Module 2B
Building a NetLogo Model

We are going to build a simple model where a butterfly visits flowers. After the flowers have been visited, they will change color.

Open NetLogo and click on the Code tab. Our model will have two agents: flowers and insects. In NetLogo these are known as breeds. If we define them at the beginning of the program, we can use these names later in the program. We also need a property which will keep track of whether a flower has been visited. We will call this attribute pollen and it will be something that the flowers own.

Definitions

Type the following lines at the top of the procedures page:

```
breed [flowers flower]
breed [insect]

flowers-own [pollen]
```

Setup

Now we can create the setup program. First we type clear-all to clear the world of previous runs. Then we create 50 flowers. There is a default flower shape in NetLogo. So after we create 50 flowers, we have to tell the agents the shape, color and size. We also will place them randomly in the world and give their pollen attribute a value. For the purpose of this model, when the pollen is set to 1, the flower will be yellow.

We will also create one insect, for which we define the shape, color, size and position. In this case, we will have the butterfly start in the center of the world. The word end closes this part of the program.

```
to setup
    clear-all
    create-flowers 50
        [set shape "flower"
            set color yellow
            set size 2
            setxy random-xcor random-ycor
            set pollen 1]
    create-insect 1
        [set shape "butterfly"
            set color blue
            set size 2
            setxy 0 0]
end
```
When you have finished typing this section, click on the Check button to make sure that the code is acceptable. This step only checks to make sure there are no “grammar” errors in the code.

Before adding more code, we will check this portion to make sure it is doing what we think it is. Click on the Interface tab. Once at the Interface, click on the blue button next to the word Add and then click anywhere in the white space. A button should appear and a dialog box open. Type the word setup in the box and click OK. The button should now have the word setup in it. Click on the setup button just created. Fifty yellow flowers should appear randomly distributed around the world and a blue butterfly should appear in the center.

The Heart of the Model

The center of a NetLogo model is usually the go portion. This is where the agents get their instructions, so the go procedure usually starts with the ask command, since we will be asking the agents to do something. We will be asking our butterfly to move around and we will be asking the flowers that are visited to change color when pollinated. It is best to handle each step in its own section which we can name. So we can stop and check, first we will make the insect move.

In the “go” section, we tell the insect to move around and to pollinate flowers. The brackets define everything that we are asking the insect to do. Let’s skip a space after the setup section and type the following:

```netlogo
to go
  ask insect [move]
end
```

Next we have to define what move means. We want to have the insect randomly pick a direction each time before it moves (set heading random 360). Then we want it to move forward one space. Skip a space after the go section and type the following:

```netlogo
to move
  set heading random 360
  forward 1
end
```

Let’s check what we have done so far. First use the check button at the top of the page. Then go to the interface page and create a go button just like you created a setup button, only this time type the word go in the dialog box. What happened?

You should have noticed that the insect only moves once. To have the insect continue moving, right-click on the go button and choose edit. Note that at the top of the dialog box, there is a check-box with the word Forever next to it. Check this box and click OK. You will notice that the appearance of the button has changed. Now there is double arrow. Now when you click the button, it stays “depressed” and the action continues. Now to stop the insect, you must click the go button again.

You should also have noticed that the insect is moving very quickly. One way to slow it down is to use the speed slider. We can also build a wait command in the program. To the end of the move section add wait 0.1.
to move
  set heading random 360
  forward 1
  wait 0.1
end

Return to the interface check the effects of this command.

Now we want to have the flowers change color when the butterfly has actually visited them. This will allow us to visually mark the progress the butterfly. It will also make it easier to count the visited flowers later.

First, return to the go section and insert the word pollinate on a line after move.

to go
  ask insect [move pollinate]
end

Then we define what pollinate means. First the insect checks to see if there are any flowers at its position. Then it asks one of those flowers if it still has pollen (pollen is equal to 1). If that is true, then we ask the flower to change its pollen to 0, we again have the program wait and then we ask the flower to change color. Type the following lines:

to pollinate
  if (any? flowers-here) [ask one-of flowers-here [if pollen = 1 [set pollen 0 wait 0.5 set color orange]]]
end

After checking the code, go to the interface and see how the model has changed.

Adding input and output devices

So far our model is a nice image, but the user doesn’t have any input into the initial conditions and the user must count the orange flowers to see how the butterfly is doing. Let’s add some input and output devices for the user.

Sliders
First we will give the user some control over the number of flowers in the world. We will need to create a slider in the interface and add some code in the procedure. Until both steps are finished, don’t be surprised if there are error messages.

Start in the interface, click on the drop down menu that you used to create the buttons and select slider. Then place the slider somewhere in the white space under the go button. As with the buttons, a dialog box opens. Type
number_of_flowers

The name of this slider will become a global variable and cannot have spaces in its name. Also make sure that the slider is set so that the minimum is 1, the maximum is 100, the increment is 1, and the value is 50.

Now go to the procedures section. In the setup section, replace create-flowers 50 with

```netlogo
create-flowers number_of_flowers
```

Now instead of 50, the number of flowers created will equal the value on the slider. Return to the interface page and try out the change in the model.

Plots

Finally, we will add a plot. This will also require changes to the interface page and the code page. Let’s start with the code page. In NetLogo v.5, plot code has been simplified. We just need to add two lines. At the end of the setup section, add “reset-ticks”.

```netlogo
to setup
clear-all
create-flowers 50
set shape "flower"
set color yellow
set size 2
setxy random-xcor random-ycor
set pollen 1
create-insect 1
set shape "butterfly"
set color blue
set size 2
setxy 0 0
reset-ticks
end
```

Then in the “go” section, add the command “tick” at the end of that procedure.

```netlogo
to go
ask insect [move pollinate]
tick
end
```

Click on the Check button to make sure the code is acceptable and then go to the interface page.

Here click on the drop-down menu next to Add and choose plot. Click on some open white space to create the plot. Again, a dialog box will open. For the name, type Pollination. Then
in the line next to Plot Pens, rename the default pen to **Pollinated flowers**. Change the color of the pen to orange. Then in the space below Pen-update-commands type

```plaintext
plot count flowers with [color = orange]
```

The dialog box should look like this:

![Plot dialog box](image)

Click OK. Run the model again to see the plot.