

**INTRODUCE:**

Start by asking the class if anyone has heard of robotics. **What is a robot? Does a robot really “understand” what people say?** The answer to the last question is:







***“Not the same way that a person does.”***

Robots operate off of “instructions,” specific sets of things that they have been preprogrammed to do. In order to accomplish a task, a robot needs to have a series of instructions (sometimes called an algorithm) that it can run.

To get more familiar with the concept of an algorithm, it is helpful to have something to compare it to. For this exercise, we will introduce a programming language made of lines and arrows.

**PROGRAMMING KEY**


---

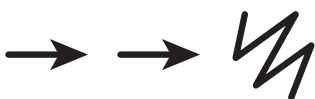
	—	Move One Square Forward
	—	Move One Square Backward
	—	Move One Square Up
	—	Move One Square Down
	—	Change to Next Color
	—	Fill-In Square with Color

---

In this instance, the symbols on the left are the “program” and the words on the right are the “algorithm” piece. This means that we could write the algorithm:

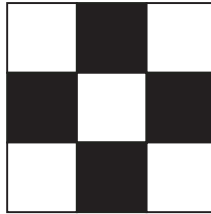
***“Move one square forward, Move one square forward, Fill-in square with color”***

and that would correspond to the program:



Now it's time to get a little more practice. Start your class off in the world of programming by drawing or projecting the provided key onto the board.

Select a simple drawing, such as this one to use as an example.

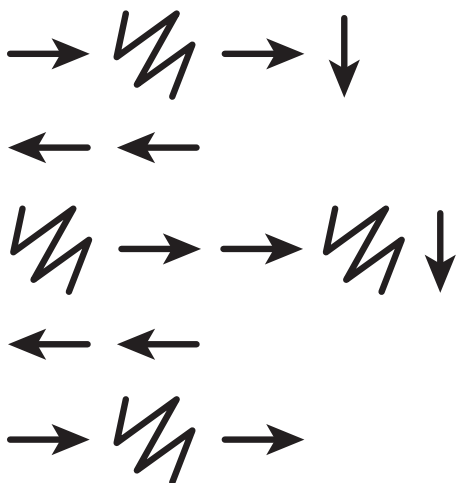


This is a good way to introduce all of the symbols in the key. To begin, you can show them the least confusing way to code an image, which is to return to the left of the image whenever you drop to the next line. Fill in the graph for the class, then ask them to help describe what you've just done. First, you can put it into words with an algorithm, then you can program what you've laid out.

**A sample algorithm:**

**“step forward, fill-in, step forward, next row,  
back, back,  
fill-in, step forward, step forward, fill-in, next row,  
back, back,  
step forward, fill-in, step forward”**

Some of your class may notice that there are some unnecessary steps, but hold them off until after the programming stage. Walk the class through programming the image:

