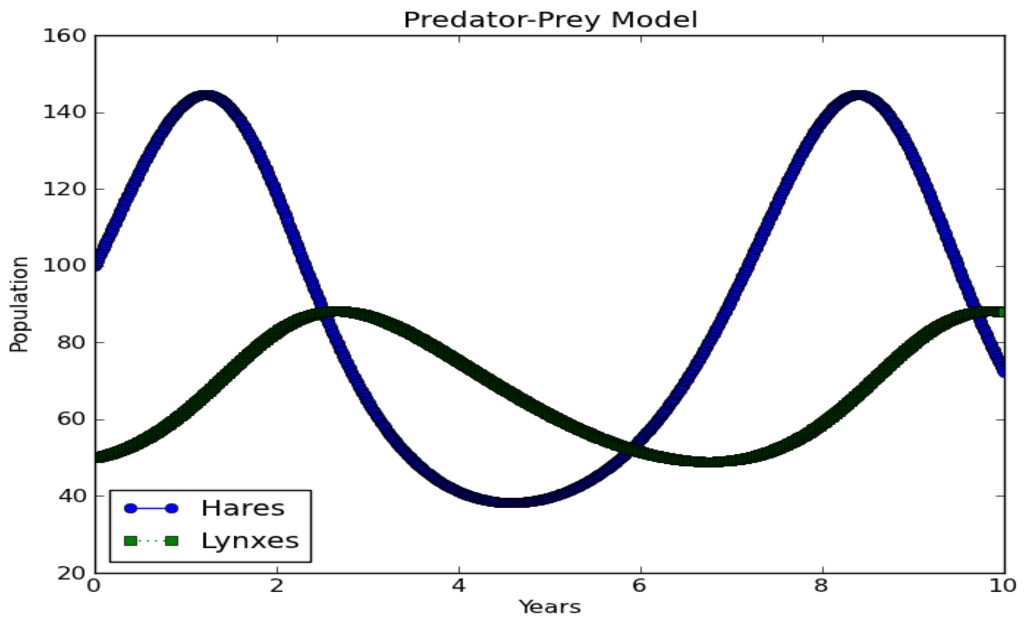
The hare birth rate is constant, as their food supply is unlimited. Hares only die when eaten by a lynx, and the number of hares eaten is proportional to how often hares & lynxes meet, i.e., the chance of a lynx catching a hare.

The lynx birth rate is also proportional to how often hares & lynxes meet, i.e., the food available for each lynx family. Lynxes only die from natural causes, and their death rate is constant.



