Achoo! Modeling the Spread of the Flu

New Mexico

Supercomputing Challenge

Final Report

April 20th, 2007

Team Number 56

Los Alamos Middle School

Emily TenCate

Sponsoring Teachers: Jessie Gac, Donna Grim

Project Mentor: Deborah Summa

Overview:

A simple computer program was developed to model the spread of the flu in a contained environment such as a school. The StarLogo programming environment was chosen to implement the algorithm. StarLogo allows many agents to be acting independently but at the same time. This feature correlates well with a school environment. The basic model elements were desks and walls (StarLogo "patches") and students (StarLogo "turtles"). Desks and walls were combined to make classrooms, and classrooms were combined to build a school. Classrooms were populated with students, each of whom received a random schedule. A school day consisted of 8 periods, with students changing classrooms randomly each period.

Within a class each student could walk randomly. Student attributes included color (corresponding to health status), chance of sneezing on and infecting a desk, and chances of getting the disease from either a desk or another student. To test whether the model was realistic, two sets of simulations were to be run. Thirty trials with one set of parameters was used to baseline the program, then chances of sneezing, getting infected, and initial number of ill students coming to school were systematically varied. For each set of initial conditions, 15 simulations were run. The output of each simulation was the number of sick people at the end of one day. The same set of conditions was run again, but this time, a command including student-to-student interactions was included.

Simulation results for the simulations run without direct contact between agents showed a wide distribution of potential outcomes, but general trends were apparent. This model was most sensitive to the student susceptibility, and least sensitive to the initial number of

ill students, which was likely due to the absence of student-to-student interactions.

Introduction:

The World Health Organization is warning that a potentially devastating global flu pandemic is imminent. President Bush has requested $7.1 billion to stockpile medications to reduce the impact of such an outbreak. Bioterrorism threats are also of major concern. Computer models are helpful in predicting the likelihood of epidemics and studying their movement within a population. Even simple models can provide insight to help guide public health policies.

The goal of this project was to develop a computer simulation of the spread of the flu in a semi-contained population, such as a school, and to use the model to examine the relative effect of different parameters on the evolution of a small-scale epidemic.

I predict that even with a simple model it should be possible to show a valid relationship between the spread of infection and factors such as susceptibility of a population, the number of times people sneeze, and the initial number of infected students who come to school on any given day.

I think infection rates will climb as susceptibility increases (corresponding to weak, tired individuals such as students who continually stay up too late), or as people openly sneeze more (practicing poor hygiene such as not covering their mouth or washing their hands), and as the number of ill students coming to school increases. I think the latter variable is most important.

I suspect that in most cases the spread of the infection will be self-limiting, but at some point there may be conditions that cause exponential growth or widespread epidemic.

Procedure:

This work had 3 independent phases. The main thrust was to develop the logic, then to write, test, and refine the computer algorithm. A layered approach, where the simplest elements are defined first, then progressively more complex elements are built up, was taken. Special attention was paid to interactions between students and objects and interactions among students.

The second component was to find information on the mode of transmission, incubation period, virulence, etc. of the influenza virus. Assumptions were made about student movement and daily interactions in a typical school, based loosely on observations at my school.

The final phase was to apply the code. A test matrix consisting of different 'experiments' (in which all conditions were held constant except one) was drawn up and simulations run. Intuition, guided by literature data and classroom observations, was used to choose the initial conditions.

Developing the Algorithm:

To construct a Virtual Schoolhouse, I broke the problem into three categories: 1) Physical setting (stationary items); 2) People (mobile); 3) Interactions between stationary and mobile objects; and

4) Interactions between mobile objects.

The basic elements for each category were defined and used as building blocks for more complex things. In the physical setting, first level objects were desks and walls. Four walls enclosed a classroom, which was filled with 25 desks. Nine classrooms made up the school building. The lowest level mobile object was a single student. The next level was a classroom full of students, then all classrooms filled with students. Each student's random "schedule" assigned him or her to a certain classroom. A school day consisted of 8 periods, with students switching classes every 100 clock cycles. Students could interact with their environment and with each other. There was a certain probability of disease transmission whenever an infected student sneezed on a desk or a healthy student came in contact with a germ-covered desk, as well as a probability of transmission when a healthy student made contact with an infected student.

