Table of Contents

Chapter 1: Introduction 1
Chapter 2: miniVEX Basics 4
Chapter 3: What is a Robot? 20
Chapter 4: Flowcharting 25
Chapter 5: How Far? 28
Chapter 6: How Fast? 32
Chapter 7: How Many Sides? 36
Chapter 8: Help, I'm Stuck! 41
Chapter 9: Let's go Prospecting! 46
Chapter 10: Stay Away from the Edge 49
Chapter 11: Prospecting and Staying Safe 52
Chapter 12: Going Up and Going Down 54
Chapter 13: Landing Area Preparation 57
Chapter 14: Touch LED and Bumper Switch 62
Chapter 15: Meet your Adoring Public! 64
Chapter 16: As seen on TV! 66
Chapter 17: Mini-Golf 67
Chapter 18: Dancing Robots 70
Chapter 19: Robot Wave 73
Chapter 20: Robot Butler 76
Appendix 1: Student Worksheets 78
Appendix 2: Building Instructions 101

Chapter 1: Introduction

This book is a guide for teachers implementing a robotics course in the classroom. It is aimed at middle years schooling (ages 9 - 15) but the wide range of activities can be adapted to suit older or younger students. The book is based around the VEX IQ platform, and more specifically a single robot design, the miniVEX, which is used in all activities. Using a single simple robot design approach is valuable in resource limited classrooms, as it allows the teacher to work with a 'standard' robot, rather than using valuable classroom time building and breaking down unique robots each lesson. The miniVEX design can be found at the back of the book, as well as being freely available online – www.damienkee.com. Please send me an email and let me know if you are using the design!

All activities can be completed with the following VEX IQ kits

- VEX IQ Starter Kit with Smart Sensors (228 – 3080)
- VEX IQ Super Kit (228 – 2500)
- VEX IQ Starter Kit with Controller + Distance Sensor, Colour Sensor, 2 x Touch LED

There are several different software languages that can be used with the VEX IQ kits, this book focusses on the ROBOTC Graphical language by Robomatter. This can be downloaded from www.robotC.net.

While the VEX IQ Controller is very useful, this book will focus on learning how to program the robot to behave in a purely autonomous manner. The controller is not necessary for any of these activities.

The book is divided into sections that follow a 10-week course, although this can be modified to suit the needs of the teacher. The first 6 weeks takes students through a series of activities, progressively exposing them to new aspects of the miniVEX programming environment. Following is a set of open ended challenges from which teachers may pick and choose to suit their particular class.

All challenges follow a similar structure:

- Scenario setup + background information. Teachers are free to develop each scenario further as they see fit.
- Equipment list. Aside from the standard VEX IQ robotics kit, all other required resources are easily sourced within a school environment.
- Teacher notes are provided on common issues that may arise with each challenge and how they are best dealt with.
- Programming examples in the ROBOTC Graphical software environment for VEX IQ.
- Student worksheets to fill out (photocopy / print permission is provided).
- Extension activities.

Chapter 2: miniVEX Basics

Overview: Build a robot that is capable of driving around an obstacle course.

Project: NASA is in the market for a new planetary rover to explore the recently discover planet Zezuno. You are required to construct and test a robot that is capable of following a set of commands to explore the planet's surface. Before the robot is deployed, it must be extensively tested to ensure it will perform as expected. You can't fly a technician to Zezuno to reboot the robot!

Equipment required

- 1 VEX IQ robot kit per group
- 1 computer per group
- Masking tape and Tape measure

Teachers' Notes

This section will cover the following topics amongst others

- Basic numeracy
- Decimal and fractional numbers
- Conversion between millimetres and inches

