Technical Learning

Divide your team into groups of builders, programmers, and possibly testers. The builders will build 2 of the Constructopedia robots and the programmers will write a program to drive the robots in a square. If you have testers, they will be responsible for running the programs and making observations about the performance of the robots. However, everyone should get a chance to observe the 2 robots move around while executing the program. They can then see how the exact same program will run differently depending on the engineering design and the environment. (If you only have one RCX programmable brick, then one robot will have to be built to completion, tested, then taken apart slightly to use the RCX brick for the other robot.)

You'll have to figure out how to divide up the groups appropriately. You probably want everyone to get an opportunity to try programming and building. However, the building (especially if a group is only to make 1 robot) will probably go much faster than the programming. Therefore, you might have to come up with other activities for the builders to work on while the programmers are finishing up. If you have a testing group, then they will have to wait until both groups are finished in order to start their task.

Coaches in the past have found that assigning specific roles to each group member works best. For the builders, one can find the pieces and the other can build the robot. In programming, one can read the directions and one can operate the computer. For the testers, one can operate the robot while the other writes down observations. The kids should be trading roles so that they can have as much exposure to each aspect of building and programming as possible. For the building, roles can be swapped when building the second robot. For programming, roles can be swapped after the program has been saved for the first time. For testing, roles can be swapped between each of the two robots.

Here are some questions that you can ask the kids as the robots are being tested:

- Did the robot go in a perfect square? (It most likely won't.)
- Why didn't the robot go in a perfect square?
  A: The turns were not perfect 90 degree turns and/or the robot did not travel straight.
- What are the two things that you can change in the program to change the degrees of the turn?
  A: You can change the motor speed or the amount of time that the robot turns for.
- What could you change in the robot design to change the degrees of the turn?
  A: wheel size.
- How did the performance of the same robot differ when on different surfaces?
- How did the performance of the two robots differ when on the same surface?
- Why do you think there was a difference?

To Get Started

1. Follow the directions in the **Constructopedia** to build **Robo 1**.

2. Follow the directions in the **Constructopedia** to build **Pathfinder 1**.

If you only have one RCX programmable brick, you can build one robot completely, test it, then remove the brick and add it to the other robot for testing.

Do You Need to Install Firmware?

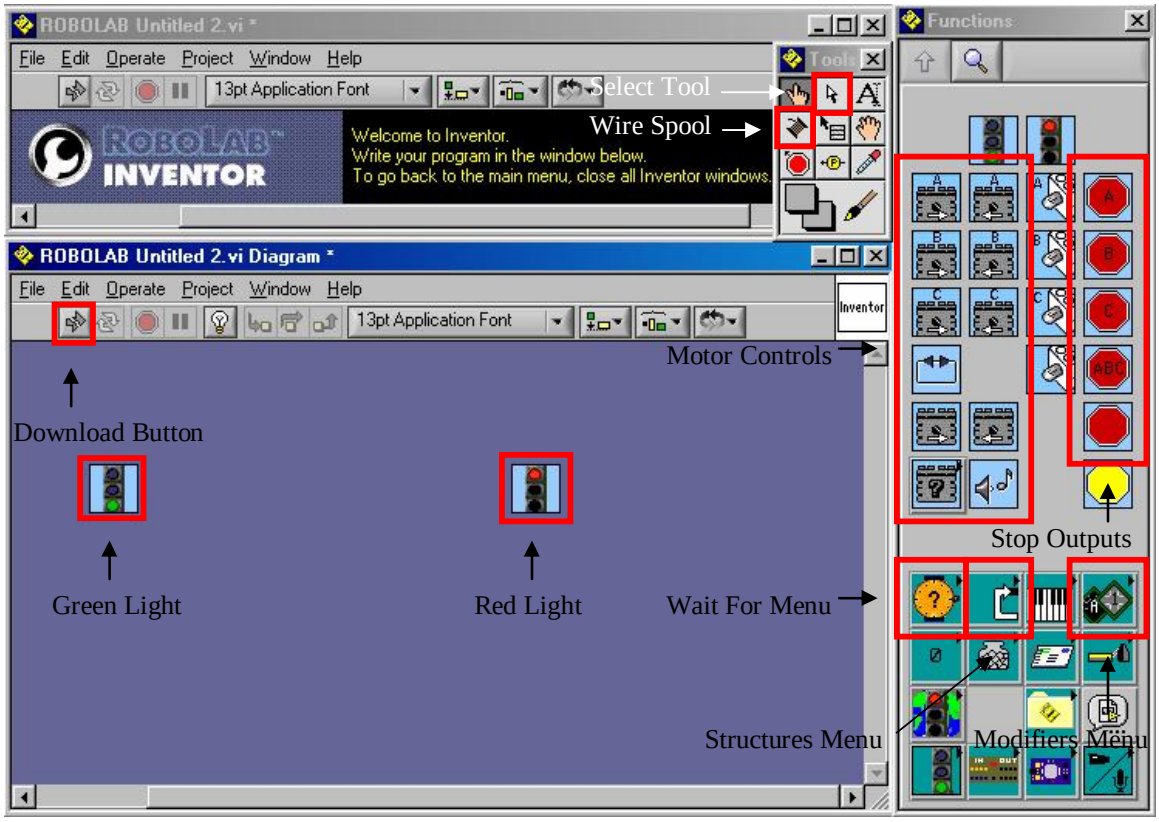
Firmware is a special program that must be installed onto your RCX so that other programs can communicate with the RCX. Firmware is installed if your RCX display shows:

If firmware is NOT installed, it will be installed the first time you try to download a program.

To Get Into Programming Mode

1. Select **Programmer** from the main menu.
2. Double click on **Inventor Level 4** to begin a new program.
3. Make sure that both the function palette and the tool palette are open. If not select **show tool palette** and **show function palette** under the **window** menu in the program window.

You should now see the programming area as shown below.



Take a moment in your group to review each of the icons as described in Figure 1. Feel free to click the menu on the right side so that you can see the different commands that you have access to. Try using Context Help to investigate the various commands.

Designing the program

Together, we are going to create a program to move your robot in a square.

To move the robot in a square, it must

1. move forward some distance
2. turn 90 degrees in one direction
3. move forward the same distance as before
4. turn 90 degrees in the same direction as before
5. move forward the same distance as before
6. turn 90 degrees in the same direction as before
7. move forward the same distance as before
8. turn 90 degrees in the same direction as before

Together, these steps are called an *algorithm*. An algorithm is a step-by-step plan of what the program should do, but it does not indicate how exactly to perform those steps. It is important that you come up with an algorithm *before* you start creating your program on the computer.

Notice that we are repeating the same two steps (move forward, turn) 4 times. Therefore, let's first create a program that moves forward then turns.

Create a Program to Move Forward

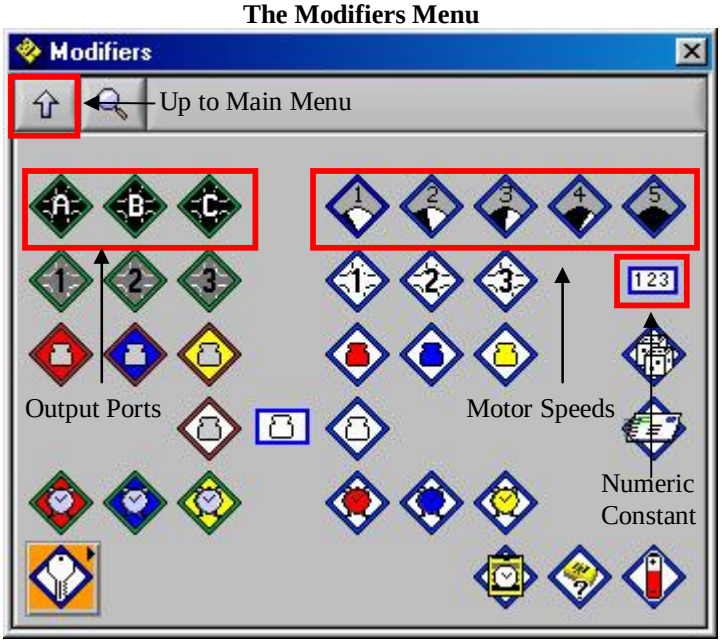
1. Click on the **Motor Forward** command.