Computational Thinking

Problem description

Abstraction

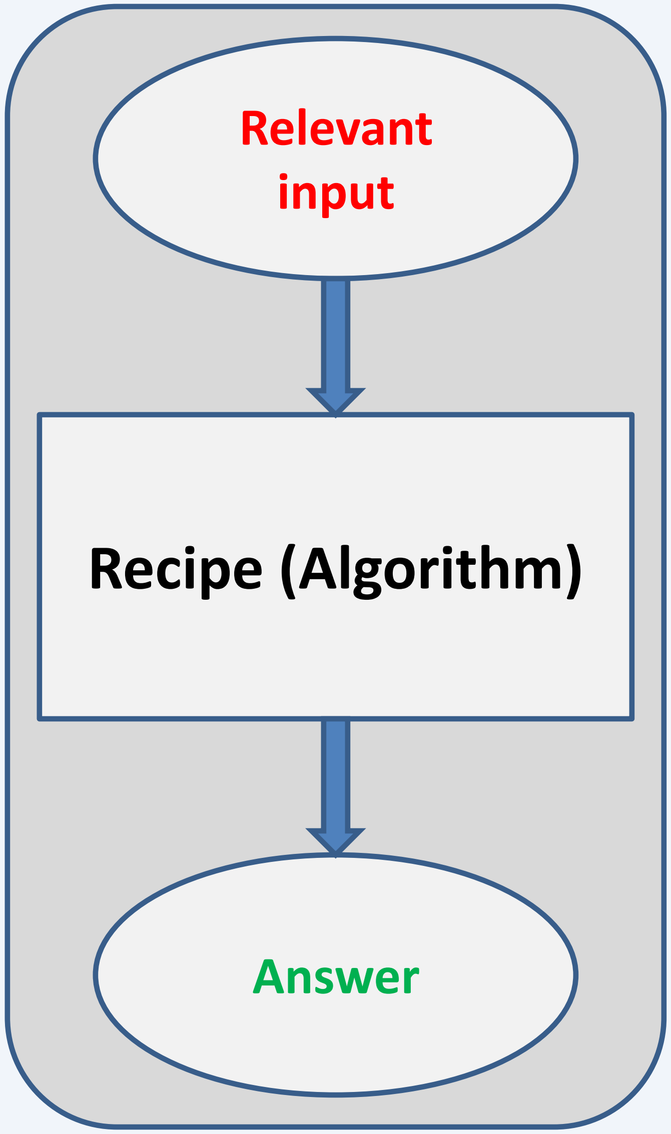
Automation

Computational goal

Information extraction

Algorithm design

Algorithm implementation



1. Generate population data.
    a. Repeatedly, generate next population.
2. Display population data.

Given $h_0, l_0, hareBirth, harePredation, lynxBirth, lynxDeath, years$.

Repeat these steps for $y = 0, \ldots, years - 1$:

$$h_{y+1} = h_y + h_y \cdot (hareBirth - harePredation \cdot l_y)$$
$$l_{y+1} = l_y + l_y \cdot (lynxBirth \cdot h_y - lynxDeath)$$

Plot (y, h_y) for $y = 0, \ldots, years$.

Plot (y, l_y) for $y = 0, \ldots, years$.

Improve Algorithm

- Separate parts
- Generalize
- Articulate goals

1. Generate population data.
    a. Repeatedly, generate next population.

---

2. Display population data.

Given $h_0, l_0, hareBirth, harePredation, lynxBirth,$ $lynxDeath, years.$

Repeat these steps for $y = 0, \ldots, years - 1$:

$$h_{y+1} = h_y + h_y \cdot \left(hareBirth - harePredation \cdot l_y\right)$$
$$l_{y+1} = l_y + l_y \cdot \left(lynxBirth \cdot h_y - lynxDeath\right)$$

Plot $\left(y, h_y\right)$ for $y = 0, \ldots, years.$

Plot $\left(y, l_y\right)$ for $y = 0, \ldots, years.$

Given $h_0, l_0, hareBirth, harePredation, lynxBirth,$
$lynxDeath, years.$

Repeat these steps for $y = 0, \ldots, years - 1$:

```
    $h_{y+1} = h_y + h_y \cdot \left(hareBirth - harePredation \cdot l_y\right)$
    $l_{y+1} = l_y + l_y \cdot \left(lynxBirth \cdot h_y - lynxDeath\right)$
```

Return $\boldsymbol{h}$ and $\boldsymbol{l}$.

Why Separate Parts?

- Simpler
- Independently useful

Generalize beyond Hares & Lynxes

Given $prey_0, pred_0, growth, predation, conversion,$
$death, years.$

```
Repeat these steps for $y = 0, \dots, years - 1$:
    $prey_{y+1} =$
     $prey_y + prey_y \cdot (growth - predation \cdot pred_y)$
    $pred_{y+1} =$
     $pred_y + pred_y \cdot (conversion \cdot prey_y - death)$
```

Return $prey$ and $pred$.

Articulate Goals

Populations:

Given $prey_0$, $pred_0$, $growth$, $predation$, $conversion$, $death$, $years$.

Returns the predicted populations of two species, given their initial populations, the prey's growth rate, the predation rate, the predator's food conversion rate, the predator's death rate, and the number of years to predict.

Repeat these steps for $y = 0, \ldots, years - 1$:

$$prey_{y+1} = prey_y + prey_y \cdot (growth - predation \cdot pred_y)$$
$$pred_{y+1} = pred_y + pred_y \cdot (conversion \cdot prey_y - death)$$

Return $prey$ and $pred$.

Express as Python Code

Use What Kinds of Python Data?

Populations:

Given $prey_0$, $pred_0$, $growth$, $predation$, $conversion$, $death$, $years$.

Returns the predicted populations of two species, given their initial populations, the prey's growth rate, the predation rate, the predator's food conversion rate, the predator's death rate, and the number of years to predict.