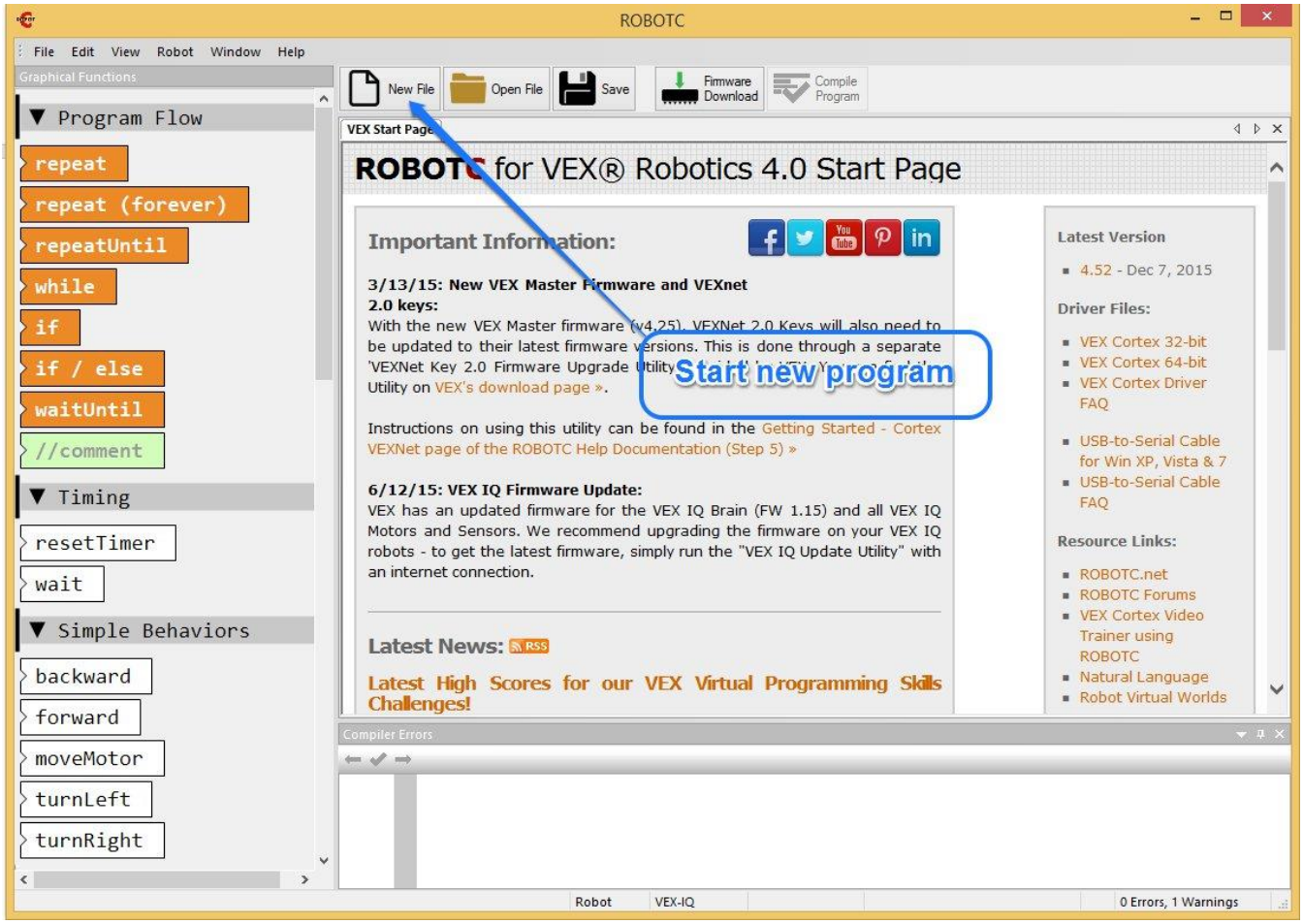
Get the students to build the miniVEX robot presented in Building Instructions.

Photocopy and hand out Student Worksheet – miniVEX Basics. This worksheet gives the students a range of different activities to follow that progressively increase in difficulty. To make our robot move, we need to send instructions to the Smart Motors which in turn drive the wheels. The miniVEX design is often referred to as a wheelchair configuration, as it has a Left and Right Smart Motor that allows the robot to drive forwards, backwards and make turns.

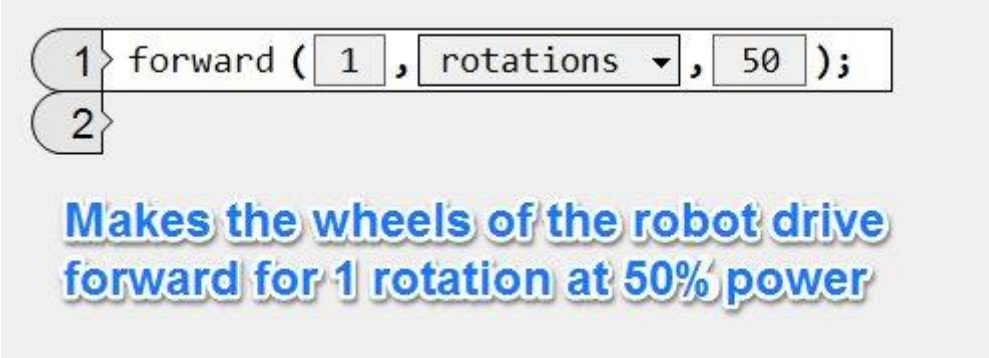
ROBOTC Graphical

The software we'll be using throughout this book is ROBOTC Graphical from Robomatter. ROBOTC Graphical is available within all versions of ROBOTC from version 4 and above.

