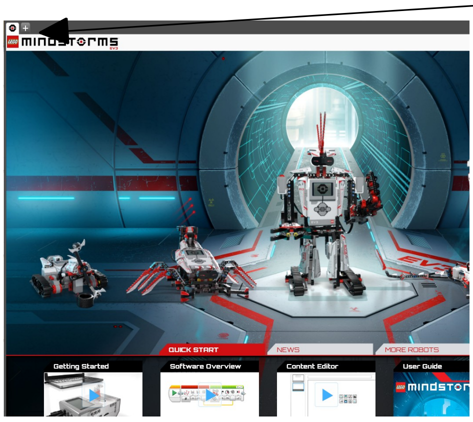
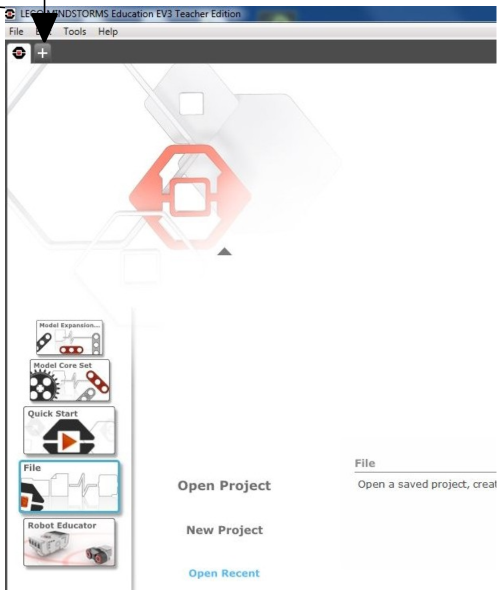
Getting started:

If you see a screen like one of these: click on the + tab.

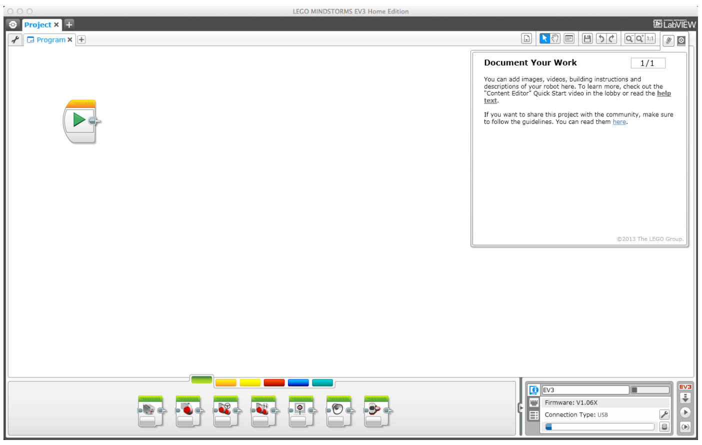




(Home edition above.

LEGOEducation edition to the right.)

You will see a screen like this.



Document Your Work 1/1

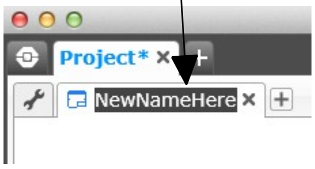
You can add images, videos, building instructions and descriptions of your robot here. To learn more, check out the "Content Editor" Quick Start video in the lobby or read the **help text**.

If you want to share this project with the community, make sure to follow the guidelines. You can read them here.

©2013 The LEGO Group.

This is the space where you write your robot program. The program starts on the left, at the green arrow, and is built up to the right by connecting blocks together in a line. The blocks you can use are chosen from the coloured tabs along the bottom.

The program is called Program and it is part of a project called Project. A Project might have several files associated with it: programs, images and sound files. Give your program a name by double-clicking on the word Program and typing in another name.



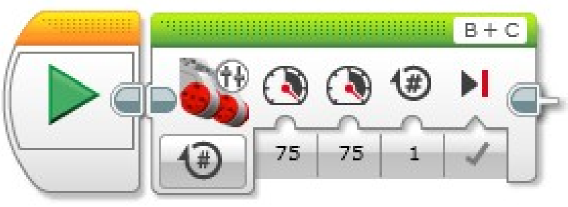
Choose Save Project As **from the File Menu and you will see a** Save As **window to choose a name for your Project and a place to save it. Projects are saved with** .ev3 **after their names.**

Moving

The robot can move forward and backward (reverse). It has one motor for each wheel. The motors are connected to the B and C ports of the robot by cables.

(If this is not the case on your robot you will need to adjust the instructions in this book accordingly.)

1. **Drag a** Move Tank **block (4th along, green tab) and connect it to the green arrow.**



The **Move Tank** block lets you control how far your robot will move but it doesn't use metres or centimetres, and you might start to see why soon.

You can change the settings on your block.

If you don't change anything, the robot is set to move the **B** motor and the **C** motor at power 75 for 1 rotation and then brake.

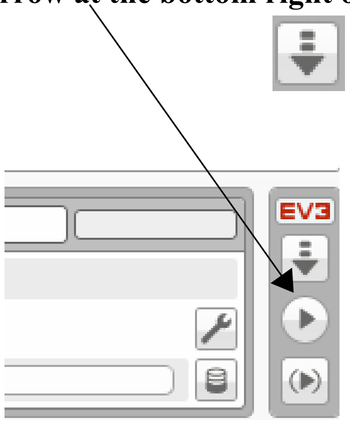
Rotations

A rotation is a turn. One rotation is one whole turn of the wheel.

Let's find out how far the robot will move with one rotation.

You will need to download the program you have written on the computer (the Move Tank block) to the robot. Download means to copy the program from your computer to the robot.

2. **Download your program to the robot using the Download arrow at the bottom right of your screen.**



- The USB cable needs to be connected to the robot's USB port while the computer is downloading.
- The robot needs to be turned on. Press the Dark Grey button on the robot if your robot isn't turned on.

