We are simulating the animals of the tundra. There are colonies of prey and predator animals. The number of animals in a colony affect the number of animals in other colonies. There are three predator species: the snowy owl, the arctic fox and the wolf. There are three kinds of prey: the lemming, the arctic hare and the gopher.

If the number of prey animals increase, predators can reproduce more quickly. If the number of prey is very large, most of them will wander away because they cannot find enough food. If the number of predators is large, the number of the prey decreases quicker as they are preyed upon.

Each colony has a name, a species, and the number of animals in the colony. The prey species are affected by the different predator species as follows. The number of animals in their own colony changes first, then they influence the predators.

Lemming: If they are preyed upon by a predator colony, the number of animals in their colony decreases by four times the number of animals in the predator colony. The number of animals in their colony doubles every second turn. If there are less than 20 animals in the colony, all predator colonies have 20% less offsprings. If there are more than 200 animals in the colony, the number of animals in the colony decreases to 30.

Hare: If they are preyed upon by a predator colony, the number of animals in their colony decreases by double the number of animals in the predator colony. The number of animals in their colony grows by 50 percent (to one and a half times their previous number) every second turn. If there are less than 10 animals in the colony, every predator colonies have 20% less offsprings. If there are more than 100 animals in the colony, the number of animals in the colony decreases to 20.

Gopher: If they are preyed upon by a predator colony, the number of animals in their colony decreases by double the number of animals in the predator colony. The number of animals in their colony doubles every fourth turn. If there are less than 30 animals in the colony, every predator colonies have 20% less offsprings. If there are more than 200 animals in the colony, the number of animals in the colony decreases to 40.

Predators choose and attack a prey colony randomly in each turn. If there are not enough animals in the attacked colony (for example, there are not four times the number of predators in a lemming colony), the number of predators also decreases: every fourth predator out of the ones who didn't get prey perishes. Predators have offsprings every eighth turn. Normally, the snow owls have 1 offspring per 4 animals, the foxes have 3 offsprings per 4 animals, and the wolves have 2 offsprings per 4 animals. If the number of all prey animals is more than 10 times the number of predator animals, predators have 1 more offsprings: owls have 2 per 4 animals, foxes have 4 per 4 animals, and wolves have 3 per 4 animals.

The program should read the colonies from a text file. The first line contains the number of prey and predator colonies separated by a space. Each of the next lines contains the data of one colony separated by space: their name, their species, their starting number of animals. The species can be: o - owl, f - fox, w - wolf, l - lemming, h - hare, g - gopher.

**Simulate the process until each of the prey colonies becomes extinct or the number of prey animals quadruples compared to its starting value. Print the data of each colony in each turn.**

# Analysis

The class “Tundra” which is the simulation and representation of an environment that includes preys colonies and predators colonies. The Tundra class has a composition relationship with the Colony class (base), for Prey (abstract) and Predator (abstract), both are derived from Colony.

There are three animal classes in the prey colony and three in the predator colony. Each class inherits from Prey or Predator class. Each animal has its own methods for offspring and attacking. Tundra class will have methods and operations to calculate the total number of prey and predator animals to determine if the condition for increased predator offspring is met.

Visitor design pattern can be used to make the implementation easier, allowing external classes (visitors) run operations on the animal classes.

The design involves organizing the classes in a ordered structure, using inheritance and composition relationships, and implementing the interactions between the prey and predator animals within the Arctic Tundra.

Interactions Between Prey and predators' colonies:

| **Condition** | **Lemming** | **Hare** | **Gopher** |
| --- | --- | --- | --- |
| if preyed upon | cnt - (pred.cntx 4) | cnt - (pred.cnt x 2) | cnt - (pred.cnt x 2) |
| if increases in number | if cnt > 200 then cnt := 30 | if cnt > 100 then cnt := 20 | if cnt > 200 then cnt := 40 |

Offspring of Animals of Prey and Predator Colonies:

| Species | Each ... turns | Increased number |
| --- | --- | --- |
| Lemming | 2 | cnt x 2 |
| Hare | 2 | cnt x 1.5 |
| Gopher | 4 | cnt x 2 |
| Owl | 8 | cnt/4 (or 2 per 4 animals if prey > 10x predators) |
| Fox | 8 | (cnt/4) x 3 (or 4 per 4 animals if prey > 10x predators) |
| Wolf | 8 | (cnt/4) x 2 (or 3 per 4 animals if prey > 10x predators) |

Process of Perishing of Predators:

| **Condition** | **Lemming** | **Hare** | **Gopher** |
| --- | --- | --- | --- |
| number of predators if the number of prey is not enough | if prey.cnt < 4 \* cnt then cnt = cnt - ((cnt - prey.cnt) / 4) | if prey.cnt < 2 \* cnt then cnt = cnt - ((cnt - prey.cnt) / 4) | if prey.cnt < 2 \* cnt then cnt = cnt - ((cnt - prey.cnt) / 4) |

# Plan (UML)

|  |
| --- |
| Tundra |
| +turn: int  +endSim: bool |
| +Tundra()  +StartSim(preys:Prey[])  +EndSim(preys:Prey[], preyCount: Prey[]) |