When you open ROBOTC Graphical, you'll see a screen similar to the following (your version number and 'latest news' may be different). Click 'New File' to start a new program.

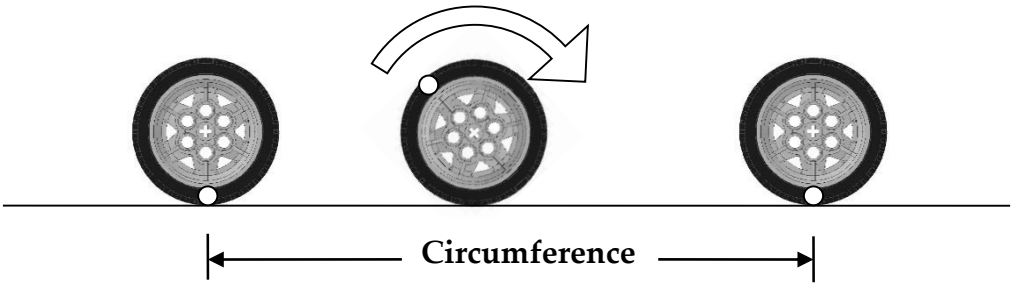


So what happened? If everything went to plan, probably not much at all. Let's look at the program and see what we told the robot to do.



Run the program again and watch the wheels very carefully. You should notice that both wheels turned exactly 1 rotation causing the robot to drive forward a short distance. How far is this 'short distance' however? There are a few ways we can figure this out.

Experimentally: Take a wheel off the robot and make a mark on the tire with either chalk or masking tape. Create a starting mark on the table and line up our tire mark with it. Now slowly roll the wheel until the tire mark again touches the ground. Make another mark at this point and use a ruler to measure the distance.



Mathematically: The circumference of a wheel can be calculated using the formulae: $c=\pi \times d$

Where c= circumference, $\pi = 3.14$ (approx) and d is the diameter of the wheel.

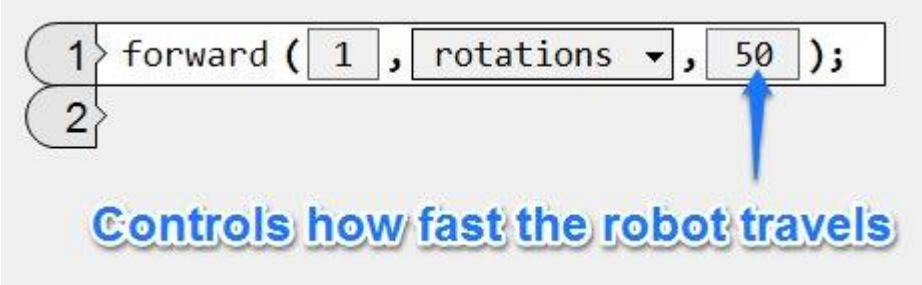
The wheel that comes as part of the standard VEX IQ set is 63.7mm in diameter which results in a circumference of 200mm.

Observationally: If you look at the side wall of the VEX IQ tire, you'll see that it has a 'Travel' of 200mm.

This means, that for one complete rotation of the wheel, the robot will travel 200mm.

