Programmering og Problemløsning
Datalogisk Institut, Københavns Universitet
Uge(r)seddel 4 – gruppeopgave

Torben Mogensen

Deadline 29. september

I denne periode skal I arbejde i grupper. Formålet er at arbejde med lister.

Opgaverne i denne uge er delt i øve- og afleveringsopgaver.

Øveopgaverne er:

4.1. HR: 4.1, 4.7, 4.8, 4.14, 4.16.

4.2. En tabel kan repræsenteres som en liste af lister, hvor alle listerne er lige lange.

Listen `[[1; 2; 3]; [4; 5; 6]]` repræsenterer for eksempel tabellen

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

i Lav en funktion `isTable : 'a list list -> bool`, der givet en liste af lister afgør, om det er en lovlig ikke-tom tabel, altså om alle listerne har ens længde, og at der er mindst en liste med mindst et element.

ii Lav en funktion `transpose : 'a list list -> 'a list list`, der *transponerer* en tabel. Transponering er spejling over diagonalen, så den transponerede tabel til den herover viste tabel er

$$\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

Kaldet `transpose [[1; 2; 3]; [4; 5; 6]]` skal altså returnere `[[1; 4]; [2; 5]; [3; 6]]`. Bemærk, at `transpose (transpose t) = t`, hvis `t` er en tabel. Hvis argumentet til `transpose` ikke er en lovlig tabel, skal en passende fejl rapporteres.

Afleveringsopgaven er:

4.3. HR: 4.4, 4.12 og 4.15.

4.4. Lav en funktion `removeDuplicates : 'a list -> 'a list when 'a : equality`, som fjerner duplikater i en liste. For eksempel skal kaldet `removeDuplicates [1; 2; 1; 3: 2]` give resultatet `[1; 2; 3]`. Bemærk, at den første forekomst af et givet elememt bevares, mens de øvrige forekomster fjernes. De bevarede elementer bevarer deres indbyrdes position.

Vink: I kan bruge `isMember` funktionen fra side 79 i HR.

Afleveringsopgaven skal aflevres som både LaTeX, den genererede PDF, samt en fsx tekstfil med løsningen for hver delopgave, som kan oversættes med fsharpc, og hvis resultat kan køres med mono. Det hele samles i en zip-fil med navnekonventionen

<instructor's initials>_<firstname.lastname$_1$>_..._<firstname.lastname$_n$>_<exercise-number>.zip

I zip-filen skal en delopgave navngives med opgavenummer, således at filen for opgave 4.3 hedder opg4_3.fsx osv.

Aflever kun en gang per gruppe, og brug Absalon's gruppeafleveringsfunktion.

God fornøjelse

Ugens nød 1

Vi vil i udvalgte uger stille særligt udfordrende og sjove opgaver, som interesserede kan løse. Det er helt frivilligt at lave disse opgaver, som vi kalder "Ugens nød", men der vil blive givet en mindre præmie til den bedste løsning, der afleveres i Absalon.

Ugens nød i denne uge omhandler en variant af spillet „Minestryger" (`http://da.wikipedia.org/wiki/Minestryger_%28spil%29`).

I vores variant er det for alle felter på forhånd kendt, hvor mange bomber, der i alt er i de otte nabofelter samt feltet selv. Det gælder både felter med og felter uden bomber.

Et eksempel på et sådant spil er

```
221
232
232
```

svarende til bombeplaceringen

```
000
110
001
```

hvor tomme felter er angivet med `0` og bomber med `1`.

Der er ikke altid en entydig løsning. For eksempel vil spillet

```
11
11
```

have fire løsninger, blandt andet:

```
01   og   10
00        00
```

Der er heller ikke altid løsninger, idet for eksempel

```
10
00
```

ikke har en løsning.

Vi repræsenterer et spil som en tabel af heltal, som alle ligger mellem 0 og 9. Løsninger repræsenteres som tabeller af heltal, der alle har værdi 0 eller 1.

Nød 1.1 funktion `minelaegger : int list list -> int list list`, som givet en *løsning* returnerer et spil. Det kan antages, at input er en liste af lister af tal, der alle er enten 0 eller 1.

For eksempel skal kaldet `minelaegger [[0; 0; 0]; [1; 1; 0]; [0; 0; 1]]` returnere spillet `[[2; 2; 1]; [2; 3; 2]; [2; 3; 2]]`.

Nød 1.2 Lav en funktion `minestryger : int list list -> int list list option`, som givet et spil finder en løsning, hvis en sådan findes, og returnerer `None`, hvis der ingen løsninger er. Det kan antages, at input er en liste af lister af tal mellem 0 og 9.

For eksempel skal kaldet `minestryger [[2; 2; 1]; [2; 3; 2]; [2; 3; 2]]` returnere løsningen `Some [[0; 0; 0]; [1; 1; 0]; [0; 0; 1]]`, mens `minestryger [[1; 0]; [0; 0]]` returnerer `None`, og `minestryger [[1; 1]; [1; 1]]` returnerer en af de fire mulige løsninger, f.eks. `Some [[0; 0]; [1; 0]]`

Det er ikke vigtigt, om løsningen findes hurtigt, men hvis flere korrekte løsninger indleveres, gives præmien til den hurtigste.

Der skal uploades både en LaTeX-fil, der beskriver fremgangsmåden, samt en fsx fil, der indeholder definitionerne af de to funktioner. Navngivningen af filerne er ikke vigtig.

Endnu eksempelspil til afprøvning er angivet herunder.

```
33333
33333
33333
33333
33333
```