Tweede serie sommen.

1. (6 punten) Een boom is een ongerichte graaf waarbij tussen elk paar knopen precies één pad is. Bedenk een algoritme die het langste pad in een boom vindt. Wat is de complexiteit van uw algoritme?

2. (6 punten) Een skischool heeft n leerlingen en n paren skis. De regel is dat de lengte van de skis gelijk moet zijn aan de lengte van de leerlingen. Dat gaat natuurlijk niet, maar kan wel worden geoptimaliseerd door te eisen dat de som van de verschillen van de lengtes zo klein mogelijk moet zijn. Bedenk een greedy algoritme die de skis van lengte h_j zo verdeelt over de leerlingen van lengte ℓ_i dat $\sum |\ell_i - h_j|$ minimaal is (leerling met lengte ℓ_i krijgt skis van hoogte h_j). Wat is de complexiteit van uw algoritme?

3. (2+3+2 punten) Een greedy algoritme voor MST kan (ook) gebaseerd worden op de volgende observatie: Gegeven een cycle in een graaf en een kant e in die cycle met maximaal gewicht, er is een MST waarin die kant niet zit.

   (a) Bewijs dit.

   (b) Geef een greedy algoritme gebaseerd op dit principe.

   (c) Wat is de complexiteit van uw algoritme?

4. (2+3+2 punten) Input is een ongesorteerde verzameling rationale getallen, bijvoorbeeld $\{\frac{7}{4}, \frac{7}{2}, \frac{1}{2}, 2, \frac{3}{2}, 0\}$. Doel van de algoritme is het vinden van een minimale verzameling éénheidsintervallen die de input overdekt, in dit geval $\{[0,1], [\frac{5}{4}, \frac{9}{4}], [\frac{5}{2}, \frac{7}{2}]\}$.

   (a) Een greedy strategie zou kunnen zijn: vind het eenheidsinterval dat de meeste input-getallen overdekt. Noteer deze en streep de overdekte getallen weg. Herhaal totdat er geen getallen over zijn. Bewijs dat deze strategie niet werkt.

   (b) Vind een greedy strategie die wel werkt en bewijs dat deze werkt.

   (c) Wat is de complexiteit van deze strategie?

5. (3 punten) Bedenk een algoritme die in een graaf tussen twee punten A en B het op één na kortste pad vindt.

6. (programmeren 10 punten) Onderdelen van de opgave: documentatie (evt. in de vorm van commentaar); code; complexiteitsanalyse; testresultaten.
   De naïve methode van kortste pad bepalen is het terugvinden van alle paden van A naar B in een graaf en de kortste daarvan selecteren. Maak een random graaf en vergelijk daarop de complexiteit van de algoritme van Dijkstra met de naïve algoritme.
   Een random graaf (op bijv. 20 knopen) maak je bijvoorbeeld met:

```
int graph[20][20];
int i,j;
srand(4321854);
for(i=0;i<20;i++)
for(j=0;j<20;j++)
if( rand() %2 )
graph[i][j]=rand() %100;
```

```
else
graph[i][j]=-1;
```

In de knopen van de boom kan de lengte van het pad worden bijgehouden. Het is waarschijnlijk al genoeg om de wall-clock te gebruiken om de tijd besteed door de algoritmen te vergelijken. Naarmate je de n (aantal knopen) groter neemt. Zal het verschil in bestede tijd ook groeien.