The School Simulation program was written modularly using StarLogo. A screen-shot of the control panel is shown in Figure 1. The first procedure written was "Set-Up". It created the walls for 9 classrooms, then filled each classroom with 25 desks. Next, 224 healthy students (blue) and one ill student (red) were created. The number of initially ill students could be changed by using the Create-Sick procedure. The School-Day procedure assigned each student a schedule, then set the students in motion. It kept track of clock cycles, and told each student to change classes (another procedure) every 100 cycles, until 8 class periods had been completed. Since the flu virus can only live on surfaces for several

hours, a Sanitization procedure was written. This procedure, invoked at the end of each class period, had students clean up a random number of infected desks.

Students moved by selecting a random direction, then going forward 0 or 1 space. If the student was ill (red) AND the patch immediately ahead was blue (a clean desk), THEN the student would check its sneeze probability. If the student did sneeze, germs were spread to 1/4 of the desk in front of him and that piece of desk turned red. IF the student was healthy (cyan) AND the patch immediately ahead was red (infected desk) THEN the student would check its susceptibility chances. If the student was in a weakened state, it became infected (turned red). The same procedure was repeated for disease transmission between students. This cycle was repeated until the school day ended. As the school day progressed, a real-time graph tracked the number of infected students. The complete, annotated code listing is attached in Appendix 1.

