Eksamen 9. desember 2020 **IDATA2302 Algoritmer og Datastrukturer**
Eksamenstid $4\frac{1}{2}$ timar *Nynorsk*

1. Sett at me har to sorterte lister a og b. Objekta i lista kan ha vilkårleg type, so lenge der er ein ordningsrelasjon $\leq$ på dei.

   (a). Gje ein mest mogleg effektiv algoritme som flettar dei to listene, dvs. returnerer ei sortert liste c som inneheld alle elementa frå a og b.

   (b). Forklar kvifor denne algoritmen er korrekt.

   (c). Finn tidskompleksiteten og grunngje svaret.

2. Redigeringsavstanden (*Edit*-avstanden eller levenshteinavstanden) mellom to strengar a og b er definert som talet på redigeringsoperasjonar ein treng for å transformera a til b eller omvendt. Tre redigeringsoperasjonar er definerte, *insert*, *delete* og *replace*.

   Fylgjande algoritme reknar ut redigeringsavstanden mellom delstrengane $s_{1..m}$ og $t_{1..n}$.

```
Algorithm editDistance(s, t, m, n):

if m = 0:   return n
if n = 0:   return m

if s[m] = t[n]: return editDistance(s, t, m-1, n-1)

return 1 + min(editDistance(s, t, m, n-1),     # Insert
               editDistance(s, t, m-1, n),     # Remove
               editDistance(s, t, m-1, n-1)    # Replace
               )
```

   (a). Forklar kvifor denne algoritmen korrekt finn redigeringsavstanden.

   (b). Vurder tidskompleksiteten på algoritmen.

   (c). Der finst ein standardteknikk som løyser problemet med vesentleg lågare tidskompleksitet (i verste fall). Kva teknikk er det tale om?

   (d). Gjev pseudokode for den meir effektive løysinga.

   (e). Forklar kort kvifor den nye løysinga gjev same svar som den rekursive løysinga over.

3. *LinkedHashMap* er ein datastruktur som implementerer både ei lenka liste og ein spreietabell (*hash table*). I tillegg til oppslag, sletting og innsetjing i konstant tid, som i ein vanleg spreietabell, skal den lenka lista gjera det mogleg å iterera over alle elementa i den rekkjefylgja elementa vart innsetne (FIFO).

   (a). Forklar korleis datastrukturen for *LinkedHashMap* kan byggjast opp. Bruk objekt- og gjerne klassediagram til hjelp. Hugs at du må støtta *delete()* i konstant tid i neste deloppgåve.

   (b). Forklar korleis *delete()*-operasjonen kan gjerast i konstant tid. Hugs at du må sletta både i lista og i tabellen.

   (c). Forklar kort korleis *insert()*-operasjonen kan gjerast i konstant tid.

4. Sett at me har ei usortert liste a med n element. Me skal finna det $i$te største elementet. Dersom me sorterer lista i synkande orden fyrst, kan me sjølvsagt henta ut det $i$te elementet relativt enkelt.

   (a). Kva er køyretidskompleksiteten på denne naïve løysinga inklusive sortering? Grunngje svaret kort.

   (b). Der er fleire moglege løysingar som gjev lågare kompleksitet (anten i verste fall eller i gjennomsnitt). Gje pseudo-kode for minst éi og forklar kvifor ho gjev korrekt svar.

   (c). Vurder kompleksiteten på løysinga som du har vald.