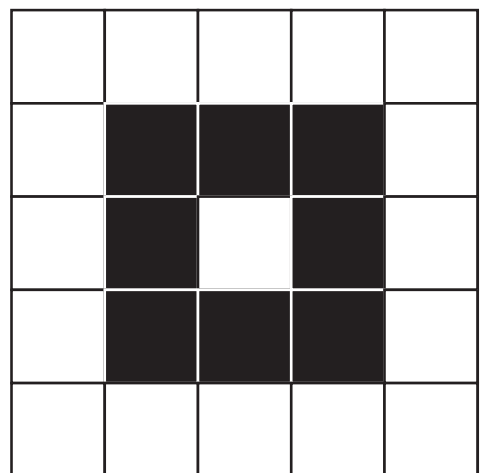
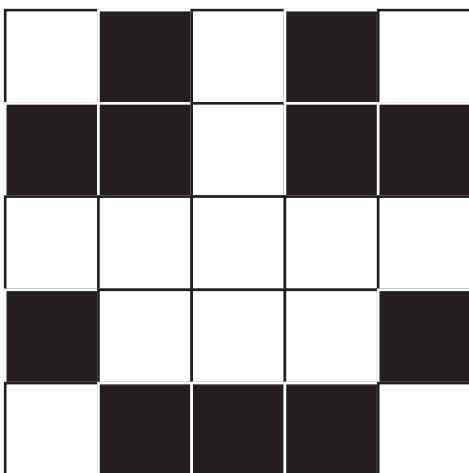
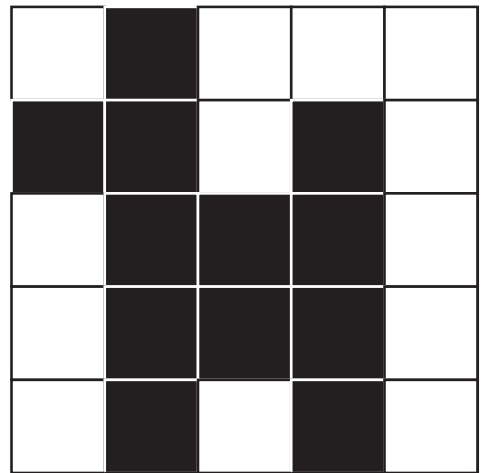
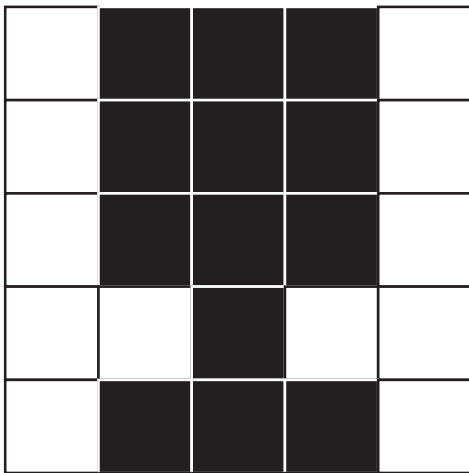
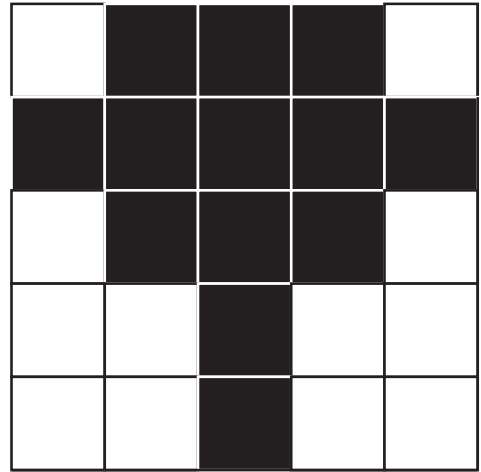
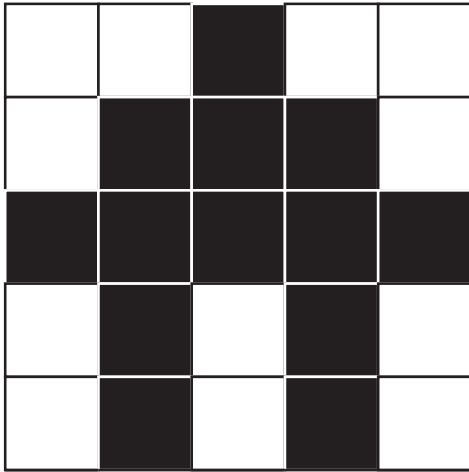


