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 Predator-Prey

0.1.1 Teacher's guide

Emne Inheritance

Sværhedsgrad Middel

0.1.2 Introduction

Consider a simulation of a natural habitat as two groups of animals interact. One group is the prey, a population of animals that are the food source for the other population of animals, the predators. Both groups have a fixed birthrate. The prey usually procreate faster than the predators, allowing for a growing prey population. But as the population of prey increases, the habitat can support a higher number of predators. This leads to an increasing predator population, and, after some time, a decreasing prey population. Around that time, the predator population grows so large as to reach a critical point, where the number of prey can no longer support the present predator population, and the predator population begins to wane. As the predator population declines, the prey population recovers, and the two populations continue this interaction of growth and decay.

An actual example of studying predator-prey relationships is the one between wolves and moose on Isle Royale in Lake Superior (http://www.isleroyalewolf.org/). Its population of wolves and moose are isolated on the island.

0.1.3 Exercise(s)

0.1.3.1: In the following, we will build a simulator of a predator-prey relationship in a closed environment using the following rules:

(a) The habitat updates itself in units of time called clock ticks. During one clock tick, every animal in the island gets an opportunity to do something.

(b) All animals are given an opportunity to move into an adjacent space, if an empty adjacent space is found. One move per clock tick is allowed.

(c) Both the predators and prey can reproduce. Each animal is assigned a fixed breed time. If the animal is still alive after breed time ticks of the clock, it will reproduce. The animal does so by finding an unoccupied adjacent space and fills that space with the new animal – its offspring. The animal's breed time is then reset to zero. An animal can breed at most once in a clock tick.

(d) The predators must eat. They have a fixed starve time. If they cannot find a prey to eat before starve time ticks of the clock, they die.

(e) When a predator eats, it moves into an adjacent space that is occupied by prey (its meal). The prey is removed and the predator's starve time is reset to zero. Eating counts as the predator's move during that clock tick.

(f) At the end of every clock tick, each animal's local event clock is updated. All animals' breed times are decremented and all predators' starve times are decremented.

Lav et program, som kan simulere rov- og byttedyrene som beskrevet ovenfor og skrive en lille rapport. Kravene til programmeringsdelen er:

(a) Man skal kunne angive antal af tiks (clock ticks), som simuleringen skal køre, formeringstid (breeding time) for begge racer og udsultningstid for rovdyrene ved programstart.

(b) Antallet af dyr per tik skal gemmes i en fil.

(c) Programmet skal benytte klasser og objekter

(d) Der skal være mindst en (fornuftig) nedarvning

(e) Programmets klasser skal bla. beskrives ved brug af et UML diagram

(f) Programmet skal kommenteres ved brug af fsharp kommentarstandarden

(g) Programmet skal unit-testes

Kravene til rapporten er:

(h) Rapporten skal skrives i LaTeX.

(i) I skal bruge `rapport.tex` skabelonen

(j) Rapporten skal som minimum i hoveddelen indeholde afsnittene Introduktion, Problemanalyse og design, Programbeskrivelse, Afprøvning, og Diskussion og Konklusion. Som bilag skal I vedlægge afsnittene Brugervejledning og Programtekst.

(k) Rapporten må maximalt være på 10 sider alt inklusivt.