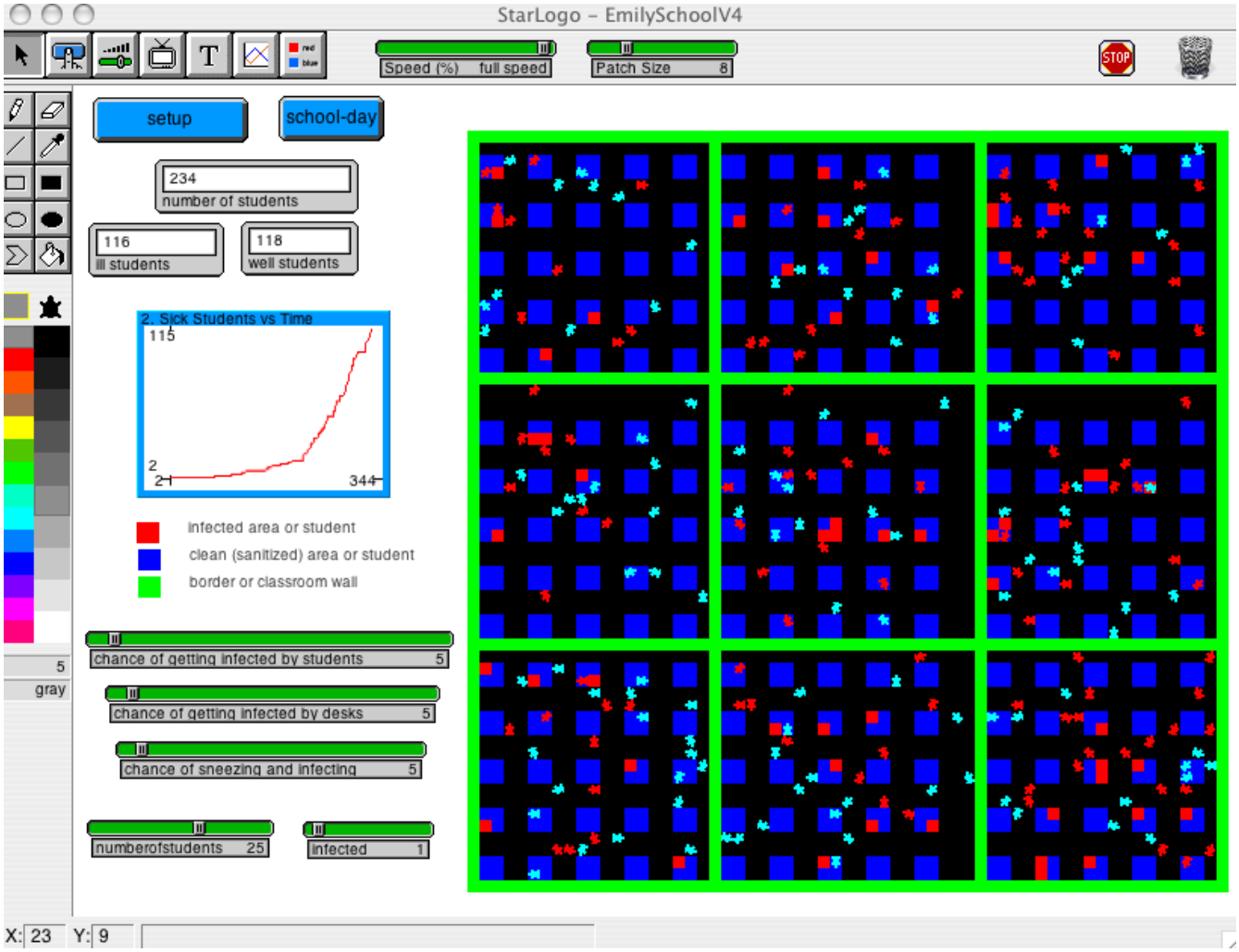

Figure 1: School Simulator Control Panel The virtual school is on the right. The bright green lines are walls, and the square blue patches are desks. The turtle-shaped blobs are students (cyan = healthy, red = infected). On the left are the buttons that invoke the different procedures as well as a counter that tells the number of infected students. The sliders are for changing the initial values of the variables. The plot in the center left graphs the number of sick people as a function of time.

Simulations:

A base set of parameters was selected to test the code with and without the addition of interactions between students. Thirty trials were run, and the total number of infected students at the end of one day was recorded. Three variables were systematically changed and the simulation was rerun 15 times for each new set of initial conditions. The range of parameters tested is shown in Table 1.

Approximately 30 hours of CPU time was required to generate a total of 330 data points. The matrix shown in Table 1 has yet to be run with the inclusion of student-to-student interactions.

Experimentation Matrix:

Sneezing	Susceptibility	# of Initially Ill	
5%	5%	1	(Baseline)
5 - 20%	5%	1	
5%	5 - 25%	1	
5%	5%	1 - 18	

Table 1: Test matrix showing baseline values and range of parameters tested. In each series, only one variable changed; others were held at baseline values.

Results:

A frequency plot of the baseline data for the first simulation is shown in Figure 2. The data is spread out with the peak on the lower end. A range of results is expected because there are probabilities associated with transmitting or catching disease.

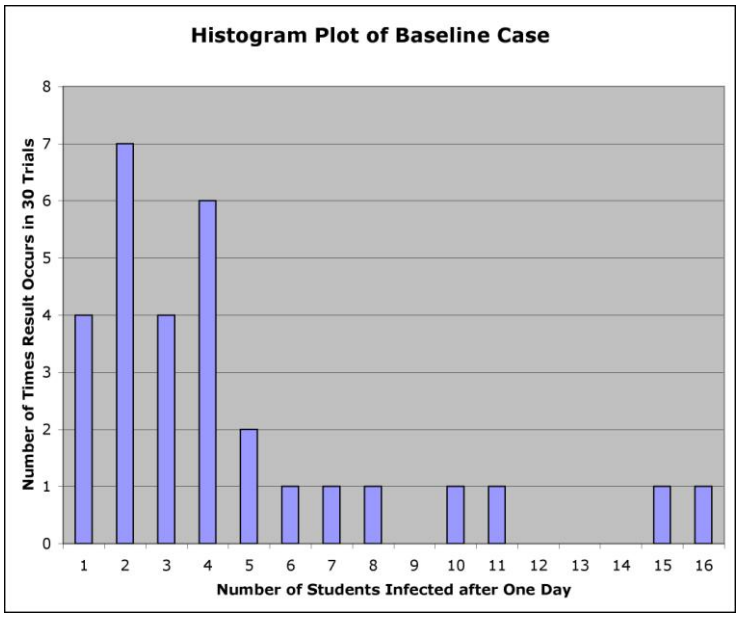

Figure 2: Histogram Plot of Baseline Data (5% Sneeze, 5% Susceptibility, 1 Initially Ill)