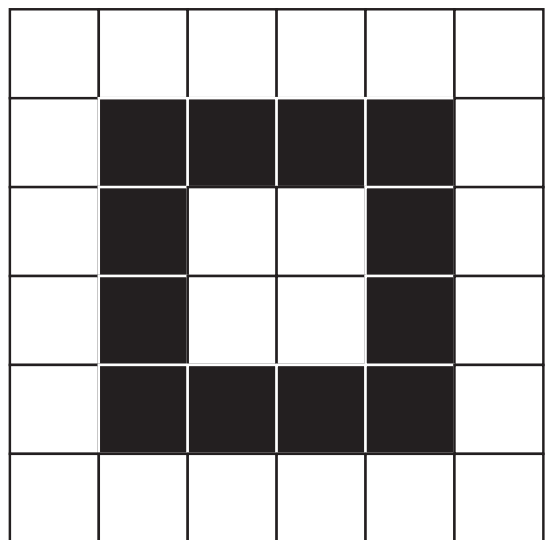
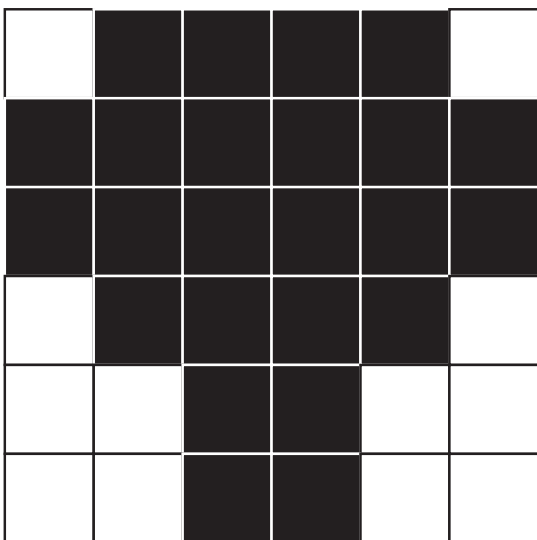
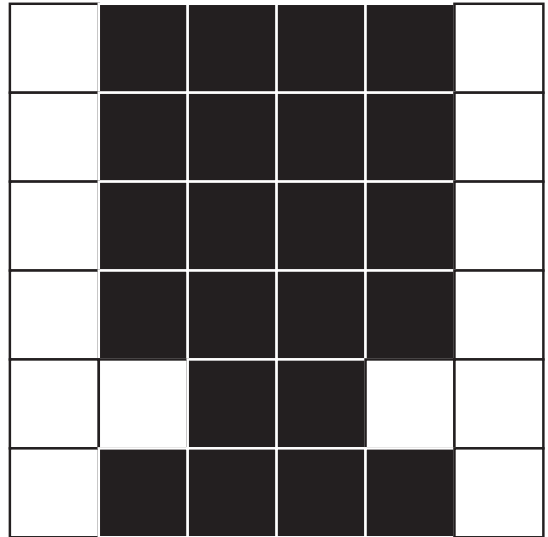
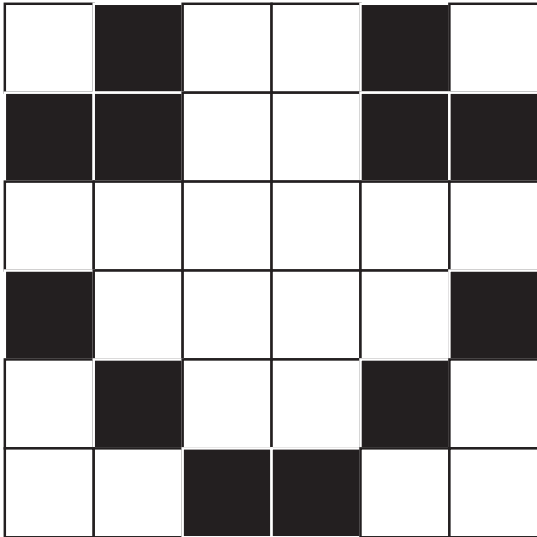
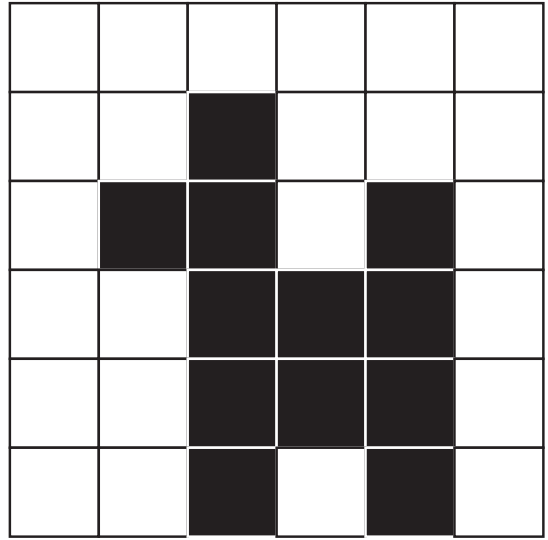
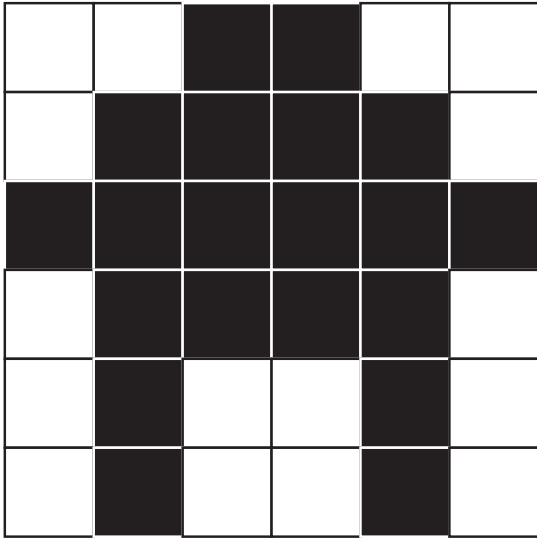
The classroom may be buzzing with suggestions by this point. If the class gets the point of the exercise, this may be a good place to entertain those.