2. Place the motor forward command next to the green light in the programming area by clicking.

3. If you held the command next to the green light for a while before setting it down, it might have wired itself to the green light. If not wire it by selecting the spool tool and clicking on the right side of the green light and then the left side of the motor command.

4. Select the **Modifiers** menu and observe the output port modifiers that are the black icons with letters. Select and wire **Output A** to the forward command. Then select and wire **Output C** to the motor port A.

The Modifiers Menu



You have just programmed the motors connected to ports A and C on the robot to turn in the same direction. However, this program will not run in its current state. The last command would have to be wired to the red light in order to have it run.

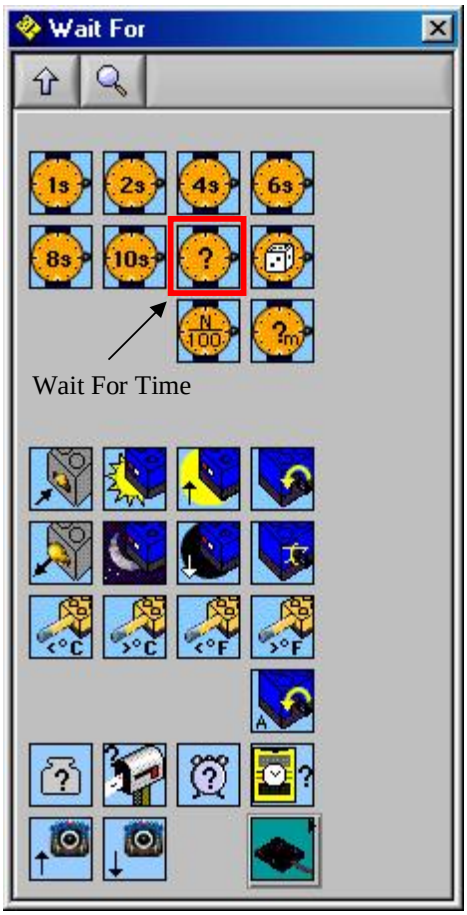
5. While still in the modifiers menu, select the icon with the partially filled circle segment and the number 4 on the gray background. This sets the motor power level to be 4. Click and place the **Power Level 4** modifier as you wired the output modifiers.

You have just programmed the power level to 4 for those motors connected to ports A and C. The power level indicates how quickly the motors will turn, thus how quickly your robot will move. You can change the power level by deleting the current modifier and including another similar modifier.

6. Press the **up arrow** in the command pallet to get to the main menu from the modifiers menu.

7. Select the **Wait For** menu.

The Wait For Menu.



8. Select the **Wait For Time** command that looks like a watch with a question mark and wire it to the motor command.

9. Press the **up arrow** to go up to the main menu and reenter the **Modifiers** menu.

10. Select the **numeric constant**, which is a blue box with numbers inside, and wire it to the wait for time command and enter the value 0.5 for a half-second time delay.

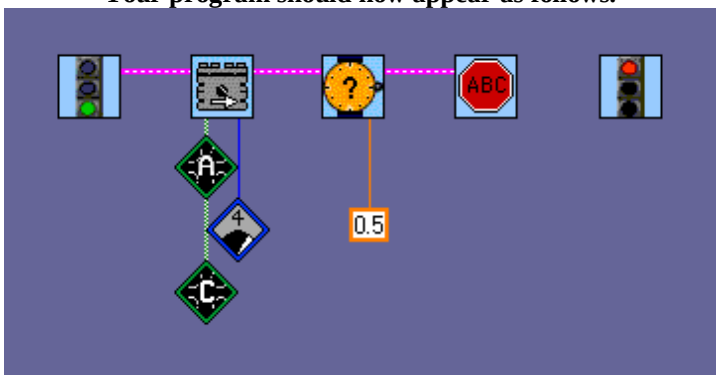
11. Return the main menu. Wire the stop sign labeled **Stop All Outputs** to the time delay.

You have just programmed the motors to turn on for 0.5 seconds then turn off.

