9-26 – 10-5 – Introduction to NetLogo

* GUI
  + Anything other than plain text
  + Includes visual indicators, buttons, etc.
  + Convenient for experts but needed to support usability
* Commands
  + A short piece of code
  + When we type a command, the program calls (executes) the corresponding piece of code
* User-defined Procedures
  + Go, setup, etc.
  + Commands created by the user
* Comments
  + Start with ;;
  + Illustrates the user’s thought process for others
  + Code is written for people, not just machines
  + Good comments
    - Summarize a large piece of code
    - Translate specific bits in human language
    - Explain all the decisions you made
  + Bad comments
    - Are misleading
    - Repeat the code
    - How explains how, not why
    - Makes little sense for anyone but the person who wrote them
* All models have two main parts
  + 1. Initialization (setup), assigns the baseline value of all entities
  + 2. Computes simulation results (go), updates entities values
* Topology
  + Formed by the set of cells and their connections
  + Depends on whether your neighbors with cells sharing a side or corner
  + The concept of neighbors is flexible as you can decide on the relevant distance
  + Periodic boundaries – cells wrap up horizontally and vertically.
    - It looks like a grid but it’s actually a torus
  + The direct neighbors for cells can be based on
    - Closed or periodic boundaries
    - Whether cells share sides only or sides and corners
    - The shape of cells
    - The distance
  + The topology has a strong influence on the model’s outcome
    - By creating more contacts, thus increasing potential for a spread
    - By stopping dynamics at the boundaries, or mimicking 3D
* States
  + Normally looked at differently over three steps
    - Describe states
    - Encodes states
    - Visualize states
    - NetLogo allows you to bypass this
  + If we want to follow a proper process
    - Tell NetLogo that cells have states
    - Assign these states
    - Map states to colors
* How does the computer store states?
  + Everything is stored as 0 and 1
  + Transistors
    - Two values: current on, current off
* How do you encode numbers using only 0s and 1s?
  + We only have two signs; their position stands for a power of 2
  + 0-1, 1-2, 2-4, 3-8, 4-16, 5-32, 6-64, 7-128, 8-256, 9-512, 10-1024
  + If you are given a decimal number
    - Subtract the largest powers of 2
    - OR
    - Repeatedly divide the number by 2
* How do we store binary?
  + Computer storage is abstracted as linear. The storage is always 1D
  + To map the linear storage organization to actual data, you need to know how many rows and columns you have
  + First two bits tell you your columns and rows and then the numbers follow