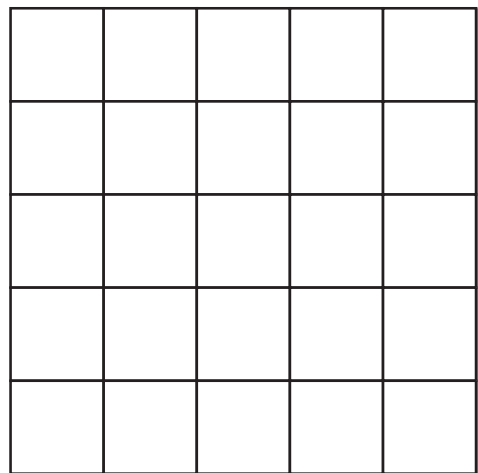
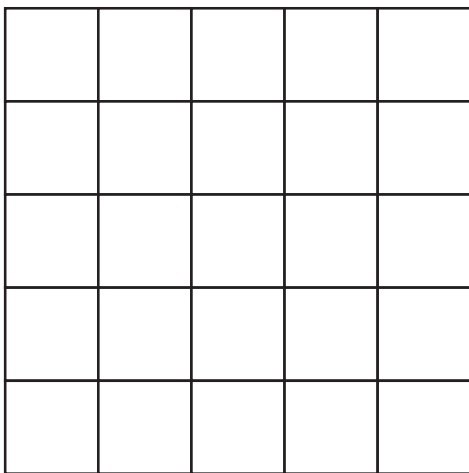
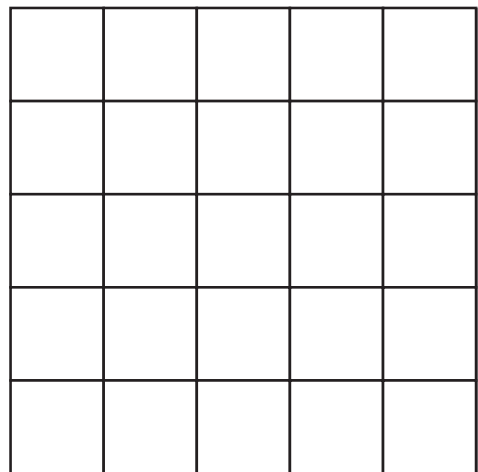
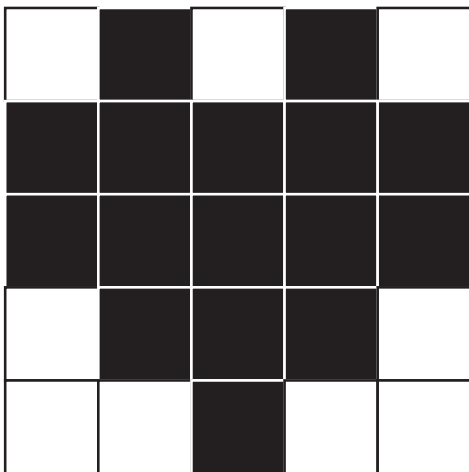
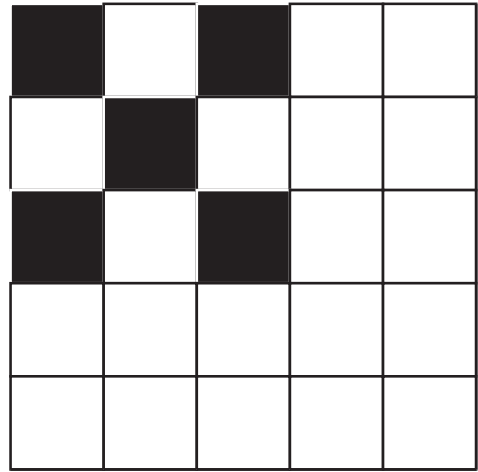
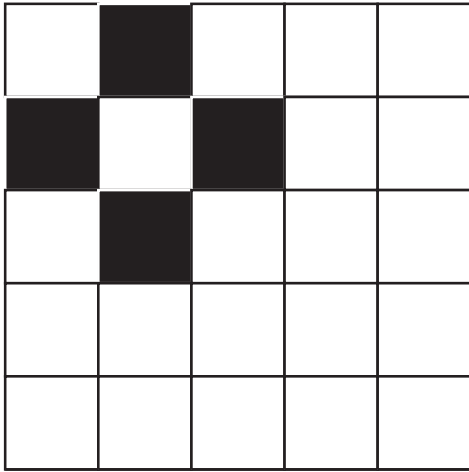
It is true that there is a bit of redundancy above, but it can be extremely confusing to code without it. Sometimes, until a programmer is experienced, it is helpful to code in the most understandable way to make sure it works, then remove unneeded steps later.