0..\*

|  |
| --- |
| Colony |
| #name:string  #spec: string  #cnt : int |
| <getters>  +Colony(name:string, spec:string, cnt:int)  +Offspring():void {virtual} |

+preys +predators

|  |
| --- |
| <<abstract>>  Prey |
| +Prey(name:string, spec: string, cnt:int)  +AboveLimit():bool {virtual}  +NewCnt(): void {virtual}  +gAttacked(predator : Prey):void {virtual} |

|  |
| --- |
| <<abstract>> predator |
| +Predator(name:string, spec: string, cnt: int)  + AttackLemming(l:Lemming): void {virtual}  +AttackHare(h:Hare): void {virtual}  +AttackGopgher(g:Gopher) {virtual} |

|  |
| --- |
| Lemming |
|  |
| +Lemming(name:string, spec: string, cnt:int)  +AboveLimit():bool {override}  +NewCnt(): void {override}  +gAttacked(predator : Prey):void {override}  +Offspring(): void {override} |

|  |
| --- |
| Owl |
|  |
| +Owl(name:string, spec: string, cnt: int)  + AttackLemming(l:Lemming): void {override}  +AttackHare(h:Hare): void {override}  +AttackGopher(g:Gopher) {override}  +Offspring():void {override} |

|  |
| --- |
| Fox |
|  |
| +Fox(name:string, spec: string, cnt: int)  + AttackLemming(l:Lemming): void {override}  +AttackHare(h:Hare): void {override}  +AttackGopher(g:Gopher) {override}  +Offspring():void {override} |

|  |
| --- |
| Hare |
|  |
| +Hare(name:string, spec: string, cnt:int)  +AboveLimit():bool {override}  +NewCnt(): void {override}  +gAttacked(predator : Prey):void {override}  +Offspring(): void {override} |

|  |
| --- |
| Wolf |
|  |
| +Wolf(name:string, spec: string, cnt: int)  + AttackLemming(l:Lemming): void {override}  +AttackHare(h:Hare): void {override}  +AttackGopher(g:Gopher) {override}  +Offspring():void {override} |

|  |
| --- |
| Gopher |
|  |
| +Gopher(name:string, spec: string, cnt:int)  +AboveLimit():bool {override}  +NewCnt(): void {override}  +gAttacked(predator : Prey):void {override}  +Offspring(): void {override} |

# Specification

A = *prey*: *Preyn*, *predators*: *Predatorm*,  *endSim*: *bool*

Pre = *prey = prey’ AND predator* = predator’

Post = Ɐi ɛ[1..m]: (ꓱj ɛ [1..n]: prey[j].gAttacked(predator[i]) ʌ endSim =

m< prey [j]. GetName()> )

Analogy:

|  |  |
| --- | --- |
| enor(*E*) | *i = 1 .. n* |
| *f*(*e*) | <*prey*[*i*]*.getName()* |
| *Cond(e)* | Prey[i].endSim() |
| *s* | endSim() |
| H, +, | *prey\*,*  *,* *<>* |

endSim

*:*

=

False,turn:=1,

*i*

=

*1*

*..*

*n*

ʌ !endSim

***currPrey:=random(), currPred:=random***

prey[currPrey].gAttacked(pr)

endSim(), turn:= turn+1

# Testing

Black Box Testing:

1. Simulation end condition: all prey extinct

* Description: Verify the simulation ends when all prey are extinct.
* Test setup:
  + Add a Lemming with a minimal population of 1
  + Add an Owl with a sufficient population of 5 to extinct the prey quickly
* Expected Outcome: The simulation should terminate, endSim should be true, and the count of prey colonies should be zero
* Test Method: “TestSimulationEndsWhenPreyExtinct”

1. Simulation End Condition: Prey Population Quadruples

* Description: Verify the simulation ends when the prey population quadruples its initial size
* Test setup:
  + Add a Lemming with an initial population of 50
  + Add an Owl with a minimal population of 2
* Expected Outcome: The simulation should terminate, endSim should be true, and the Lemming population should be at least four times the initial size (200)
* Test Method: “TestSimulationEndsWhenPreyQuadruples”

1. Multiple Creatures Initialization

* Description: Verify the simulation handles multiple creatures correctly during initialization
* Test setup:
  + Add two prey colonies: Lemming (population 10) and Hare (population 5)
  + Add one predator colony: Owl (population 3)
* Expected Outcome: The counts of prey colonies should be 2, and predator colonies should be 1
* Test Method: “TestMultipleCreatures”

4. Total Prey Count Calculation

* Description: Verify that the total number of prey animals is calculated correctly
* Test setup:
  + Add two prey colonies: Lemming (population 50) and Hare (population 20)
* Expected Outcome: The total prey count should be 70
* Test Method: “TestTotalPreyCount”

1. Total Predator Count Calculation

* Description: Verify that the total number of predator animals is calculated correctly
* Test setup:
  + Add two predator colonies: Owl (population 5) and Fox (population 15)
* Expected Outcome: The total predator count should be 20
* Test Method: “TestTotalPredatorCount”