The robot will make a noise when the downloading is finished. After this, you may pull out the USB cable.

It is best **not** to use the Run arrow unless the robot is on the floor already.

3. **Run your program on the robot.**

- Push the right button to select the second tab and see your Project's name.
- With your Project's name highlighted, use the dark grey button to open your project folder. You will then see your program's name.
- Use the lower button to highlight your program's name.
- Then press the dark grey button to run the program.

Now, if you move back to the arrow tab, your program will be listed here because it was one of the most recent programs run.

Run the program again from the arrow tab.

4. **Mark with a pencil a place on the side of a wheel hub. Watch how far the mark moves when your program is run.**

© LEGO Group 43.2x28 S

Task 1. Fill in the 10 Boxes about Measuring the Move

5. **Measure how far your robot moves with 1 rotation.** **1.1**

6. **Calculate how far your robot will move with 2 rotations.** **1.2**

Try it, and measure it. Note – don't use a second block. Change the 1 to 2 on your existing block. **1.3**

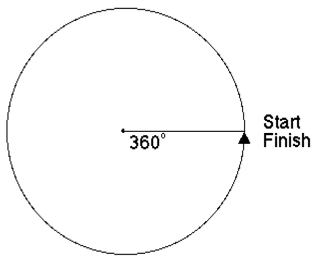
Question: Does doubling the rotations double the distance travelled?

Degrees

You can measure how far the wheel will go round using a measure called degrees.

All the way round a circle (or a wheel) is 360 degrees, which is written 360^o.

If you choose either 1 rotation or 360 degrees, your robot should do exactly the same thing - turn the wheel 1 complete circle.



1. **Change your** Move Tank **block's setting from** On for Rotations **to** On for Degrees**. The 1 will automatically update to** 360 Degrees.

If you download and run your program now, the wheel should still go round exactly once.

Can you work out how many degrees would make the wheel turn half way round? **1.4**

Can you work out how many degrees would make the wheel turn a quarter of the way round? **1.5**

Degrees are really useful if you want the wheel to turn just a little.

Task 2. Distance Challenge

1. **Make your robot travel exactly 8 cm. See if you can get it right first time, using the measurements you have already made to calculate the settings required.**

2. **Make your robot travel exactly 1 metre. Once again, work it out first.**

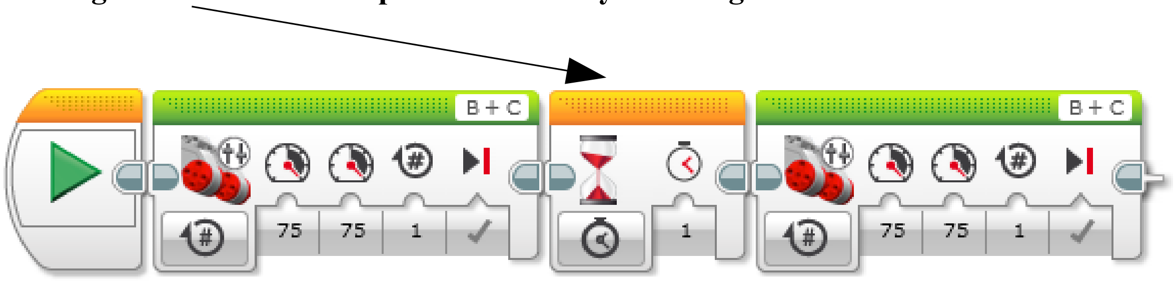
Building programs.

Longer programs are built up by joining blocks together. The program runs the blocks in order from left to right.

1. **Drag another** Move Tank **block to your program space and connect it to the right of the one you already have.**

2. **Select the orange tab and choose the** Wait **(sand timer) block.**

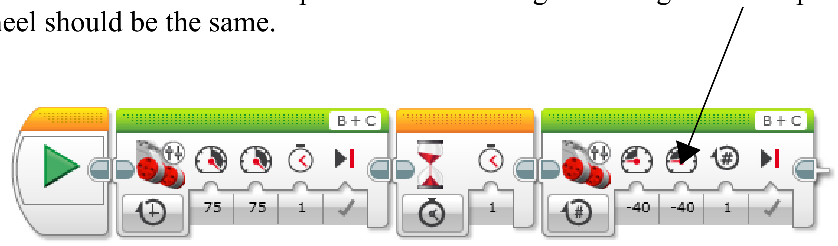
3. **Drag the** Wait **block so it pushes between your two green** Move Tank **blocks**.



The program pictured above will make your robot perform three tasks, one after the other, from left to right. They are connected in sequence. (A sequence is an ordered list.)

You can change the settings on any block, then download again to update the program on your robot.

4. **Change the settings on all 3 blocks in your program to make your robot**
   - move forward for 1 second
   - stop and wait for 2 seconds
   - move backwards (reverse) at power -40 for 1 rotation. You can drag the slider OR type the number in the boxes. To keep the robot travelling in a straight line the power driving each wheel should be the same.



5. **Download and run.**

Have you noticed the changes to the pictures on the block when you change the settings?

When you write longer programs these pictures will help remind you which block is doing what. The two Move Tank blocks in the picture above have different mode settings – the one on the left is set to On for Seconds and the other to On for Rotations.

Task 3. Move Questions

3.1 What do you think would happen if you put smaller wheels on the robot then ran your “exactly 1 metre” program again?

3.2 What do you think would happen to your robot if one wheel was bigger than the other when you ran your “exactly 1 metre” program? Why?

3.3 Why do you think the MINDSTORMS program doesn’t just let you tell the robot how many centimetres or metres you want it to go?

3.4 When you make a change to your program and download it to the robot, what has happened to the old version of your program on the robot?