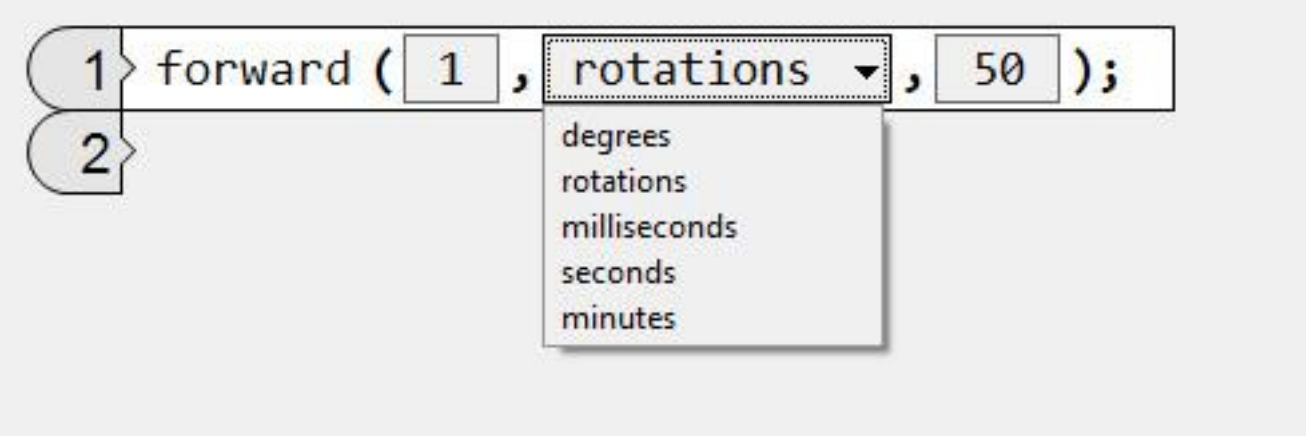
Power Level

As well as controlling how far the robot travels, it is also possible to control how fast it goes. The last number in the *forward* instruction specifies the power level to be applied to the robot. This can be from 0 – 100.



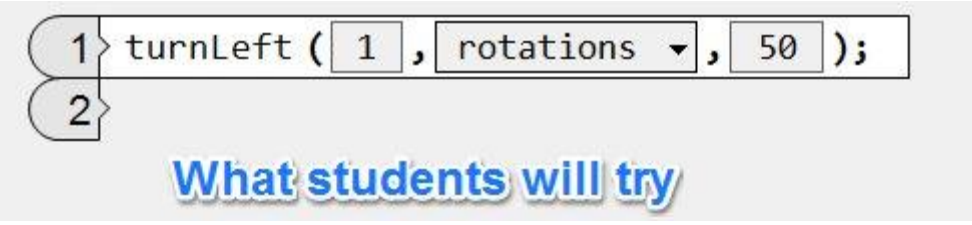
Additional Modes

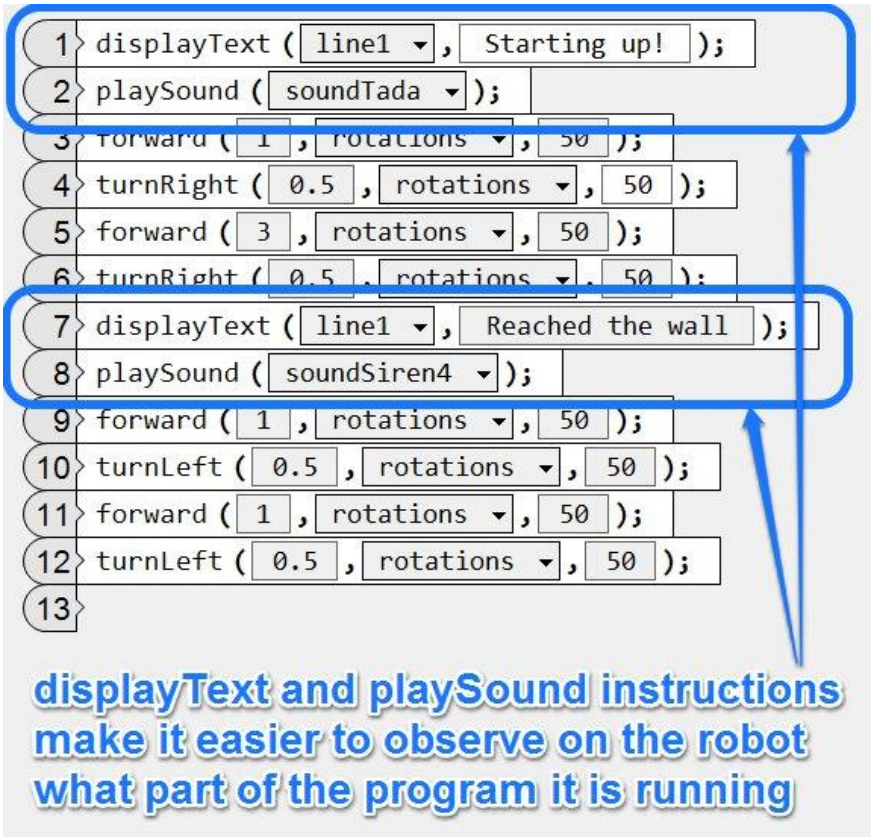
While this example shows how to control the number of rotations the wheels do, ROBOTC Graphical can also control other properties including Degrees, Rotations, Milliseconds, Second and Minutes. The one chosen will depend on the challenge being attempted.