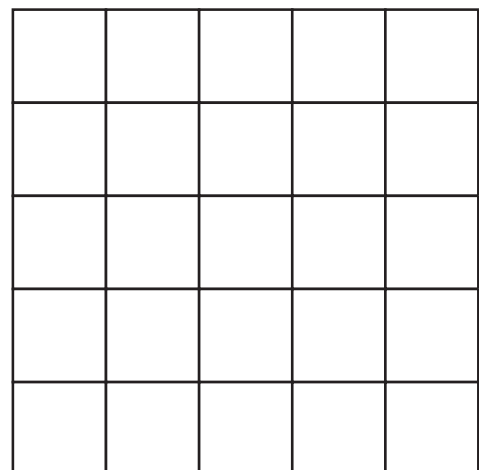
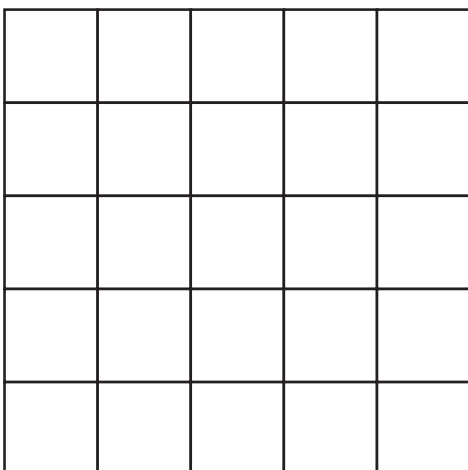
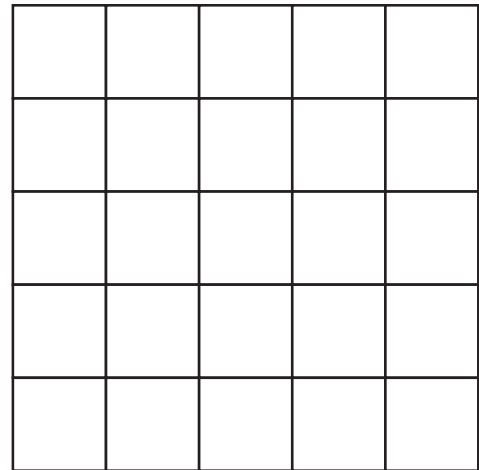
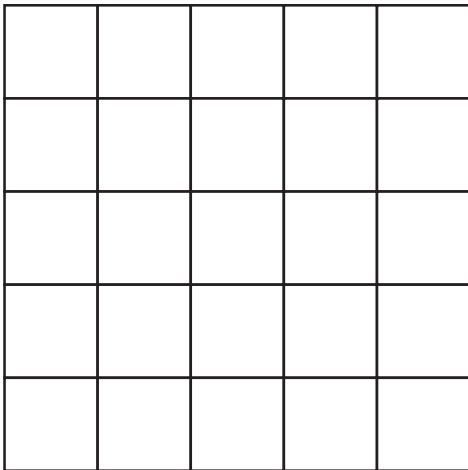
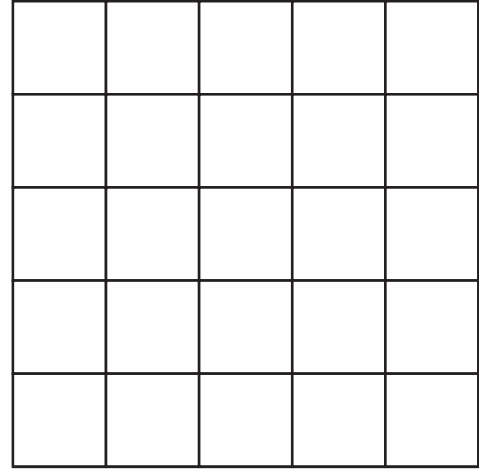
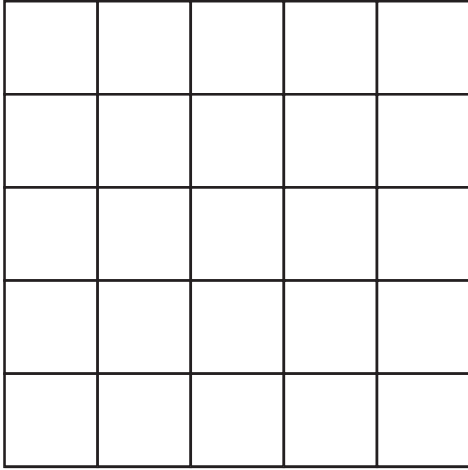
**STEPS:**