3.5 What do you think would happen if you put the brake on one wheel and turned the other wheel?

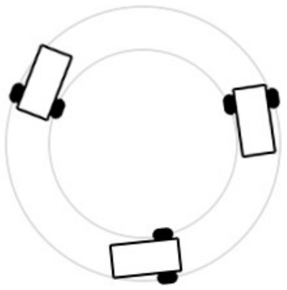
Turning

Task 4. Fill in the 11 Boxes about Turns

How can your robot turn when it doesn't have a steering wheel?

The picture shows a robot moving in a circle. It shows 2 wheels which we can call the inside wheel and the outside wheel.

The inside wheel makes the small circle and the outside wheel makes the big circle as the robot goes around.



Which wheel moves the furthest? **4.1**

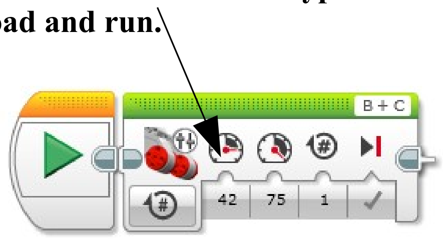
Which wheel turns the fastest? **4.2**

Move Tank Turn

1. **Delete all blocks in your program. Drag one** Move Tank **block to the Start position.**

2. **Change 1 of the motor's power settings, leaving the other at 75. You can type or use the slider. Make sure the turn is for** 1 Rotation. **Download and run.**

Typing the Power is often easier than using the slider if you have an exact number in mind.



3. **Put a marker on the side of each wheel. Watch each wheel in turn as the program runs.** Do they **both** make one whole rotation?

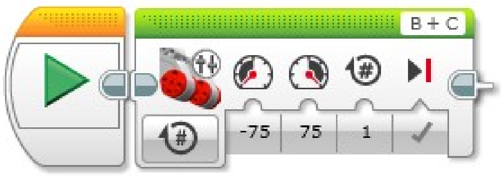
4.3

If not, describe what happens.

4.4

4. **Change the** Rotations: **setting to 10 rotations. Download and run.** Can you see the circle the robot is trying to travel around?

5. **Now change the power setting to** -75. **Download and Run.**



6. **Mark with a pencil a place on the side of each wheel hub. Watch each wheel in turn as the program runs.** Do they both make one whole rotation?

4.5

Have you noticed that one wheel is now turning the other way?

Question: Where is the centre of the turning circle now?

4.6

7. **Choose a start position for your robot that you can remember, and run your program. Estimate (or measure) how many degrees your robot turns.**

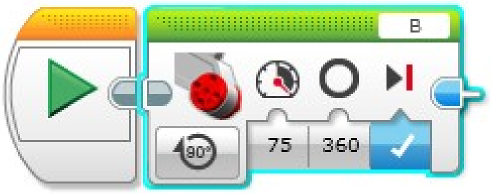
4.7

8. **Now change the power setting to** 10 **and** -10. **Download and Run using the same starting position as before. Does your robot turn the same distance as before?**

4.8

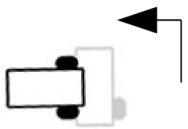
One Wheel Turn

1. **Delete all your blocks. Drag a** Large Motor **block to your program. Set it to** On for Degrees.



2. **Try different numbers in the** Degrees **box until you get the robot to make a quarter turn.**

How many degrees will give you a quarter turn? **4.9**



This number will be very useful for the Little Red Riding Robot challenge, and other challenges ahead.

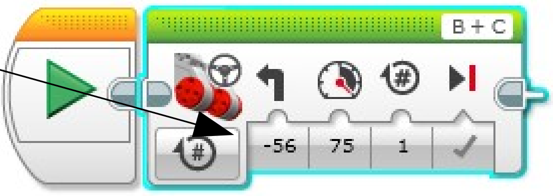
3. **Change the motor selected to motor** C **to make the robot turn the other way. Test your program.**

Move Steering Turns

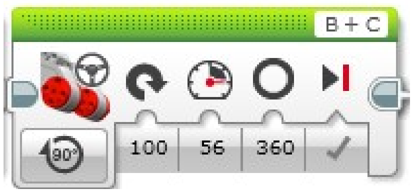
There is another Move block for moving both wheels together. The Move Steering block. This block just has one setting for power. It has a slider for choosing the angle of the turn. The slider range is from -100 to 100. A setting of 0 should theoretically make the robot travel straight ahead, but in reality the Move Tank block is better for this.

1. **Delete all blocks in your program and drag one** Move Steering **block to the start position.**

2. **Experiment with the slider settings.** Notice which way the wheels are turning. Make sure you can program a wide turn as well as a tight one using this block.

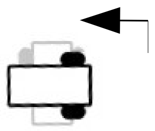


3. **Make your robot perform a Steering = 100 turn for 1 rotation. Lower the power if you like. Change the** Rotations **setting to** Degrees. **Download and run.**



4. **Without changing the** Steering **setting, try different numbers in the degrees box until you get the robot to make a quarter turn.**

How many degrees will give you a quarter turn? **4.10**

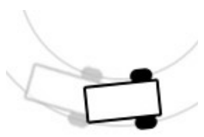


Turn summary:

These are the three different types of turn your robot can make, but there are different ways of making them happen.

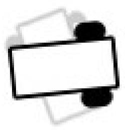
1. **The <u>curve</u> turn** - the robot will move in a circle.

   This turn is made by turning the robot's wheels at different speeds.



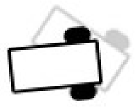
2. **The <u>point</u> turn** - the robot spins around a point under its middle.

   This turn is made by turning the robot's wheels in different directions; one forward, the other backwards.



3. **The <u>wheel</u> turn** - the robot spins around one wheel.