The effects of increasing the number of initially ill students arriving at school, increasing the chances of sneezing and spreading germs, and increasing the susceptibility of healthy students are plotted in Figures 3 – 5. The red lines are the minimum and maximum values obtained for each set of initial conditions. The average value is shown in blue. The area between the red lines is the expected response envelope. A discussion of the results is given in the next section.

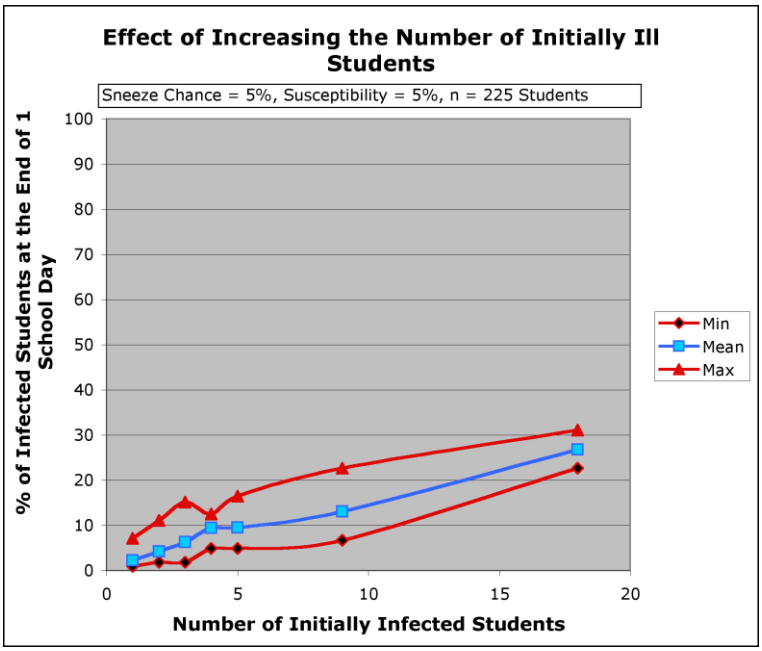

Figure 3: Effect of Increasing the Number of Initially Ill Students (without inclusion of student-to-student interactions)

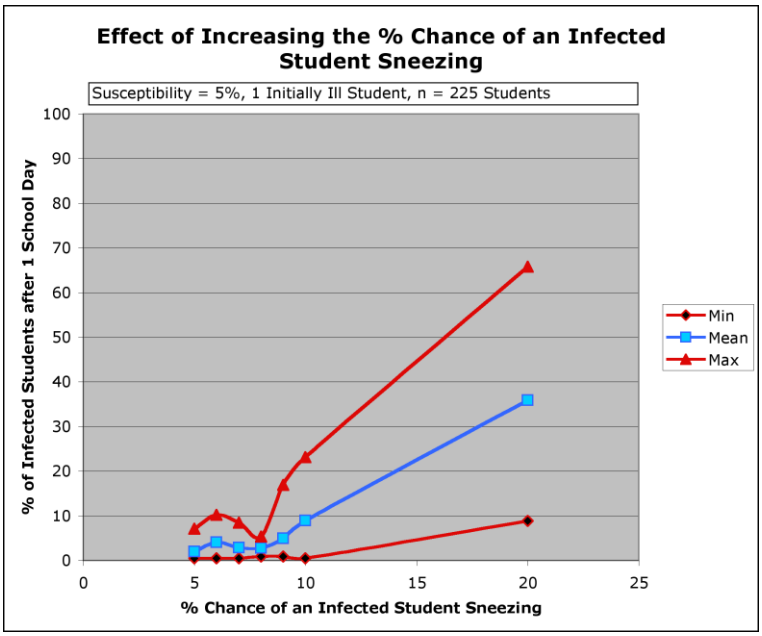

Figure 4: Effect of Increasing the Chance that an Infected Student Sneezes (without inclusion of student-to-student interactions)