Making your robot turn

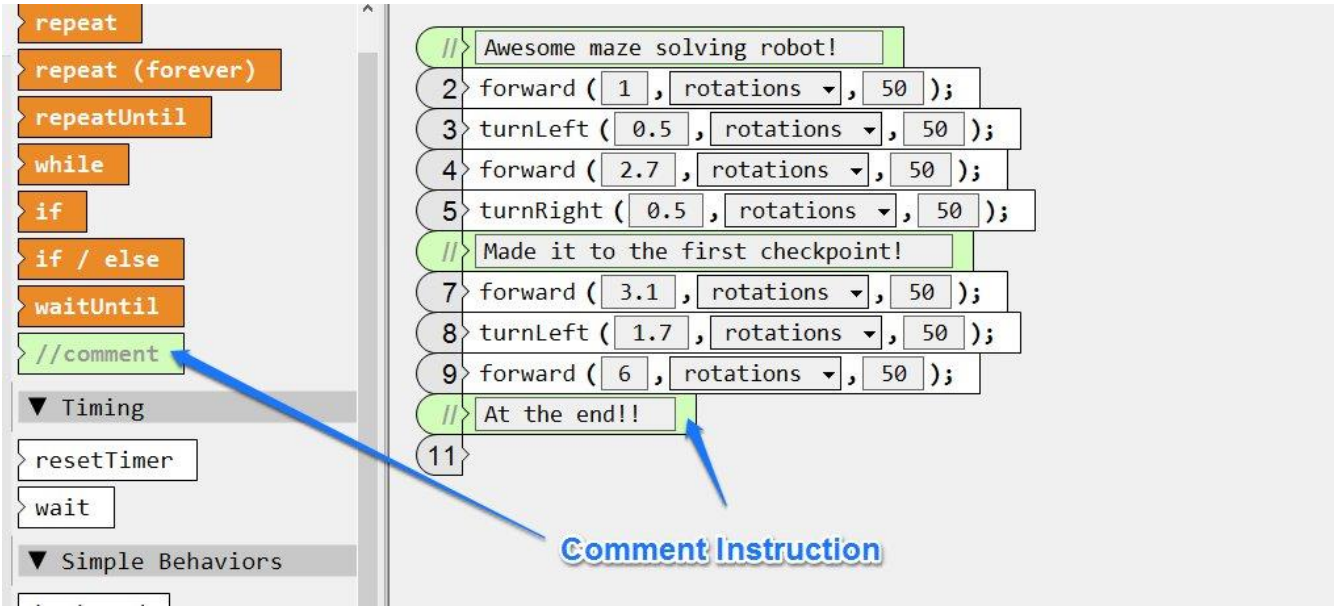
For the challenge where students are asked to make their robots turn around in a full circle (360 degrees), they will typically type in 1 rotation and proceed to run the program. When run however, if they are using the miniVEX, they will find that their robot does not actually turn 360 degrees but in fact far less.