   This turn is made by one wheel staying where it is and the other wheel turning.

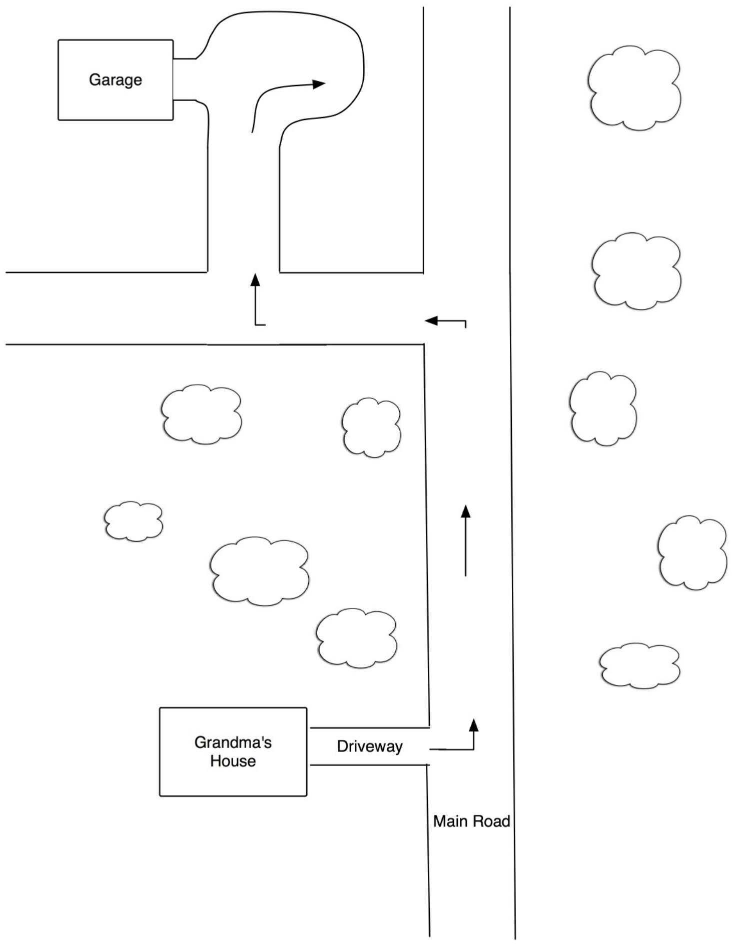


Task 5. Little Red Riding Robot Challenge

Your Little Red Riding Robot has been to visit Grandma and it is time to go home. The map shows you the shape of the turns she has to make.

Make sure she stops and waits to look for cars at the end of Grandma's driveway before turning into the main road.

She always reverses into her garage, because she likes to come out facing forwards.



Program your robot to make the journey home.

Use the table to help with your planning.

If you have been careful filling in the boxes so far you might be able to make the journey correctly on the first attempt.

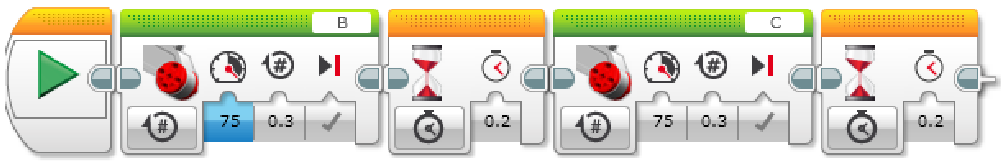
Now there's a challenge!

List of journey actions, in order	Measured distance or turn	Block used	Block Setting (degrees, rotations, seconds, power, B/C . . .)
along Grandma's driveway			
stop			
wait			

Task 6. Robot Walk Loop

The task is to make your robot look as much as it can like it is walking by moving one wheel just a little, then the other.

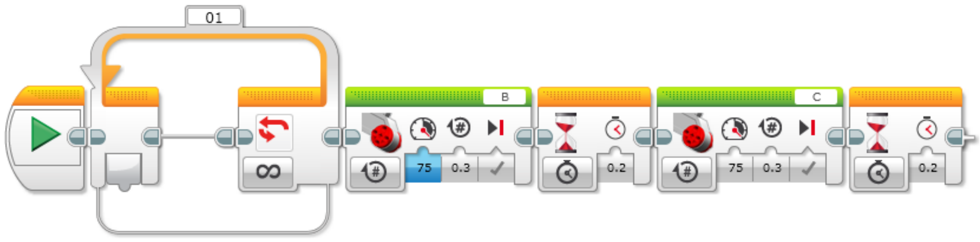
1. **Program the robot's first two 'steps'. Use the** Large Motor **block for each step.**
2. **Drag a** Wait **block after each 'step'.** How long do you want your robot to wait between 'steps'? A second is a long time for a robot to do nothing. Try 0.2 of a second.



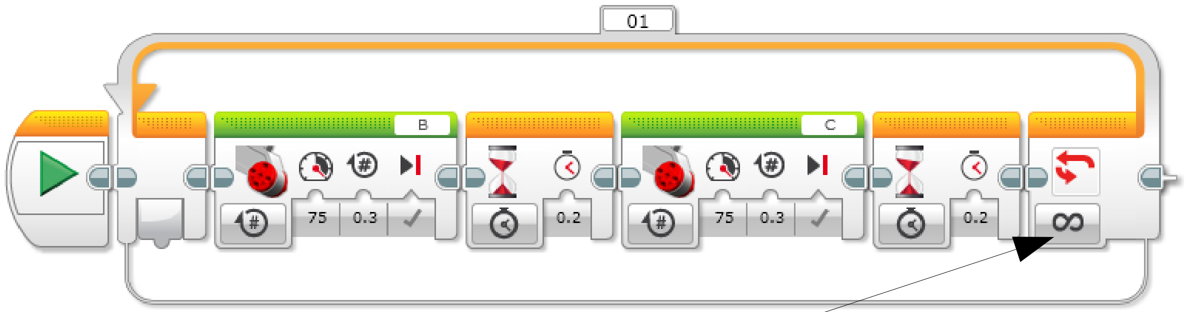
Loops

Loops are very useful for saving boring repetition. All the other robot steps will just be the same as these two. Rather than drag more and more **Move** and **Wait** blocks down, you can make the program repeat these steps over and over by putting them in a loop.

