

Technical Learning

Preface: this instructional material is based on the Mindstorms programming language called RIS Version 2.0. This product is included in the retail version of the Lego Mindstorm Robot Invention System. It is also available for around \$20 from www.legoshop.com or call 800-453-4652. Please be aware that this product runs only on Windows computers, and will not work on the Apple Macintosh platform! RIS 2.0 is a wonderful development platform for Lego robots because it is very easy to learn and yet extremely powerful.

The other programming development platform allowed in First Lego League rules is called Robolab. Robolab is NOT discussed in this training document.

The winning robots in past competitions have been evenly divided between Robolab and RIS. It would appear that there is little advantage of one over the other. The one advantage that many people would agree on is that RIS is easier to learn. If you are going to use RIS, be sure to buy and use RIS 2.0 only!! RIS 1.0 and RIS 1.5 are a definite and huge disadvantage!!

Introduction

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?

Technical Learning

Builders

Follow the directions in the Constructopedia to build Robo 1.

Follow the directions in the Constructopedia to build Pathfinder 1.

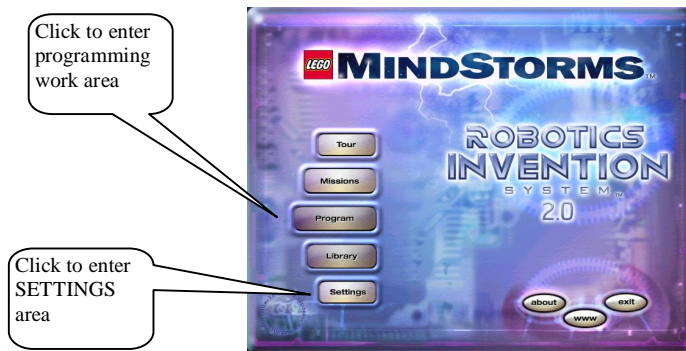
If you only have one RCX programmable brick, you can build one robot completely, test it, then remove the brick and add it to the other robot for testing.

GETTING STARTED – First-Time Users

1. Start up the Robotics Invention System.
2. Select **New User**.
3. Type your name and then Enter.

Watch the video for awhile, then press “**enter**” to move on.

You should now see a menu starting with the **TOUR** option.



Main Menu

You are now free to select from the main menu options.

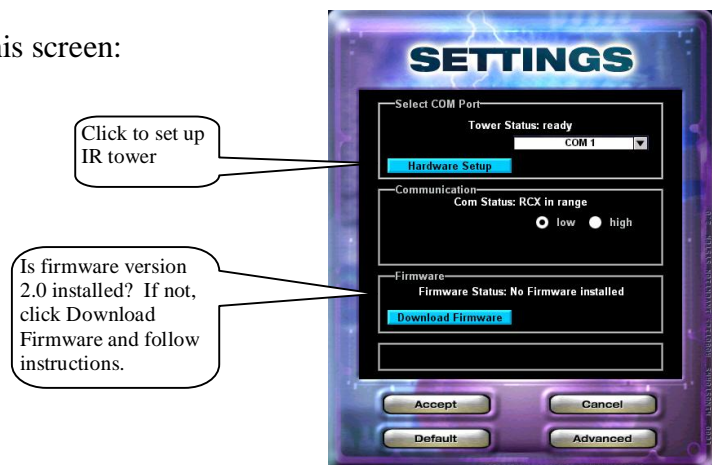
The Settings screen

The “Settings” menu option can help with all sorts of issues that may arise.

Connect the IR tower to your computer and turn on the RCX yellow brick. Make sure the switch on the front of the tower is moved to the left. That is the low power setting.

Point the RCX towards the tower – the black lens on the RCX should be facing the tower.

Click “Settings” on the PC. You will see this screen:



SETTINGS screen

At this point, the PC and the RCX will communicate with each other. This will only occur if the green light appears on the front of

the tower.

If the green tower light turns on then skip to the next sentence. If the tower's green light does NOT light, click "**Hardware setup**" on the SETTINGS screen to set up your IR tower.

If the RCX has firmware loaded, the PC will verify the firmware version. If the RCX firmware is missing, the PC will prompt you to download the firmware to the RCX. If required, go ahead and download the firmware.