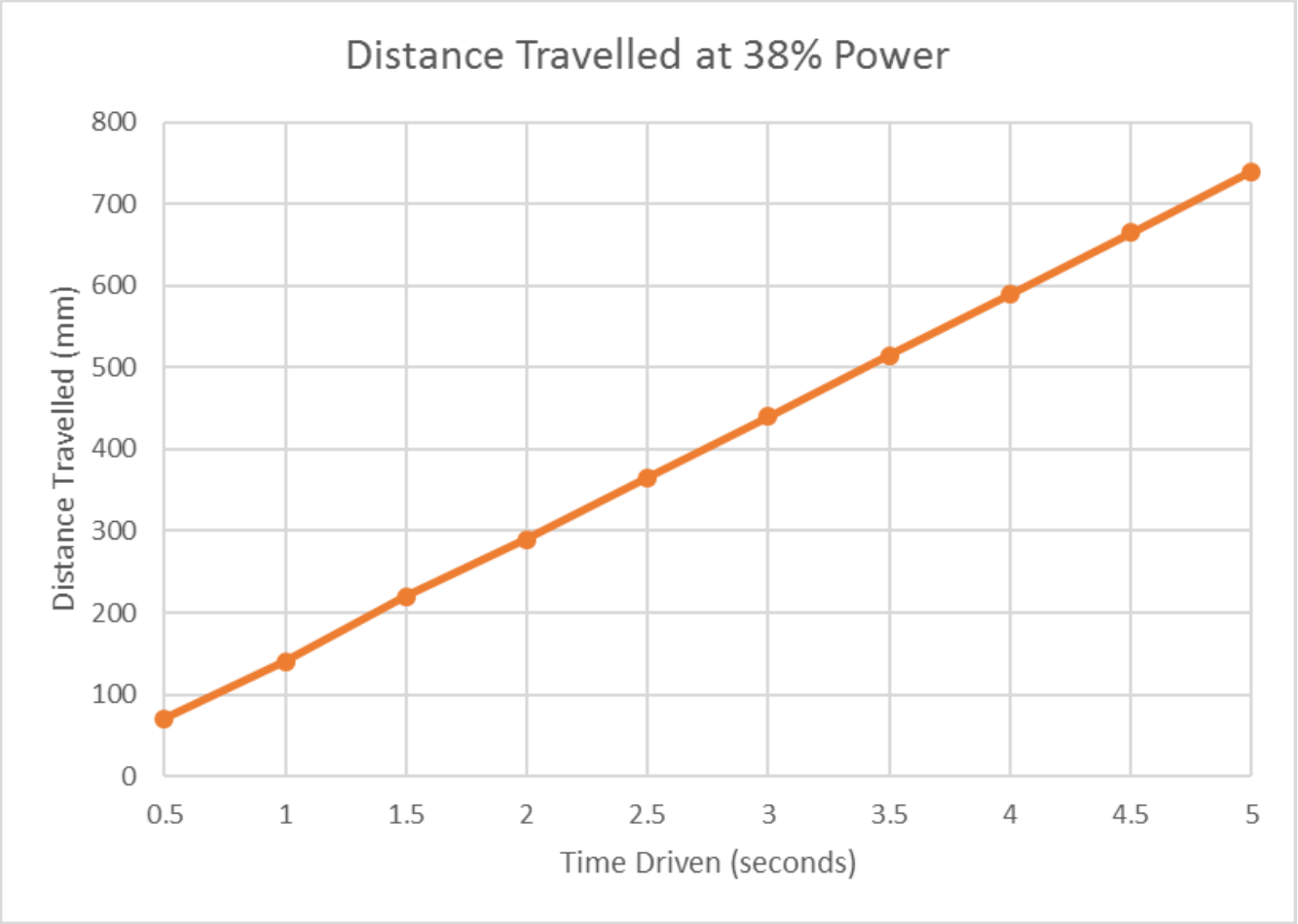




ROBOTC Graphical 'Commenting'

'Commenting' is a practice that programmers used to debug and annotate their code. A 'comment' is totally ignored by the robot, and often serves as a great way to put in a readable description of what the program is trying to achieve. This is especially useful when you have a complex piece of code, and have to come back to it after a few days. The 'comment' can serve as a great way to remind you of what the program was attempting to do. Notice the // in place of the line number, in front of the description, which denotes the comment.