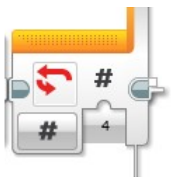
3. **Drag a** Loop **block to the first position of your program space.**

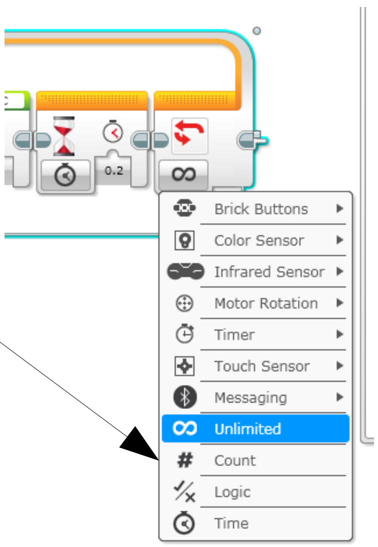


4. **Drag your program blocks, keeping them in the same order, inside the** Loop **block.**



5. **Change the** Loop **control by clicking on the infinity sign (∞) at the end of the loop.**
6. **Choose** Count **then type** 4 **in the box.**





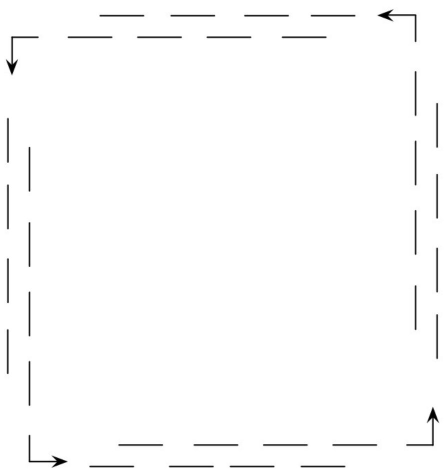
7. **Download and run.** Did your robot stop after 8 (4 pairs of) steps?

8. **Change the settings on the blocks to give your robot different kinds of walking action.** Can you program it to look like

   - an old man
   - a running child
   - someone with a limp

9. **Name your program and save your Project using** File > Save Project As **because you will want to use it again later.** The best name for a program or project is one which will remind you what it does.

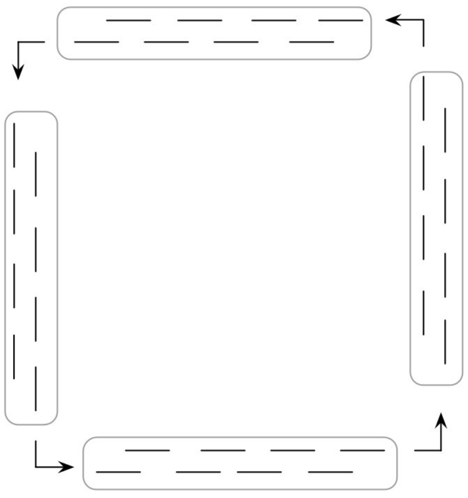
Task 7. Square walk

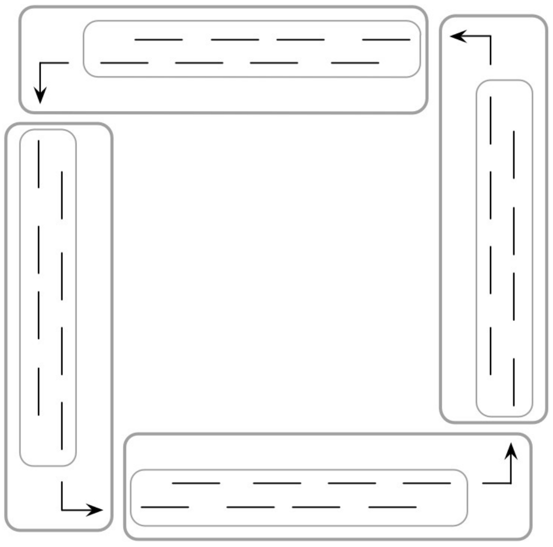


Build on your Walk Loop to make your robot "walk" in a square. Start with your walk loop from the last exercise. After it finishes its 8 step loop, make it turn a quarter turn (1 rotation with 1 wheel is one way to do this).

Make it repeat these action 4 times:

- You could copy your loop and turn blocks 4 times





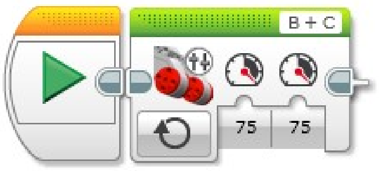
- or you could save yourself a lot of bother and put them in another loop.

