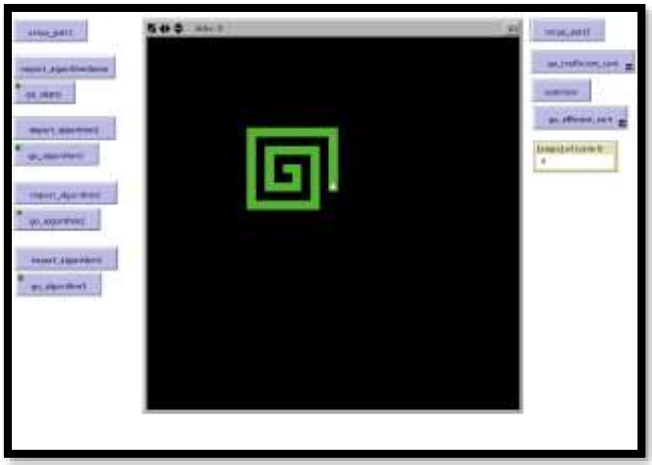
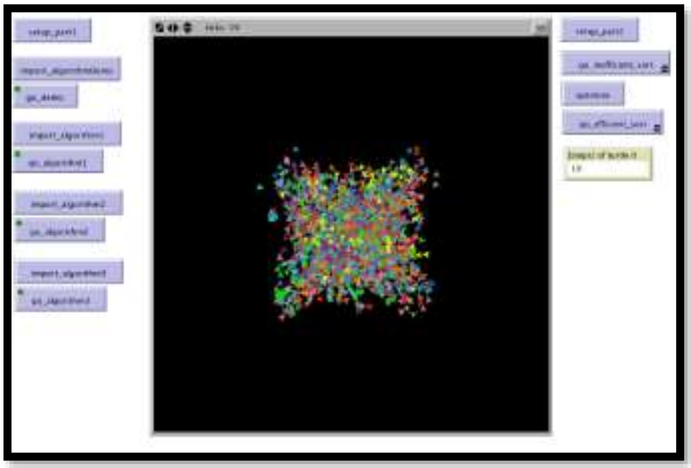
CS108L Computer Science for All
Module 7: Algorithms





Part 1 Patch Destroyer Model Overview:

For part 1, you will need to get your turtle to destroy the green patches as efficiently as possible. You will download the base model and some image files. The image files create different patterns in green patches in the model. You need to create procedures that move your turtle around the NetLogo world **one patch at a time** destroying the green patches (turning them black) as it goes. The object of the assignment is to do so in the fewest steps possible, while still following the rules listed below.

NOTE: You must download the images and base program from the link on the class website. The images must be in the same folder as your program for it to work.

Setup:

1) The turtle must start at the center of the NetLogo world and be facing up and be a fixed color that is NOT green or black**.**
2) Your turtle can only move 1 patch at a time. You cannot use a different command (such as setxy) to move the turtles.
3) You must count the total number of steps that your turtle takes to munch the patches.
4) You must create **a monitor** to report the number of steps your turtle takes.
5) You must create a **go procedure** for each of the imported images you solve:
    - **a. Each solution must include a repeat or while loop to make it more efficient.**
    - **b.** Minimize the steps your turtle takes to munch the patches!

Part 2 ColorSort Model Overview:

In Part 2, we will sort turtles 2 ways: efficiently and inefficiently.

Sort inefficiently: On each tick, we ask turtles to move towards another turtle that has the same color. So every time through, we are creating a set of turtles with the same color. (This is a forever button)

Next, we'll optimize how they sort. Let's make that set of turtles with the same color *just once* in our setup. Then we don't have to do it every time!

Last, we'll sort efficiently, using the set of turtles we created at the start. (This is a forever button)

Do the following first:

1) Turn on world-wrapping if you haven't already.
2) You should already have a turtles-own variable from part 1. Now, add one more variables in the turtles-own section:
   - create a variable that will store a list of other turtles with the same color, like **turtlesWithMyColor.**

Setup:

Make a setup_part2 button and procedure that clears the world, resets ticks, creates 2000 turtles, and sets those turtles to random locations.

Inefficient Method Procedure:

The algorithm for the inefficient method is:

a) Each turtle has a color. You should create a **local** variable to keep track of that color for you.
b) Each turtle looks at the other 1999 turtles one at a time to see what color it is. If the other turtle has the same color then it is put into a NetLogo **agentset** (a list or set of agents). An **agentset** lets you keep track of a group of agents so you can use that group later.
c) The turtle then randomly chooses one of the other turtles of the same color (in the created agentset) and names it (e.g. **myTarget**).
d) The turtle turns to face the turtle chosen in step (c) (Hint: set heading towards **myTarget**)
e) The turtle takes one patch size step forward.

How to do Inefficient Method Steps b and c:

The simplest way to do steps b and c of the inefficient method is to chain a few of NetLogo's reporters together with the following commands:

- **one-of** *agentset*
- **other** *agentset*
- *agentset* **with[***reporter***]**

Efficient Method Procedure:

First, we need to build an agent list for every turtle that contains the turtle’s agentset. Then we can pick a random member from the agentset and:

- Store turtle’s color in a local variable.
- Pick a target from your created agent list.
- The turtle turns to face the chosen target.
- The turtle takes one patch size step forward.

Module 7: Algorithms Grading Rubric (20 Points Total)		
Done	Points	Task

	1	A:
		• Submit a NetLogo source code with the file name: M1.*firstname*.*lastname*.nlogo.
		• The first few lines of your Code tab are comments including the following:
		`;Student’s Name:`
		`;School:`
		`;Teacher's Name:`
		`;Date:`

	2	B: • The code in the Code tab of your program is appropriately documented with “in-line” comments.
		PART 1
	3	C: • The Info tab of your code contains a description of your program. See Coding Standards Guideline for more information.
	2	D: • Your turtle destroys all the green patches on the import_image_one.
	2	E: • Your turtle destroys all the green patches on the import_image_two.
	2	F: • Your turtle destroys all the green patches on the import_image_three.

	2	G: • Your monitor successfully reports how many steps your turtle takes.
		PART 2
	2	H: • Your Inefficient Sort sorts the turtles.
	2	I: • Your Optimize procedure creates the agentset of turtles with the same color.
	2	J: • Your Efficient Sort uses the agentset to speed up the sorting.
	1	(Extra Credit) • Following the structure of the other images, create a procedure and button to import_ec_algorithm.png. Create a procedure called go_ec_algorithm that destroys all the green patches of ec_algorithm.png when its button is clicked.