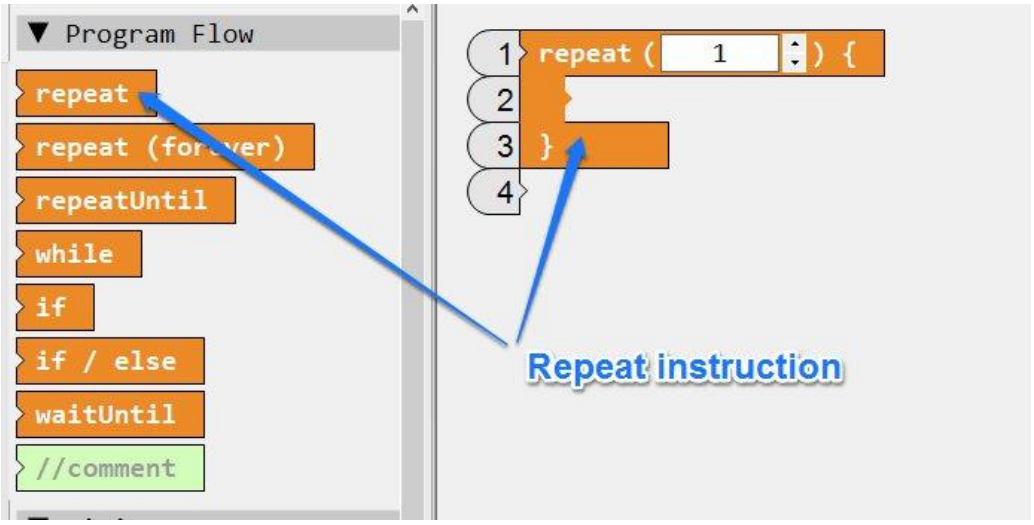
Look out for...

To set up, we will need a starting line and a tape measure. Encourage the students to work out what materials they will need and gently push them in the right direction if they miss out on anything.

If this same experiment is run on carpet, students may see a decrease in the distance of the robot due to the additional friction between the castor wheel and the carpet surface. Encourage this line of thought and get the students to run the same experiment with the same power level on multiple surfaces.

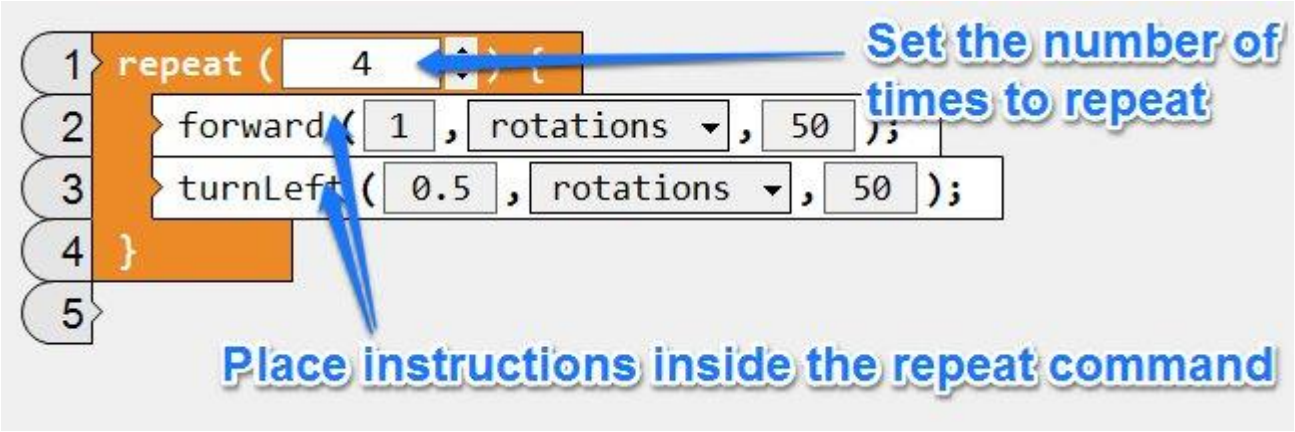
Be wary of very high power levels. Depending on the surface the robot is travelling on, sometimes the wheels can 'slip' as they start up, giving a slightly shorter distance than should be measured.

Drag a **Repeat** instruction into the code editor. You'll notice that this is not just a single line of an instruction, but rather a set of instructions that wrap around a line (in this case instruction line 2).



What this allows us to do is place different instructions inside the **Repeat** instruction.

Drag a Forward and Turn Left instruction and place them inside the **Repeat** instruction. The **Repeat** instruction will expand to take as many instructions as necessary. Set the Repeat to run a total of 4 times.



Theory

Cumulative error becomes a factor when using the **Repeat** instruction, as errors in the turn angle can build up every time the robot performs the turn. Eg. Let us assume the robot turns 92 degrees instead of 90 degrees. To the human eye, the first turn looks fine, but as the robot progresses around the square, the 2 degree error from each turn is accumulated into an 8 degree error at the end of the square.

IF the Colour Sensor sees RED
Stop the Smart Motors
Play a sound
Drive off the RED
IF the Colour Sensor detects the edge of the plateau (ie. Small Proximity reading)
Backup
Turn Slightly
ELSE
Drive Forward

Example Program