Programmering og Problemløsning
Datalogisk Institut, Københavns Universitet
Arbejdsseddel 7 — gruppeopgave

Jon Sporring

24. oktober – 21. november.
Afleveringsfrist: onsdag d. 21. november kl. 22:00

I denne periode skal I arbejde i grupper. Formålet er at arbejde med sumtyper og endelige træer. Opgaverne er delt i øve- og afleveringsopgaver. I denne periode skal I arbejde i grupper. Formålet er at arbejde med:

- rekursion
- pattern matching
- sumtyper
- endelige træer

Øveopgaver

...

Afleveringsopgave

I denne opgave skal I programmere spillet Awari, som er en variant af Kalaha. Awari er et gammelt spil fra Afrika, som spilles af 2 spillere, med 7 pinde og 36 bønner. Pindende lægges så der dannes 14 felter ('pit' på engelsk), og bønnerne fordeles ved spillet start med 3 i hver af felterne, som ikke er hjemmefelterne. Dette er illustreret i Figur 1.

Spillerne skiftes til at spille en tur efter følgende regler:

- En tur spilles ved at spilleren tager alle bønnerne i et af spillerens felter 1-6 og placerer dem i de efterfølgende felter inkl. hjemmefelterne en ad gangen og mod uret. F.eks., kunne første spiller vælge at tage alle 3 bønner fra felt 4 og skulle derefter placere en bønne i hver af felterne 5, 6 og hjemmefeltet.

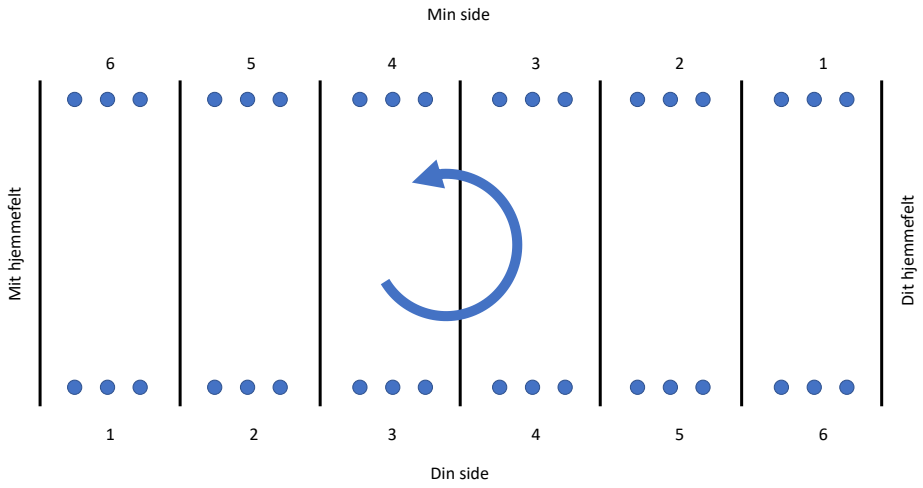

Figure 1: Udgangsopstillingen for spillet Awari.

- Hvis sidste bønne lægges i spilleres hjemmefelt, får spilleren en tur til.
- Hvis sidste bønne lander i et tom felt pånær hjemmefelterne, og feltet overfor indeholder bønner, så flyttes sidste bønne til hjemmefeltet, og alle bønnerne overfor fanges og flyttes ligså til hjemmefeltet.
- Spillet er slut når en af spillerne ingen bønner har i sine felter 1-6, og vinderen er den spiller, som efter spillets afslutning har flest bønner i sit hjemmefelt.

Afleveringsopgaven er:

7g.0 I skal implementere spillet Awari, som kan spilles af 2 spillere, og skrive en kort rapport. Kravene til jeres aflevering er:

- I skal bruge rekursion, sumtyper (discriminated unions), og Option typer. Jeres løsning kan f.eks. bygges op efter følgende skelet:

Listing 1 7gSkeleton.fsx:
Testing.

```
// Initial board, first 7 is player 1's 1-6 and home,
// second 7 is player 2's
let board = [|3;3;3;3;3;3;0;3;3;3;3;3;3;0|]
type player = Player1 | Player2

/// Print the board, player 1 is bottom row and rightmost
/// home.
let printBoard (board : int []) : unit =
  ...

/// Convert from user's coordinates to board's coordinates.
/// Players enters 1-6 corresponding to board index 0-5
/// or 7-12
let pit2ind (p : player) (pit : int) : int =
  ...

/// True if either side has no beans
let isGameOver (board : int []) : bool =
  ...

/// True if board index is the players home
let isHome (p : player) (ind : int) : bool =
  ...

/// Get the index of next move. User must enter a pit
/// number between 1-6 corresponding to a
/// non-empty board index
let rec getMove (p : player) (str : string) : int option =
  ...

/// Updates board by distributing beans counter clockwise,
/// capturing when relevant. Board coordinate of last bean
/// is returned. Sideeffect warning!
let distributeNUpdate (board : int []) (ind : int) : int =
  ...
```

- Koden skal dokumenteres vha. kommentarstandarden for F#
- Jeres aflevering skal indeholde en afprøvning efter white-box metoden.
- I skal skrive en kort rapport i LaTeX på maks. 10 sider og som indeholder:
  - en beskrivelse af jeres design og implementation
  - en gennemgang af jeres white-box afprøvning
  - kildekoden som appendiks.

Afleveringen skal bestå af en zip-fil og en pdf-fil. Zip-filen skal indeholde en mappe med fsharp koden. Koden skal kunne oversættes med fsharpc og køres med mono. Hvis der er flere filer, skal der være en kort beskrivelse af, hvordan man kører programmet/programmerne. Pdf-filen skal indeholde jeres LaTeX rapport oversat til pdf.