Sensing - Reflected Light

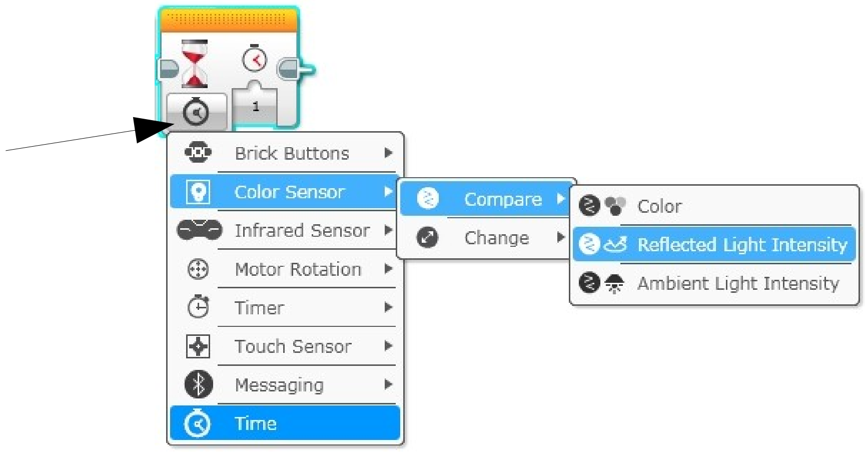
Your robot can detect when it comes to a dark line on a light coloured floor.

1. **Start a new program. Drag a** Move Tank **block to the** Start **position and set it to** On.

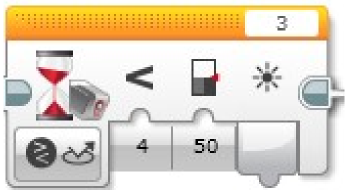
Note - NOT On for Rotations, but simply On. The block will get shorter, losing its Rotation and Brake at End options.



2. **Drag a** Wait **block to the program. We want to wait for** Color Sensor, **not** Time**, so click on the lower left button, called the** Mode **selector, of the block. This opens up choices – select** Color Sensor > Compare > Reflected Light Intensity**.**



3. By default, the block setting will test this condition: **Is the reflected light reading less than 50?** The answer is either true (yes) or false (no).



4. **Drag another** Move Tank **block to the end of the program, and set it to** Off.

5. **Download. Put your robot down on something light coloured (white or yellow). Point** your robot towards something dark, like a black stripe. **Run**.

Does it stop when it reaches the dark colour? **8.1**

What happens if you start it on the dark colour? **8.2**

What happens if you take the last **Move Tank** block (Off) away?

8.3

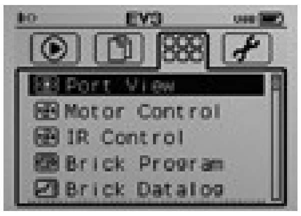
The Color Sensor is sending a light out and measuring how much light is reflected back. Different colours will reflect different amounts of light.

Other things will also change the reading. If the sun comes out, a light in the room is turned on or somebody makes a shadow, the reading will change.

It is useful to know what readings your Color Sensor makes in your current environment.

Reading the light level on the EV3 using Port View.

1. **Use the right button to get** Port View **showing on the robot's screen.** Press Dark Grey to select Port View.



2. **The default is Port 1. Find** Port 3 **by pressing the right button 2 times. It should be displaying**

   3:COL-REFLECT and some pct (percentage) underneath.

   This is the Color Sensor reading. It can read from 0 to 100.

Put your robot's Color Sensor over the black line and write down the reading.

Black

8.4

Put your robot's Color Sensor over the white background and write down the reading.

White

8.5

It is important to know these numbers so that you can make good choices about the numbers to use in your program.

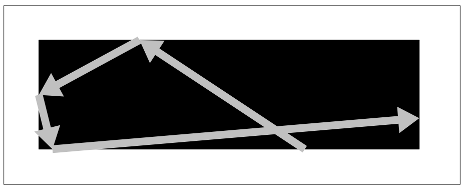
3. **Set the** Wait – Color Sensor **block so it moves on dark colours and stops when it reaches something light coloured. Test your program.**

Task 8. Tiger Robot Challenge

First, make sure you have answers for boxes 8.1 – 8.5.

Write a program to keep your tiger robot pacing in its cage. Make your robot move in a straight line across its square black tiger cage. It turns when it meets the edge of its cage then keeps on in a straight line until it meets another edge. Repeat this for ever.

You will need to use a loop, but this time set it to Unlimited rather than a Count.



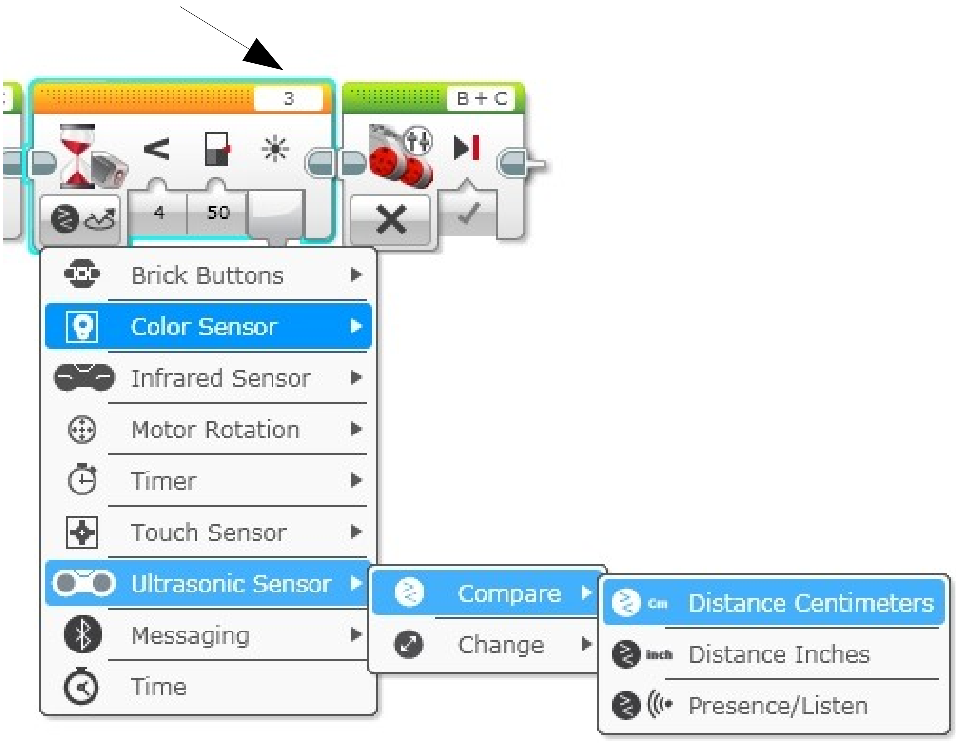
If you are keen, see if you can contain your tiger better by having it back away from the edge before it turns.