You should **verify that firmware version 2.0 is loaded into the RCX.**

You will also notice the RCX of the battery level displayed on the SETTINGS screen, which will be helpful in the future.

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 .

What is important is the "00.00" on the display. The number to the right of the man can be 1,2,3,4, or 5. If you see the "00.00" displayed, firmware is installed. However, the firmware may be an older version. For this reason, it is best to check the firmware status on the "SETTINGS" menu on the PC.

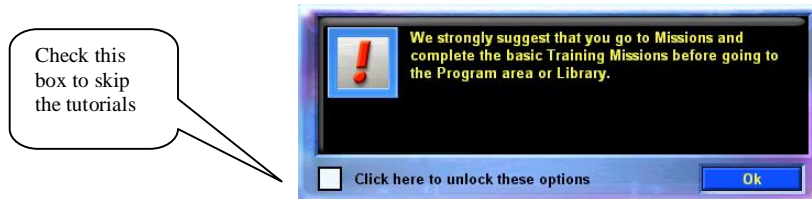
To Install Firmware

If the Firmware doesn't automatically install for some reason, or you need to reinstall firmware in the future:

1. Place the RCX 6 inches from the tower.
2. Turn on the RCX and point the RCX black window towards the tower.
3. From the **SETTINGS** screen, click "**Download Firmware**"

Once firmware is installed, you are ready to start programming!

1. Click “**Program**”
2. Check the box in the following window:



then click OK. This enables you to program without going through the tutorials. (However, the tutorials are worth going through!)

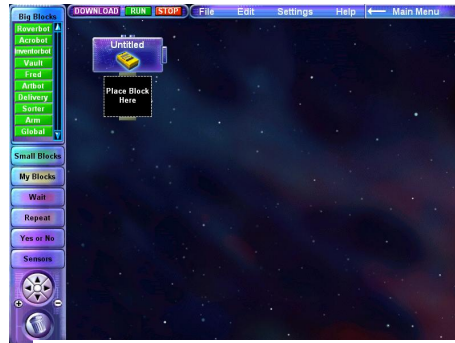
If the window doesn't appear, that is OK.

To Get Into Programming Mode

Click **“Program”** from the Main Menu

Click **“FREESTYLE”** from the Programming Menu

You should now see the main programming screen:



Programming work area

Take a moment in your group to review each of the icons. Feel free to click the menu blocks in the upper left-hand corner so that you can see the different commands that you have access to. You can scroll through the commands by using the Up and Down Triangles located at the top and bottom of each menu. Try the Help Menu too.

Setting up your programming environment

Edge Scrolling:

Edge scrolling allows your program window to scroll up and down, just like you are used to in a word processor.

To turn on edge scrolling, click “Settings” and click “Edge Scrolling On”. There will be a check mark next to it when it is on.

Commonly used icons:

The icons that your kids will be using the most often are the “**Small Blocks**” and “**My Blocks**”

The “**Big Blocks**” should never be used – they have no purpose in FLL, they are designed for one specific robot only, they are poorly written, and they do not produce repeatable results!

The image shows a vertical sidebar of icons from the FLL software. The icons are categorized into 'Big Blocks', 'Small Blocks', and 'My Blocks'. Callouts provide instructions on how to use these icons correctly and which ones to avoid.

- Big Blocks:** A callout points to the top section, stating "Never use Big Blocks!!".
- Small Blocks:** A callout points to the middle section, stating "Small blocks are used very often". This section includes icons for Power, Sound, Comm, Variable, Reset, and Advanced.
- My Blocks:** A callout points to the bottom section, stating "My Blocks should be used the most often!!". This section includes icons for Wait, Repeat, Yes or No, and Sensors.
- Wait:** A callout points to the Wait icon, stating "Commands used to wait for an event to occur".
- Repeat:** A callout points to the Repeat icon, stating "Used to repeat commands several times".
- Yes or No:** A callout points to the Yes or No icon, stating "If - then branch".
- Sensors:** A callout points to the Sensors icon, stating "Never use !! (Except variable sensor)".
- Navigation:** A callout points to the directional pad and zoom buttons, stating "Zoom view in or out".
- Trash:** A callout points to the trash can icon, stating "Discard unwanted commands (better idea is to put unwanted commands back where they came from or you can lose a Myblock by mistake!)".

