AGENT BASED PROGRAMMING, LAB 2

PROGRAMMING IN NETLOGO

NAME: _____

INTRODUCTION:

NetLogo is a cross-platform multi-agent programmable modeling environment. NetLogo was NetLogo is free of charge and can be downloaded here:
http://ccl.northwestern.edu/netlogo/download.shtml

We are using Netlogo to introduce the concepts behind computer agents. The complete programming manual for NetLogo can be found here:
http://ccl.northwestern.edu/netlogo/docs/ (alt link)

A shorter easier to user QuickStart Guide can be found here:
http://ccl.northwestern.edu/netlogo/resources/NetLogo-4-0-QuickGuide.pdf (alt link)

In Lab 1 we spent a considerable amount of time typing instructions into the Command Center prompt on the Interface tab. This was so that we could acquire a passing familiarity with the "primitives" (built in variables and functions) used by NetLogo agents. We then grouped sets of instructions under a name into what is called in NetLogo a procedure (function).

This lab has two parts! Each part should take about 20-30 minutes. Feel free to take your time, but you should try to complete both parts.

PART 1: TUTORIAL #3 (20-30 MINUTES)

For the first part of tis lab, your task is to complete **Tutorial # 3** from the NetLogo User's Manual: http://ccl.northwestern.edu/netlogo/docs/tutorial3.html (alt link)

PART 2: CHALLENGE EXERCISES (20-30 MINUTES)

In Tutorial #3 you created a very basic model. In that model your turtles could be considered to be sheep or cows or some other animal that eats grass. However the model you created is very limited and unrealistic. Let's improve that model. Below you will find a series of tasks covering common additions to models. When you are working on your own final project you may want to review this lab so that you understand how to accomplish tasks like adding images and sounds.

1) **LOAD TUTORIAL #3:**

Open NetLogo, goto the "Models Library" and open "Tutorial 3" (it is in "Code Examples").

2) **SELECT OR CREATE A NEW SHAPE FOR YOUR TURTLES:**

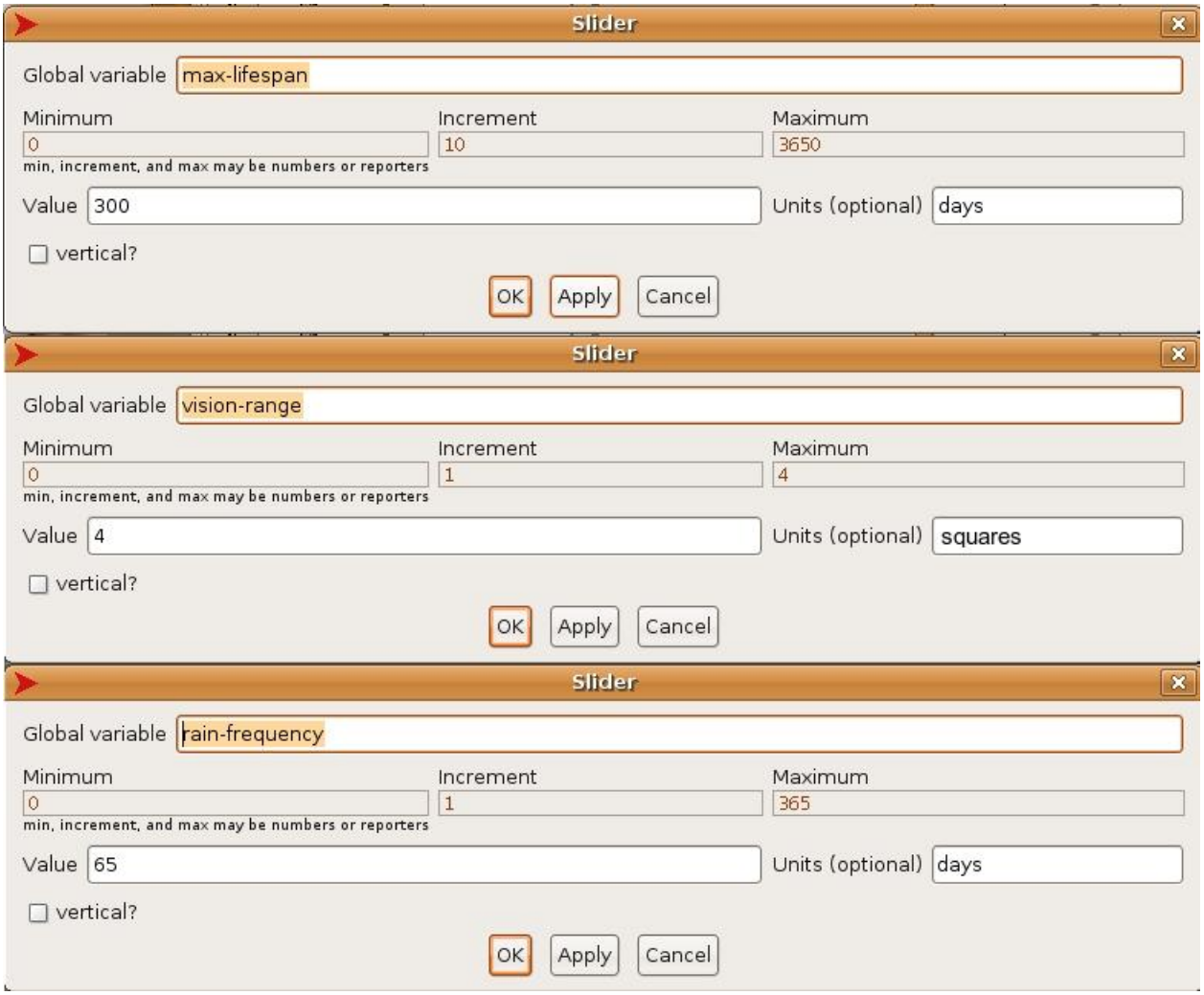
Right now your turtles are multi-color triangles. Find (or design a new shape for your turtles by going to 'Tools' and then "Turtle Shapes Editor'. Make sure and write down the name of the shape that you decide to use. In order to use the shape that you picked you will need to go to the "setup-turtles" procedure and add the following code:

set-default-shape turtles "cow" ;; change cow to the shape name you wrote down

If you want to change the color of your new shapes add this line of code:

ask turtles [set color white]

3) **CREATE THE FOLLOWING 3 SLIDERS IN THE INTERFACE:**



4) **HAVE YOUR TURTLES AGE:**

Right now provided they find enough grass to eat your turtles (or cows, or sheep or whatever) will live forever. That's certainly not very realistic. You will need to change your program in a couple of places so that your turtles can grow old and die.

- Add the variable "age" for turtles.

*turtles-own [energy **age**]*

- Change the setup-turtles procedure so that turtles start with a random age.

ask turtles [set age random max-lifespan]

- Change the move-turtles procedure so turtles age by one day every time that they move.

ask turtles [set age age + 1]

- Change the check-death procedure so turtles die if their age exceed the max-lifespan limit.

ask turtles [if age >= max-lifespan [die]]

- Change the reproduce procedure so that new turtles start off with an age of 0.

hatch 1 [set energy birth-energy set age 0]

5) **HAVE YOUR TURTLES LOOK FOR FOOD:**

At the moment your turtles wander around randomly and if they happen to stumble upon food they eat it. A turtle could easily starve to death, even when there is plenty of grass around. That's certainly not very realistic. Real animals look for food and travel towards it. We will need to change your program in a couple of places so that your turtles can do that.

- Find the following code in the move-turtles procedure.

```
right random 360
forward 1
```

- Change those two lines, to the following 5 lines.

```
let candidates patches in-radius vision-range with [ pcolor = green ]
ifelse any? candidates
[ face one-of candidates ]
```

```
[ rt random 360 ]
forward 1
```

6) **HAVE THE GRASS GROW WHEN IT RAINS.**

At the moment the grass grows randomly after it has been eaten. In many parts of the world the grass depends on seasonal rains, and only grows a few times a year. Change your program so that the "rain-frequency" variable controls how often the grass grows.

- Find the following code in the regrow grass procedure.

```
if random 100 < 3 [ set pcolor green ]
```

- Replace that line with this one:

```
if ticks mod rain-frequency = 0 [ set pcolor green ]
```

7) **ADD SOUND**

Sound is a nice feature to have in a simulation (DON'T OVERDO IT) particularly when you want to give feedback to events (like buttons being clicked) or acknowledge particularly important events in your simulation (like when it is over): link to manual page

- Add the following line of code to the VERY TOP of your program

```
extensions [sound]      ;; adds the ability to use sound to our program
```

- Add the following line to the setup procedure

```
sound:play-drum "Splash Cymbal" 64 ;; play a cymbal crash
```

- Find the following line in the go procedure

```
if ticks >= 500 [ stop ]          ;; stop after 500 ticks
```

- Change the line above to the following

```
if ticks >= 500 [
        sound:play-note "TRUMPET" 60 64 2
        stop
 ]      ;; play trumpet sound and stop after 500 ticks
```

To see a complete version of the lab (with code) click on the link below:

Completed Challenge Exercises - Part 2 Lab