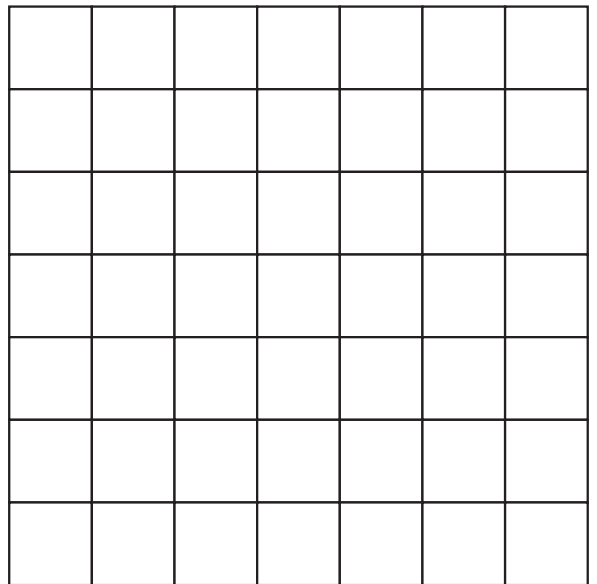
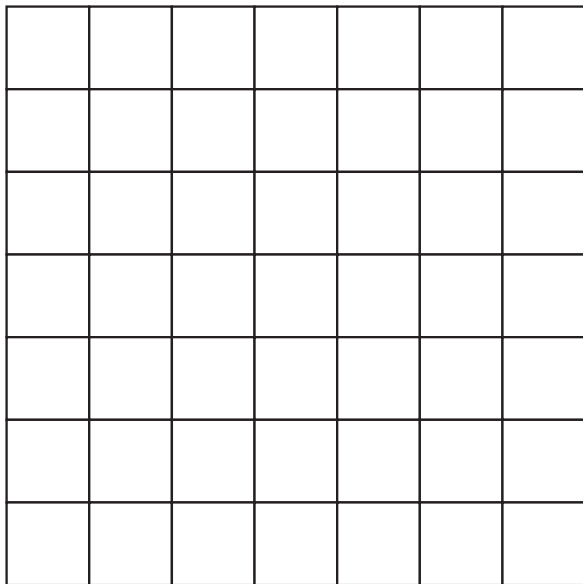
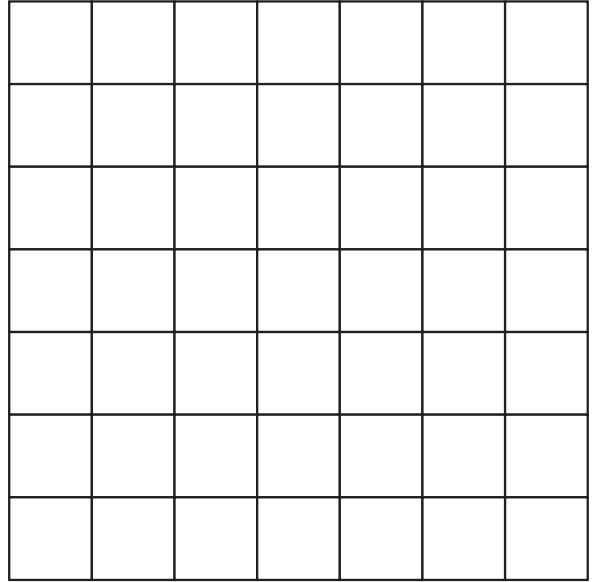
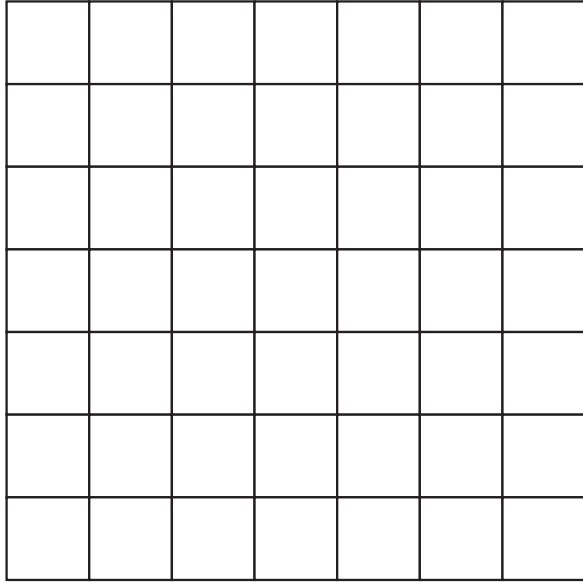
- 1) Choose image from image pack.
- 2) Write out algorithm to draw that image.
- 3) Convert algorithm into a program using symbols.
- 4) Trade programs with another team and draw their image.
- 5) Add “functions” to make programs more simple.
- 6) Write programs for more complex images.
- 7) Trade your complex programs and draw again.