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 green “**Small Blocks**” icon, and open the “**POWER**” commands.
2. Click on the lower triangle at the bottom of the Commands menu until you see **set direction**.

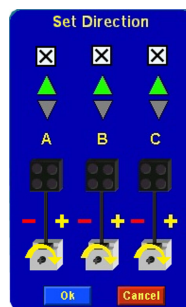


3. Click on the **set direction** tile to pick it up.
4. Drag this tile so that it is directly under the **untitled** tile (fitting it like a puzzle piece).
5. Click again to lock it in place.

Your program should look like this:



6. There are two methods to open a command up to see its options. Either click the small tab on the command’s edge, or *right-click* on the command to open it up. Right-click means to click the right mouse button while the cursor is over the described area. You will see the **Set Direction** command options.

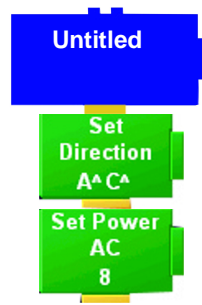


7. Click the box next to **B** to turn that motor off (your robot won’t have a motor connected to port B). Note that the selected direction arrows under A and C are pointing in the same direction.
8. Click OK to finish editing the options on the **set direction** command..

You have just programmed the motors connected to ports A and C on the robot to turn in the same direction.

9. Find the **set power** tile and place it below set direction.
10. Right-click **set power**.
11. Click in the box next to **B** to turn it off.
12. Click OK.

Your program should look like this:



You have just programmed the power level to 8 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 clicking the + or – buttons next to the power level.

13. Find the **on for** tile and place it below **set power**.
14. Right-click **on for**.
15. Click **OK**.

You have just programmed the motors to turn on for 5 seconds.

Your program should look like this:

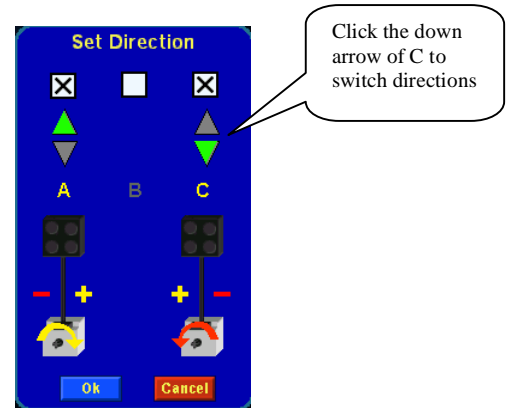


☺ **Congratulations!!** ☺

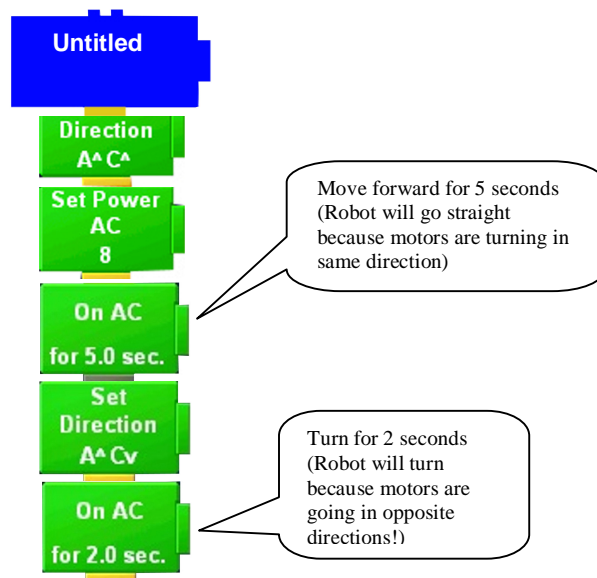
You have now programmed both motors to move forward at power level 8 for 5 seconds.

Add a turn to your program

1. Add **Set Direction** to your program.
2. Open the **Set Direction** command.
3. Turn motor **B** off.
4. Reverse the direction of the C motor by clicking the **down arrow** for **C**.
5. Click OK to close the box.
6. Add **on for** to your program.
7. Open the **on for** block.
8. Set the time to 2.
9. Click OK to close the box.