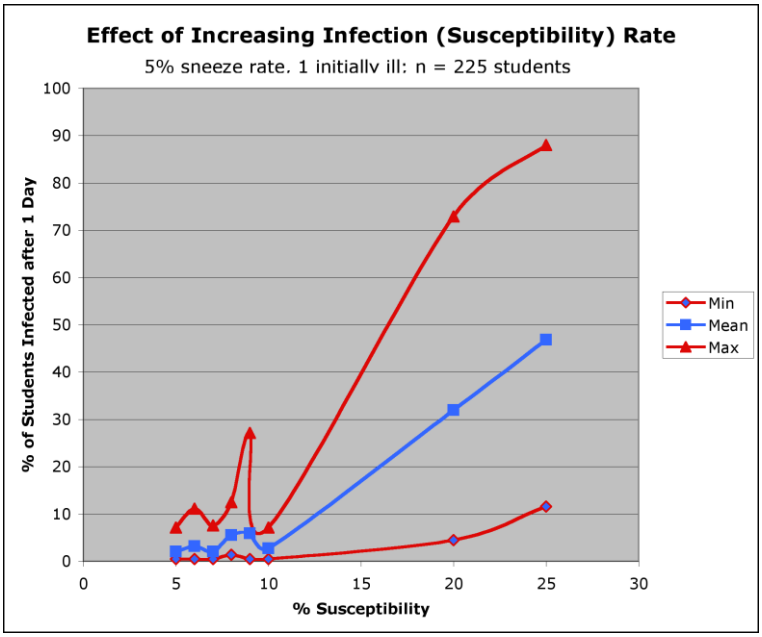

Figure 5: Effect of Increasing the Chance that a Healthy Student Gets Ill (without inclusion of student-to-student interactions)

Discussion and Conclusions:

StarLogo was a good choice of programming environments for this project. It was easy to learn, fairly simple to use, and relatively powerful even if not too robust. Having agents running independently and in parallel was essential for this application.

The layered model worked well. It was easy to build individual modules that incorporated the smaller modules from the previous layer (for instance, schools filled with classrooms filled with desks and walls).

The simulations actually worked, even though they weren't completely realistic. The results were not completely unrealistic either, given the rather broad assumptions. For each set of initial conditions the data did have a wide spread, but general trends were apparent.

This simulation is very sensitive to increasing the chance of an infected person sneezing or increasing the chance of a healthy person getting infected from touching a sneezed-on desk. When the sneezing probability went up to 20%, the infection rate jumped and diseased almost 2/3 of the population. When the susceptibility went up to 25%, the maximum infection rate skyrocketed to 88%.

It would be interesting to include external interactions, such as students going home and encountering people outside of the school, but that would be very difficult to model and code with this program. I would also like to run simulations with a much larger population and test different strategies for containing the outbreak of disease, such as quarantine versus vaccination or examining what percentage of

the population should be vaccinated in order to contain an outbreak.

References:

1. Colella, Vanessa Stevens; Eric Klopfer; and Mitchel Resnick; Adventures in Modeling: Exploring Complex, Dynamic Systems with StarLogo. Teachers College Press, New York, 2001.

2. Resnick, Mitchel; Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds. MIT Press, Cambridge MA, 1997.

3. "StarLogo on the Web." Massachusetts Institute of Technology. Online posting. http://education.mit.edu/starlogo (and associated links).

4. StarLogo User's Group postings. starlogo-users@media.mit.edu

5. "Influenza Fact Sheet". American Lung Association. Online Posting. http://www.lungusa.org/site/pp.asp?c=dvLUK900E&b=35426

6. "Health Matters: Flu". National Institute of Allergy and Infectious Diseases, National Institutes of Health. Online Posting. http://www.niaid.nih.gov/factsheets/flu.htm

7. "Fact Sheet: Key Facts About Influenza and the Influenza Vaccine". Centers for Disease Control and Prevention. Online posting. http://www.cdc.gov/flu/keyfacts.htm (and associated links)

8. "Bush unveils $7.1 billion plan to prepare for flu pandemic". Online posting. http://www.cnn.com/2005/HEALTH/conditions/11/01/us.flu.plan/. Posted November 2, 2005.