Repeat these steps for $y = 0, \ldots, years - 1$:

$$prey_{y+1} = prey_y + prey_y \cdot (growth - predation \cdot pred_y)$$
$$pred_{y+1} = pred_y + pred_y \cdot (conversion \cdot prey_y - death)$$

Return $prey$ and $pred$.

$prey$ and $pred$ as Lists

$$prey = \begin{array}{|c|c|c|c|c|c|}\hline prey_0 & prey_1 & prey_2 & prey_3 & prey_4 & \ldots \\ \hline \end{array}$$

$$pred = \begin{array}{|c|c|c|c|c|c|}\hline pred_0 & pred_1 & pred_2 & pred_3 & pred_4 & \ldots \\ \hline \end{array}$$

Translate Piece by Piece

Populations:

Given $prey_0$, $pred_0$, $growth$, $predation$, $conversion$, $death$, $years$.

Returns the predicted populations of two species, given their initial populations, the prey’s growth rate, the predation rate, the predator’s food conversion rate, the predator’s death rate, and the number of years to predict.

```
def populations(prey0,pred0,growth,predation,conversion,death,years):
    """Returns the predicted populations of two species, given their
       initial populations, the prey's growth rate, the predation
       rate, the predator's food conversion rate, the predator's
       death rate, and the number of years to predict."""

    prey = [prey0]
    pred = [pred0]
```

Translate Piece by Piece

Repeat these steps for $y = 0, \ldots, years - 1$:

```
for y in range(years):
```

Translate Piece by Piece

$$prey_{y+1} = prey_y + prey_y \cdot (growth - predation \cdot pred_y)$$
$$pred_{y+1} = pred_y + pred_y \cdot (conversion \cdot prey_y - death)$$

```
prey.append(prey[y] + prey[y] * (growth-predation*pred[y]))
pred.append(pred[y] + pred[y] * (conversion*prey[y]-death))
```

Translate Piece by Piece

Return $prey$ and $pred$.

```
return prey , pred
```

Put the Function's Pieces Together

```
def populations(prey0,pred0,growth,predation,conversion,death,years):
    """Returns the predicted populations of two species, given their
       initial populations, the prey's growth rate, the predation
       rate, the predator's food conversion rate, the predator's
       death rate, and the number of years to predict."""

    prey = [prey0]
    pred = [pred0]

    for y in range(years):
        prey.append(prey[y] + prey[y] * (growth-predation*pred[y]))
        pred.append(pred[y] + pred[y] * (conversion*prey[y]-death))

    return prey, pred
```

How to Use The Code?

```
pred, prey = populations(100,50,.4,.003,.004,.2,10)
```

A Useful Aside

```
def populations(prey0,pred0,growth,predation,conversion,death,years):
    """Returns the predicted populations of two species, given their
       initial populations, the prey's growth rate, the predation
       rate, the predator's food conversion rate, the predator's
       death rate, and the number of years to predict."""

    prey = [prey0]
    pred = [pred0]

    for y in range(years):
        print "y = ", y
        print "prey = ", prey
        print "pred = ", pred

        prey.append(prey[y] + prey[y] * (growth-predation*pred[y]))
        pred.append(pred[y] + pred[y] * (conversion*prey[y]-death))

    return prey, pred
```

Plotting

```
import matplotlib.pyplot as plt

def plotPopulations(times,prey,pred,preyName,predName):
    """Displays a plot of two populations over the given times."""

    # Prey use circles connected by solid line.
    preyPlot = plt.plot(times, prey, 'o-' )

    # Predators use squares connected by dotted line.
    predPlot = plt.plot(times, pred, 's:' )

    # Place legend box in "best" location.
    plt.legend((preyPlot, predPlot), (preyName, predName), 'best')

    plt.xlabel('Years')
    plt.ylabel('Population')
    plt.title('Predator-Prey Model')
    plt.show()
```

Putting Everything Together

```
import matplotlib.pyplot as plt

def populations(…):
    …
def plotPopulations(…):
    …

prey, pred = populations(100,50,.4,.003,.004,.2,10)
times = range(len(prey))
plotPopulations(times,prey,pred,"Hare","Lynx")
```