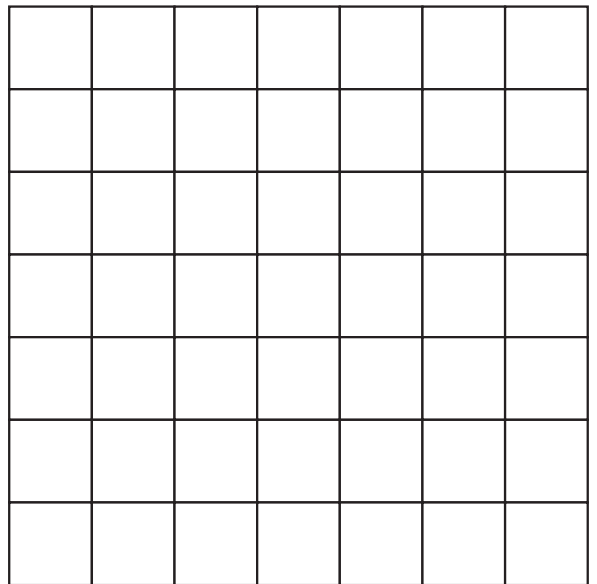
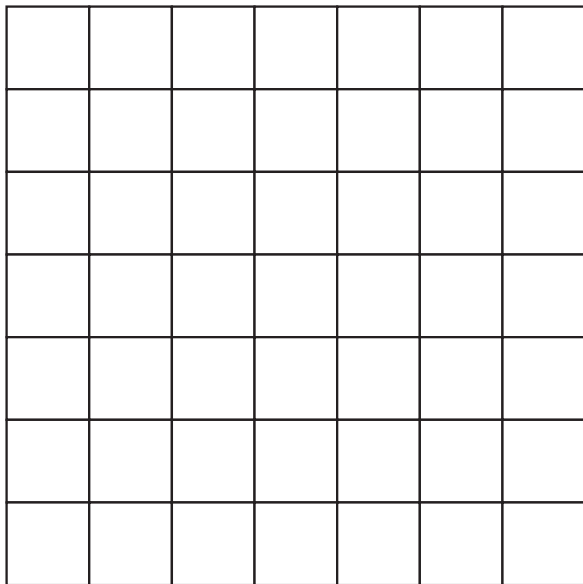
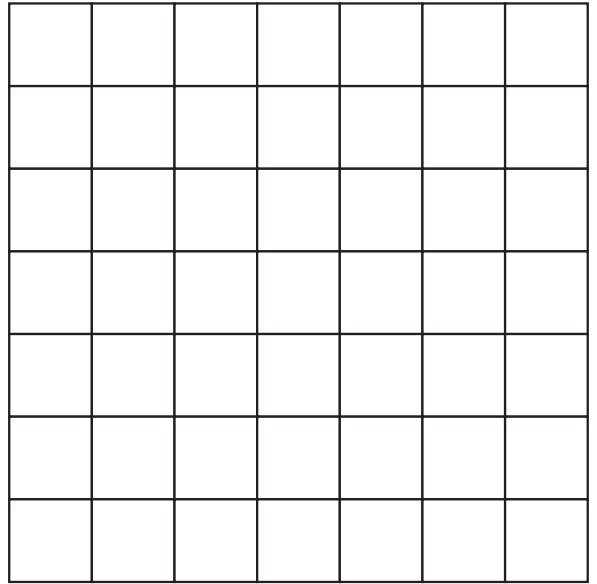
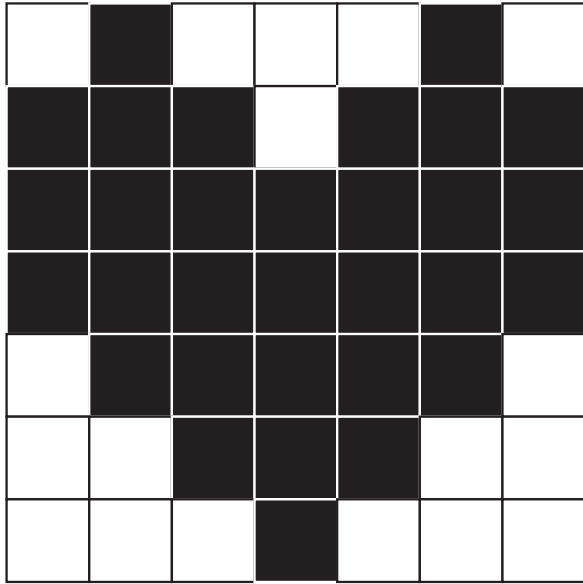












4 LESSON 4: GRAPH PAPER PROGRAMMING

 Color 1

 Color 2

