RCX Code

You've been pretending to be "stupid robots" for a while now. Do you think you're ready to program some real ones?

RCX Code is the programming language you'll use to tell your robot what to do. In fact, RCX stands for Robot Command Center. You are going to write programs using RCX Code, and then we will show you how to download those programs to your robot's big yellow brain.

Remember, the more specific your instructions, the better your chances are of getting the robot to do exactly what you want. These robots really are stupid: they won't be able to do anything unless you spell it out for them.

So how do you tell a robot what to do?

You put your robot together by joining individual blocks. You'll put your program together in the same way.

We'll show you how to get to the main programming screen on the Lego Mindstorms software. Once you're there, you'll notice a row of colorful blocks on the left side of the screen. Those are the blocks you'll be using to create your programs.

The first block is green, and it's labelled Commands! If you click on it, it will open up and give you a list of command blocks that you can use to control your robot's motors.

The commands we'll be using today are:

On: turns your motors on.
On for: turns 'em on for as many seconds as you want.
Off: turns your motors off.
Set power: How much power (speed) do you want to give to the motor?
Set direction: Which way do you want the motors to turn?
Reverse direction: self-explanatory.

Wait: Makes your robots stand in place for a given number of seconds.
Beep: You know you want to beep.
Tone: Makes the robot produce a musical note.

The next big block under Commands is blue, and it's called Sensor Watchers. The blocks in this list let you tell the robot what to do when one of its sensors is activated. We'll only be using the light and touch (aka "bump") sensors for now.

The next block is red and it's labelled Stack Controllers. First of all, what's a stack? You're going to write your programs by basically stacking the blocks you want one below the other. So a stack refers to a series of instructions, one train of thought for your robot to follow. What's a stack controller? It gives more power to your language, and lets you have better control over what your robot does. You've used some of these before:

Repeat: repeats the blocks inside it for as many times as you want
Repeat forever: just like it sounds!
Repeat while: repeats while a condition is true or false. These conditions will all have to do with your

sensors (for example, “repeat while the light sensor is getting a high reading”)

Wait until: Again, just like it sounds. For example, “wait until the bump sensor goes off”.

Those are the basic blocks we’ll be using today. In the next few days, your programs will get more advanced, and we’ll be exploring some other blocks, especially the yellow one called My Commands.

Here are some simple assignments for today. The first ones are just practice in getting the motors to do what you want. The last few involve a combination of motor and bump sensor programming.

1. Can you make your robot move forward for 1 second, wait for half a second, and come back to you?

2. Now, can you make it go forward (pick a number of seconds), wait for half a second, *turn around twice*, and come right back to you?

3. Your robot’s just a stupid computer. But maybe you can teach it to move around in a rectange (that is, move forward some number of seconds, turn right 90 degrees, move forward some number of seconds,

turn right 90 degrees.... until the robot has traced out a square or a rectangle.)

4. Bump sensors! Make your robot move forward until it's bumper gets hit. Then it should move back for half a second.

5. For this task, your robot should move forward straight until its touch sensor gets hit. Then it should keep moving forward, but in the opposite direction of the sensor that got hit (for example, if you hit the sensor on the left side, the robot should veer to the right, and vice versa.)

How many of these tasks can you get done?