☺ Congratulations!! ☺

You have now programmed both motors to move forward at power level 4 for 0.5 seconds.

Your program should now appear as follows.



Add a turn to your program

1. In order to fit the rest of the program, it may be necessary to move the red light. If necessary, select the red light and drag it to a different location while holding down the mouse button.

2. Add the **Motor A forward** command to the program wiring it to the stop sign.

3. Select the **Modifiers** menu. Wire a **Power Level 1** modifier to the motor A forward command. Afterwards, return to the main menu.

4. Similarly, add a **Motor C reverse** command and attach a **Power Level 1** modifier.

5. Enter the **Wait For** menu and select a **Wait For Time** command and wire it after the motor C reverse command.

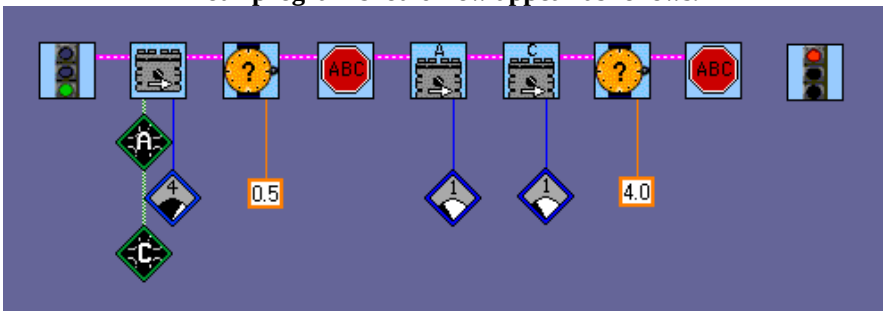
6. Return to the main menu and select the **Modifiers** menu. Wire to the wait for time command a **Numeric Constant** with value 2 signifying a 2 second wait.

7. After returning again to the main menu, select the **Stop All Outputs** command and wire it after the wait for time command.

You have now programmed the motors to move in opposite directions, at power level 1, for .2 seconds. Moving motors in opposite directions causes the robot to turn. The power level is decreased for better control during turning.

☺ Congratulations!! ☺
You have written a program to move your robot forward then turn.

Your program should now appear as follows.



Saving Your Program for the First Time

1. Select the **File** menu and select **Save As...**

2. When the dialog box opens, select and delete the name '**ROBOLAB Untitled 1'.**

3. Type in the box an appropriate name for this program such as '**Square**'.

4. Press the Save button.

Your Program is now saved

5. Click the **X** in the upper right hand corner of the window to close the programming environment and exit the program by pressing the quit button in the lower left hand corner of the program.

Opening a Saved Program

1. Open **ROBOLAB**.

2. Click on **Inventor Level 4** and then on **My Programs**. After that, on the far right you should see your program. Double click on the file to open it.

Recall that our algorithm required that we go straight and turn 4 times. Now we are going to add the *program control structure* called a *repeat block* to our program. A program control structure (those commands found in the Stack Controllers menu) is something that directs the flow of your program. In other words, it determines which of the tiles to perform and when. A repeat block is a group of tiles that are repeatedly performed for as many times as you indicate.