Ultrasonic Sensor

The 'eyes' on your robot can detect objects as far as two and a half metres away. It uses an ultrasonic sensor, like a bat or a radar. A signal is sent out, and if an object is in front, the signal bounces back. The time taken for the signal to come back is a measure of the distance between the object and the robot.

1. **Start a new program. Drag a** Move Tank **block to the** Start **position and set it to** On.

2. **Drag a** Wait for Time **block to your program. Change its setting to** Ultrasonic Sensor > Compare > Distance Centimeters**. Make sure the port number matches the port your sensor is connected to. The default for the Ultrasonic Sensor is port 4.**



If you are using the Home Mindstorms software rather than the Education version you will need to download the Ultrasonic block. One place to get it from at the time of printing was:

http://www.lego.com/en-us/mindstorms/downloads

EV3 Software Block Download (PC/MAC)

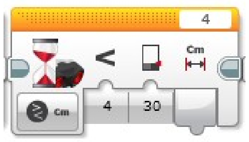
Ultrasonic Sensor Block

Once you have downloaded the block file, return to the EV3 software. From the **Tools** menu choose **Block Import**.

Browse to find your downloaded file (**Ultrasonic.ev3b**).

Close and reopen the application in order to see the new block's effect.

3. **Set it to** Distance < 30 cm.



4. **Drag another** Move Tank **block and set it to** Off. **Download**, point your robot towards a wall, and run.

   You will have to be careful not to let the robot 'see' your hands when you press the run button. The robot may also see its own cables if they are in front of its 'eyes'.

Measure how far away your robot stops from the object it has detected. **8.1**

Where should you measure from on the robot? **8.2**

Large sized objects with hard surfaces return the best readings. Objects made of soft fabric or that are curved [like a ball] or are very thin or small can be difficult for the sensor to detect.

Two or more Ultrasonic Sensors operating in the same room may interrupt each other’s readings.

Source: http://mindstorms.lego.com/overview/Ultrasonic_Sensor.aspx

Task 9. Runaway Robot Challenge

Make your robot move forward slowly. When you move in front of it, it should turn a half turn away from you and keep moving forward (away from you). Repeat these actions for 10 seconds.

Task 10. Find and Point Challenge

Make your robot turn on the spot slowly. When it sees an object close to it e.g. a coke can, it should stop. See if you can make the robot point straight at the middle of the object when it stops.

Task 11. Edge Follower Challenge #1

Find or make a black line on a white background. Determine a good number to use for the difference between black and white. We recommend a number half way between the black and white readings.

Repeat the following sequence in a new program (put it in a loop):

```
move the B wheel, power 60, on
wait until the Color Sensor finds light coloured background
stop the B wheel
move the C wheel, power 60, on
wait until the Color Sensor finds dark coloured line
stop the C wheel
```

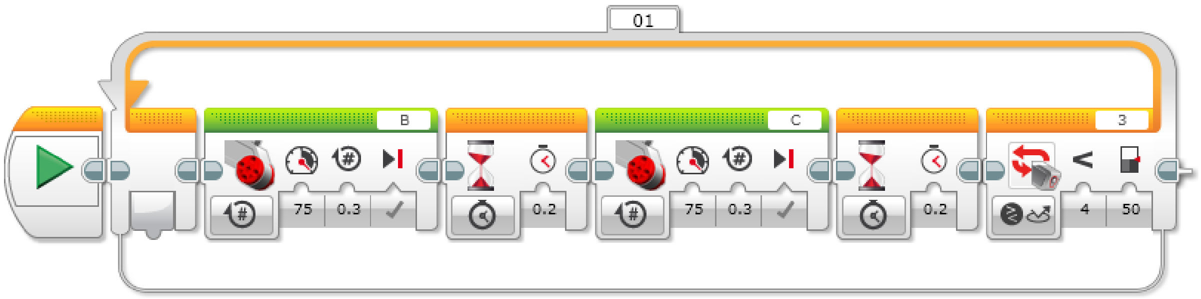
Download your program. Put your robot down with the light sensor on a black line and run the program.

Task 12. Racing Robot

Racing Robot needs to:

- walk to the starting line (a black stripe) *steps 1 & 2 below*
- wait for the dark grey Enter button to be pressed *step 3 below*
- race off to the finish line (another black stripe) *step 4 below*
- beep and stop. *step 5 below*

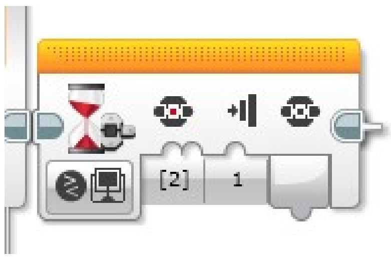
1. **Start by writing a robot walk program or opening the one you made in Task 6.** Don't make the steps too big – you'll see why shortly. Try 80 degrees.
2. **Select the** Loop **block**. **It is probably showing Count 4. The robot needs to walk until it senses the start line, so set the Loop control to** Color Sensor > Reflected Light Intensity **Set the** Loop **to finish when the light sensor sees black.**



The robot should now walk to the Start line and stop at least some of the time.

3. **Put a** Wait **block after the walk loop. Control this loop by** Brick Buttons > Compare.

