Learning to Program with F#
Exercises
Department of Computer Science
University of Copenhagen

Jon Sporring, Martin Elsman, Torben Mogensen, Christina Lioma

October 21, 2022

0.1 Awari

0.1.1 Teacher's guide

Emne Typer, lister, mønstergenkendelse, funktionsprogrammering

Sværhedsgrad Hård

0.1.2 Introduction

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. Pindene lægges så der dannes 14 felter ('pits' på engelsk), hvoraf 2 er hjemmefelter. Bønnerne fordeles ved spillet start med 3 i hvert felt pånær i hjemmefelterne. Startopstillingen er illustreret i Figur **??**.

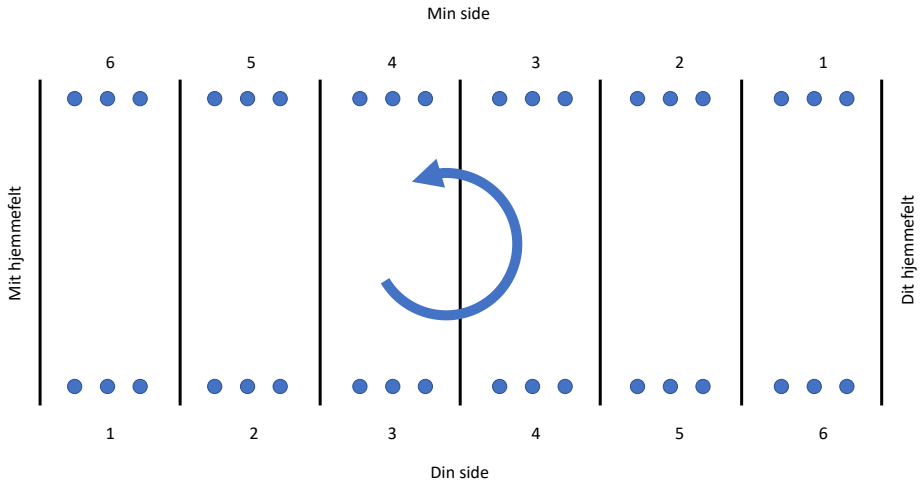

Figure 1: Udgangsopstillingen for spillet Awari.

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., kan første spiller vælge at tage bønnerne fra felt 4, hvorefter spilleren skal placere en bønne i hver af felterne 5, 6 og hjemmefeltet.
- Hvis sidste bønne lægges i spillerens hjemmefelt, får spilleren en tur til.
- Hvis sidste bønne lander i et tom felt som ikke er et hjemmefelt, og feltet overfor indeholder bønner, så flyttes sidste bønne til spillerens hjemmefelt, 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 har flest bønner i sit hjemmefelt.

0.1.3 Exercise(s)

0.1.3.1: (a) I skal implementere spillet Awari, som kan spilles af 2 spillere, og skrive en kort rapport. Kravene til jeres aflevering er:

- Koden skal organiseres som bibliotek, en applikation og en test-applikation.
- Biblioteket skal tage udgangspunkt i følgende signatur- og implementationsfiler:

Listing 1 awariLibIncompleteLowComments.fsi: En ikke færdigskrevet signaturfil.

```
module Awari
type pit = // intentionally left empty
type board = // intentionally left empty
type player = Player1 | Player2

/// Print the board
val printBoard : b:board -> unit

/// Check whether a pit is the player's home
val isHome : b:board -> p:player -> i:pit -> bool

/// Check whether the game is over
val isGameOver : b:board -> bool

/// Get the pit of next move from the user
val getMove : b:board -> p:player -> q:string -> pit

/// Distributing beans counter clockwise,
/// capturing when relevant
val distribute :
    b:board -> p:player -> i:pit -> board * player *
   pit

/// Interact with the user through getMove to perform
/// a possibly repeated turn of a player
val turn : b:board -> p:player -> board

/// Play game until one side is empty
val play : b:board -> p:player -> board
```

Listing 2 awariLibIncomplete.fs:
En ikke færdigskrevet implementationsfil.

```
module Awari
type pit = // intentionally left empty
type board = // intentionally left empty
type player = Player1 | Player2

// intentionally many missing implementations and
   additions

let turn (b : board) (p : player) : board =
  let rec repeat (b: board) (p: player) (n: int) :
   board =
    printBoard b
    let str =
      if n = 0 then
        sprintf "Player %A's move? " p
      else
        "Again? "
    let i = getMove b p str
    let (newB, finalPitsPlayer, finalPit)= distribute
   b p i
    if not (isHome b finalPitsPlayer finalPit)
       || (isGameOver b) then
      newB
    else
      repeat newB p (n + 1)
  repeat b p 0

let rec play (b : board) (p : player) : board =
  if isGameOver b then
    b
  else
    let newB = turn b p
    let nextP =
      if p = Player1 then
        Player2
      else
        Player1
    play newB nextP
```

En version af signaturfilen med yderligere dokumentation og implementationsfilen findes i Absalon i opgaveområdet for denne opgave.

- Jeres løsning skal benytte funktionsparadigmet såvidt muligt.
- 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.