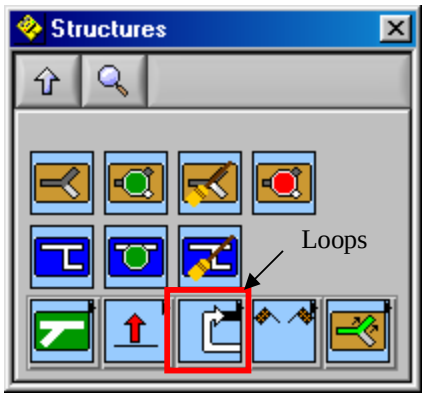
Adding the Control Structure Repeat

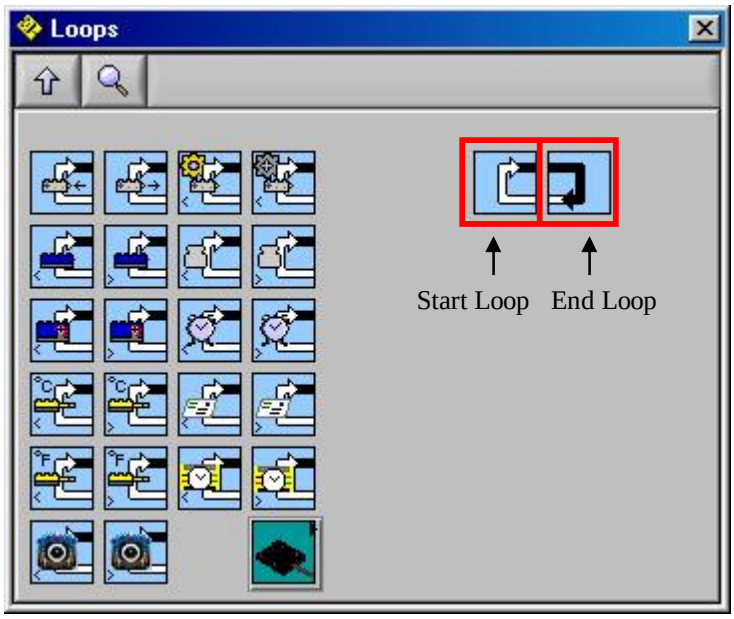
1. Select and delete the wire connecting the first motor forward command to the green light.

2. Click above and to the left of your program and, while holding down the mouse button drag a box around all of your program components. They should now be selected.

3. Click and drag your program to the right to make space for another command in front of your current program.

4. Click on the **Structures** menu and the on the **Loops** menu.





5. Wire a **Start of Loop** command in between the green light and the first motor on command.

6. Wire an **End of Loop** in between the final stop sign and the red light.

7. Return to the main menu and enter the **Modifiers** menu

8. Connect a **Numeric Constant** to the start of loop command and give it a value of 4.

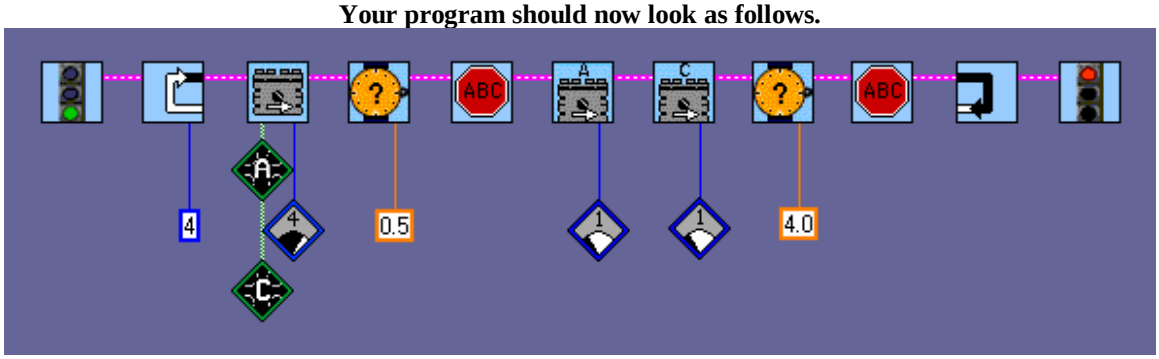
9. Note that by now your program is probably too large to fit on a single screen, so click and drag the scroll bars on the bottom and side of the window to shift its viewpoint.

Saving Your Program after Changing It

1. Select **File** and press **Save.**

You have now saved the changes to your program.

Your program should now look as follows.



Your program is ready to be *downloaded* to the robot. Downloading is the process of transferring something from one computer to another.

Download Your Program

1. Place your RCX about 6 inches in front of he transmitter so that the infrared window is facing the transmitter.

2. On the RCX, select the program number that you want the program downloaded to.

3. Press the arrow in the upper left-hand corner of the program window to download the program. If this is the first time downloading, the program will search for the port that your transmitter is connected to. In addition, if your RCX requires firmware, it will be downloaded at this time.

4. When your program finishes downloading, the RCX will beep.

☺ Congratulations!! ☺

You have just successfully written, saved and downloaded a program.