9. "Transcript of Bush speech on flu pandemic strategy". Online posting. http://www.cnn.com/2005/HEALTH/conditions/11/01/bush.transcript/index.html. Posted November 2, 2005.

10. "Fact Sheet: Safeguarding America Against Pandemic Influenza" The United States White House, press release. Online posting. http://www.whitehouse.gov/news/releases/2005/11/20051101.html. Posted November 1, 2005.

11. "Responding to the avian influenza pandemic threat: Recommended strategic actions". World Health Organization. Communicable Disease Surveillance and Response, Global Influenza Programme. WHO/CDS/CSR/GIP/2005.8. 2005.

Appendix 1: Code Listing

```
`turtle`
turtles-own [schedule gender]


to school-day     ; this command runs an 8 class period day
setgender random 2
repeat 7 [change-classes
go
clean-up
change-classes]
end


to go     ; this command moves the agents randomly for 100 clock turns, infecting as they go
 repeat 100 [enclose
  wiggle
  enclose
  random-move
  enclose
  sneeze
  enclose
  wiggle
  enclose
  infect
  enclose]
end



to random-move     ; agents move randomly in any direction
```

```
        seth random 360
        enclose
        fd random 2
    end

    to wiggle     ; changes an agents' directions, also sets chance
of agent becoming infected from other agents according to slider
        rt random 10
       grab one-of-turtles-here [if color = red and
      ((random 100) < student_infection) [setc-of partner red]]
        lt random 10
    end

    to enclose     ; contains turtles in lime green classrooms
      if pc-ahead = lime [rt 180 fd random 2]
    end

    to  sneeze     ; sets the chance of infecting a desk according
to slider
    ifelse (color = red) and ((random 100 ) <  chance_of_sneeze)
    [ if pc-ahead = blue [fd 1 stamp red]]
    [fd 1]
    end

    to infect     ; sets the chance of an agent becoming infected
from a desk according to the get-infected slider
    ifelse (color = cyan) and ((random 100) < get_infected)
    [if pc-ahead = red [fd 1 setc red]]
    [fd 1]
    end
```

```
     to box     ; involved in setup, places agents in separate rooms
to create desks

     if color = sky [set xcor -24 set ycor 10 seth 360]

     if color = cyan [set xcor -8 set ycor 10 seth 360]

     if color = turquoise [set xcor 10 set ycor 10 seth 360]

     if color = lime [set xcor -24 set ycor -8 seth 360]

     if color = green [set xcor -8 set ycor -8 seth 360]

     if color = yellow [set xcor 10 set ycor -8 seth 360]

     if color = brown [set xcor -24 set ycor -24 seth 360]

     if color = orange [set xcor -8 set ycor -24 seth 360]

     if color = red [set xcor 10 set ycor -24 seth 360]

     end




     to room     ; moves agents into separate classrooms for desk
creation

     if color = sky [set xcor -30  set ycor 12 seth 360]

     if color = cyan [set xcor -10 set ycor 12 seth 360]

     if color = turquoise [set xcor 12 set ycor 12 seth 360]

     if color = lime [set xcor -30 set ycor -10 seth 360]

     if color = green [set xcor -30 set ycor -30 seth 360]

     if color = yellow [set xcor -10 set ycor -10 seth 360]

     if color = brown [set xcor 12 set ycor -30 seth 360]

     if color = orange [set xcor 12 set ycor -10 seth 360]

     if color = red [set xcor -10 set ycor -30 seth 360]

     end


