstaande graaf. Leg uit hoe de ADT *union-find* wordt gebruikt.



e) Leg met een voorbeeld uit waarom het algoritme van Kruskal niet werkt op gerichte grafen.

f) Hebben twee verschillende *minimaal opspannende bomen* voor dezelfde graaf altijd tenminste één gemeenschappelijke tak? Leg uit.

5. a) In deze opgave beschouwen we de *ZLW (de-)codering* van teksten met symbolen `A`, `C`, `G` en `T`. De codeerboom (een trie) die bij (de-)codering wordt opgebouwd, is dus 4-air. Als eerste code, voor de *string* `A`, spreken we de waarde 1 af. Codeer de tekst `GGGG CGGA GCTG GGCG` (spaties staan hier alleen voor de leesbaarheid) met het ZLW algoritme. Geef telkens welke substrings worden 'geleerd' en de uiteindelijke codeerboom.

b) Veronderstel dat we bij codering van een tekst met het ZLW algoritme de volgende reeks codes hebben gekregen: 2 3 2 7 1 6 4 7. Reconstrueer de originele tekst. Bouw ook nu (achteraf) de codeerboom op en laat zien welke substrings van de tekst met welke code zijn gecodeerd. Geef toelichting bij de stappen die U achtereenvolgens maakt. NB: dit onderdeel staat (in principe) los van het vorige onderdeel. Wel spreken we weer de waarde 1 af voor string `A`.

c) Welk probleem lost het algoritme van Huffman op?

d) Wat is een voordeel van de ZLW codering op de codering van Huffman?