Your program should look like this:



You have now programmed the motors to move in opposite directions, for 2 seconds. Moving motors in opposite directions causes the robot to turn. The power level could be decreased for better control during turning. Try it!

☺ **Congratulations!!** ☺

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

Saving Your Program for the First Time

1. Open the FILE menu at the top of the programming screen.
2. Click **Save As**.
3. Key in a name for your program – name it **SQUARE**.
4. Click **SAVE**

You program has been saved.

5. Click the **Main Menu** button in the upper right corner to close the programming environment.

Opening a Saved Program

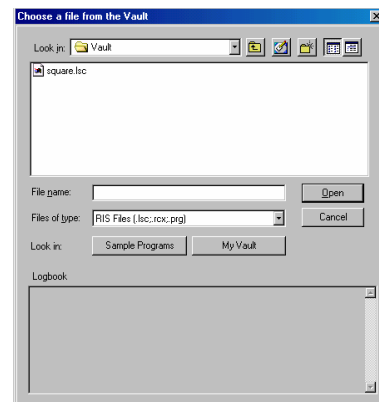
Get back into the programming environment (if you forgot how, look at the beginning of this document).

Select **VAULT** instead of **Freestyle** to open existing programs.

Choose VAULT to open existing programs



Double-click the program **SQUARE** to open it. You should now see the program as you saved it above!

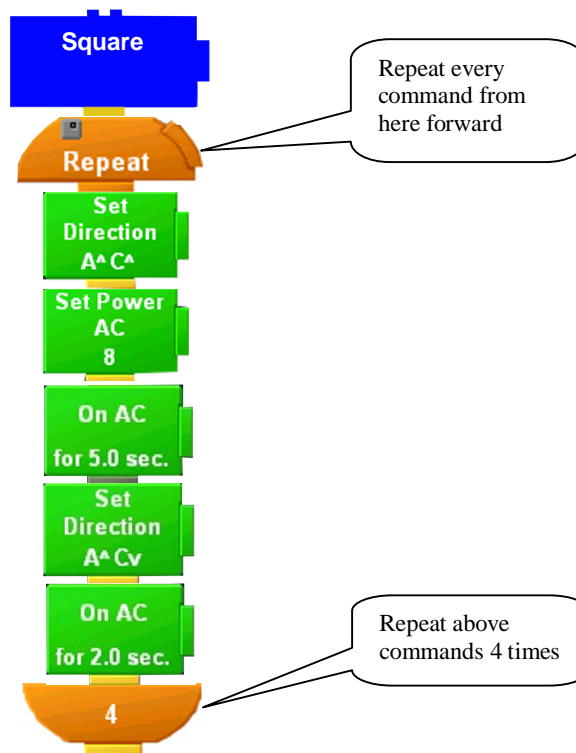


Recall that our algorithm required that we go straight and turn 4 times. Now we are going to add the *program control structure* called a *repeat block* to our program. A program control structure (those commands found in the **REPEAT** menu) is something that directs the flow of your program. In other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

Adding the Repeat

1. To move a string of commands in your program, click the first green command that is directly underneath the blue program “**square**” block.
2. Drag it to any location on the screen and click again to temporarily park it.
3. Click the **REPEAT** menu to reveal all the repeat commands.
4. Find the **Repeat FOR** command and place it under the blue program “**square**”.
5. Open the **Repeat For** command by clicking on the edge tab
6. Click the plus button (+) until 4 appears.

7. Click the first **set direction** tile that you just placed to the side.
8. Drag it directly underneath the **Repeat** block (not the **End Repeat** block) and click again to drop it. Note that the **End Repeat** block automatically moves to the bottom.



9. Move your cursor to the bottom of the screen and then back to the top. Notice how the commands scroll up and down so you can see all of them.
10. Click the gray square in the upper left corner of the **Repeat** command. Note that all the code shrinks into one single command. This makes it easier to see more of your code as you keep adding commands.
11. Click the gray square to see all the commands again.

Saving Your Program after Changing It

Save the program as before (**FILE – SAVE**)

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 the transmitter so that the infrared window is facing the transmitter.
2. Click the **download** button at the top of the screen.
3. When the RCX beeps, downloading is complete.

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

- End of Document -