     to make-desks     ; agents create one desk
```

```
     if color = sky [stamp blue fd 1 stamp blue rt 90 fd 1 stamp blue
rt 90 fd 1 stamp blue]

     if color = cyan [stamp blue fd 1 stamp blue rt 90 fd 1 stamp
blue rt 90 fd 1 stamp blue]

     if color = turquoise [stamp blue fd 1 stamp blue rt 90 fd 1
stamp blue rt 90 fd 1 stamp blue]

     if color = lime [stamp blue fd 1 stamp blue rt 90 fd 1 stamp
blue rt 90 fd 1 stamp blue]

     if color = yellow [stamp blue fd 1 stamp blue rt 90 fd 1 stamp
blue rt 90 fd 1 stamp blue]

     if color = orange [stamp blue fd 1 stamp blue rt 90 fd 1 stamp
blue rt 90 fd 1 stamp blue]

     if color = green [stamp blue fd 1 stamp blue rt 90 fd 1 stamp
blue rt 90 fd 1 stamp blue]

     if color = red [stamp blue fd 1 stamp blue rt 90 fd 1 stamp blue
rt 90 fd 1 stamp blue]

     if color = brown [stamp blue fd 1 stamp blue rt 90 fd 1 stamp
blue rt 90 fd 1 stamp blue]

     end


     to make-desk-row     ; each agent creates a row of five desks

     repeat 5 [make-desks

     seth 270

     fd 1

     seth 0

     fd 4]

     bk 2

     end


     to fill-room     ; each agent creates an five by five array of
desks

     make-desk-row

     room
```

```
     seth 90
     fd 4
     seth 0
     make-desk-row
     room
     seth 90
     fd 8
     seth 0
     make-desk-row
     room
     seth 90
     fd 12
     seth 0
     make-desk-row
     room
     seth 90
     fd 16
     seth 0
     make-desk-row
     room
     end


     to change-classes     ; this command relocates each student to a
new classroom
     setschedule random 9
     if schedule = 0 [set xcor -22 set ycor 21]
     if schedule = 1 [set xcor -2 set ycor 21]
     if schedule = 2 [set xcor 20 set ycor 21]
     if schedule = 3 [set xcor -22 set ycor -1]
```

```
     if schedule = 4 [set xcor -2 set  ycor -1]

     if schedule = 5 [set  xcor 20 set ycor -1]

     if schedule = 6 [set xcor -22 set ycor -21]

     if schedule = 7 [set xcor -2 set ycor -21]

     if schedule = 8 [set xcor 20 set ycor -21]

     end


     to clean-up     ; this procedure simulates the fact that germs
don't usually last outside of the body for too long

     repeat 100 [random-move if pc-ahead = red [fd 1 stamp blue]]

     end




     `observer`

     patches-own [germs]


     to crt-sick

     crt infected

     ask-turtles [ifelse (color = red) or (color = cyan)[fd 1 bk
1][setc red]]

     end


     to setup

     ca

     crt 1

     ask-turtles [set xcor -31

     set ycor -31

     setc lime

     pd

     fd 63
```

```
die]
crt 1
ask-turtles [set xcor -31
set ycor -31
setc lime
rt 90
pd
fd 63
die]
crt 1
ask-turtles [set xcor 31
set ycor -31
setc lime
pd
fd 63
die]
crt 1
ask-turtles [set xcor 31
set ycor 31
lt 90
setc lime
pd
fd 63
die]
crt 1
ask-turtles [ set xcor 31
set ycor 11
seth 270
```

```
setc lime
pd
fd 63
die]
crt 1
ask-turtles [set xcor 31
set ycor -11
seth 270
setc lime
pd
fd 63
die]
crt 1
ask-turtles [set xcor -11
set ycor 31
seth 180
setc lime
pd
fd 63
die]
crt 1
ask-turtles [set xcor 11
set ycor 31
seth 180
setc lime
pd
fd 63
die]
crt 9
```

```
ask-turtles [room]
ask-turtles [fill-room]
ask-turtles [ifelse color = red [fd 1] [setc cyan]]
ask-turtles [repeat numberofstudents [hatch [setc cyan]]]
ask-turtles [change-classes]
clearplots
end
```