   Use the default settings: Button ID [2] (the dark grey middle button) and State 1 (pressed). The program will just wait for the button press before continuing.



The next action the program needs to perform is to race to the finish line.

4. **Add another** Loop **to your program. Put an unlimited** Move Tank **block inside the loop, and finish the loop when the robot sees black. This is a race, so set the** Power **to** 100.
5. **Finish the program with a Beep and stop the wheels.**

This program you have written probably has at least one and maybe two serious problems. See if you can work out what the problems are. The answer is at the bottom of the next page.

Fixing problem 1

6. Make the robot move past the start line before it starts looking for the finish line. There are many ways to do this. See if you can get one working.

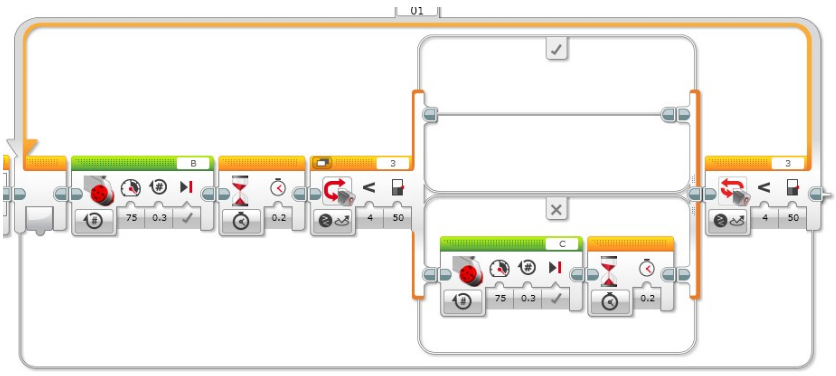
Fixing problem 2

7. **Drag a** Switch **block and put it between the robot's 2 steps.** A Switch gives the program a choice of 2 paths to follow.
8. **Set this** Switch **block to be controlled by the Color** Sensor > Reflected Light**. Set it the light level to the same as the one you have chosen to finish your walk loop**.

   The top path (sequence beam) is for the comparison being **true** (light level < your setting), so the top path is for what you want to happen if the robot is on black after the first step in its loop. (Nothing. It has found the start line and is ready to race.)

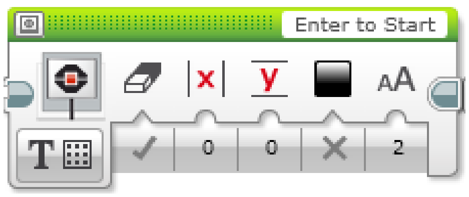
   The bottom path is for a robot which is still on white after its first step. You want it to take the second step in the loop.
9. **Drag the "second step" blocks to the bottom path of the** Switch.

   The robot is less likely to miss seeing (sensing) the start line now.



10. Finish the Racing Robot **challenge by**

- making sure your robot takes steps small enough that it NEVER misses the start line
- putting a text message on the screen saying "Enter to Start" so that other people will know what the robot is waiting for on the starting line.



- take this message off the screen when the race has started. You could replace it with one which says "Racing now"

* * * * *

The robot is still sitting on the black start line when it starts to race. It is looking for a black finish line, and thinks it has found it straight away.

The robot doesn't always find the start line. It takes 2 steps inside the loop before each check to see if it is on black, and often this is enough to go past the start line (a false start).

Task 13. Yo-yo robot

This program moves the robot forward a little if there is nothing in front of it, and backwards if there is something in front of it.

1. **Drag a** Switch **block to a new program.**
2. **Set it to** Ultrasonic **(distance) sensor,** distance less than 30cm.

You will see there are 2 paths (MINDSTORMS calls them sequence beams) in the **Switch** block.

Your program will choose which one to follow depending on the **Distance** setting on the **Switch** block and the information received by the sensor.

In this case, if the robot is less than 30cm from an object, it will follow the top sequence beam. Otherwise it will follow the lower sequence beam.

3. **Drag a** Move Tank **block into the top sequence beam. Set it to move backwards 3 rotations.** (If the robot is closer than 30cm to an object, it will move backwards.)
4. **Drag a** Move Tank **block into the lower sequence beam. Set it to move forward 1 rotation. (**If the robot is further than 30 cm away from an object, it will move forward 1 rotation.)

There is one further step needed. The ultrasonic sensor is likely to see your hand as a close object when you push the run button.

5. **Drag a** Wait **block to the beginning of your program (before the** Switch **block). Set the time for long enough to get your hand away.**

Download. Put the robot near (and pointing towards) a wall and run the program. If your robot is very close to the wall, it should move backwards. Otherwise it will move forward. Shift the robot and run the program again, and again, and again . . .

The **Switch** block can be thought of as an 'if ' then an 'else' :

If the robot is closer than 30 cm it will go back, or **else** it will go forwards.

6. **Put this program in a loop so that the robot moves backwards and forwards forever.**

Task 14. Edge Follower Challenge #2

You can use the **Switch** block in a loop to make a your robot follow a black line. The process can be described in word like this:

Do this forever:

    If the sensor is on black, keep turning to the right.

    If the sensor is on white, keep turning to the left.

Write a program which uses a Switch **block to make the robot follow a black line.**

The following tips may be helpful.

- The safest way is to use just use one wheel when making each turn.
- You will need to stop the other wheel **first** or it will keep on moving.
- Lower power settings will give you a more reliable line follower. Fast robots make mistakes by overshooting the line, especially on curves. Try Power of 60.

* * * * *

Thankyou for working through this book. I hope you had fun.

Now you are in a position to think of your own robot tasks and program solutions to them.

If you want some follow-on advanced worksheets, please email me using the address below.

Sandy Garner

Department of Computer Science

University of Otago

Dunedin

New Zealand

sandy@cs.otago.ac.nz