The Smallpeice Trust

ENGINEERING @HOME

09

The Robot Challenge

Including Robot Recycle VEXcode VR PROGRAMMING CHALLENGE & COMPETITION

#EngineeringAtHome

Suitable for ages:	Time needed:
8+	1hr

smallpeice
Dare to imagine

Curriculum links: Maths – shapes, measurement, coding; Science – materials; D&T – design, make, evaluate

Skills learnt: design, building, testing, evaluation, coding

smallpeice
Dare to imagine

Since our Smallpeice team can't visit schools, we've decided to challenge each other to make a robot and program one virtually.

09

Learning Objectives

Create purposeful, functional and appealing designs

Select from a wide range of materials and use tools to perform practical tasks

Build structures, exploring how they can be made stronger and more stable

Evaluate your ideas and products against design criteria

Topics Covered

HOW ROBOTS WORK
https://bit.ly/36oAzzV

WHAT IS CODING
https://bit.ly/2LMZ1I5

VEX VR TUTORIALS
https://bit.ly/3bYy2hi

WHAT MATERIALS TO USE

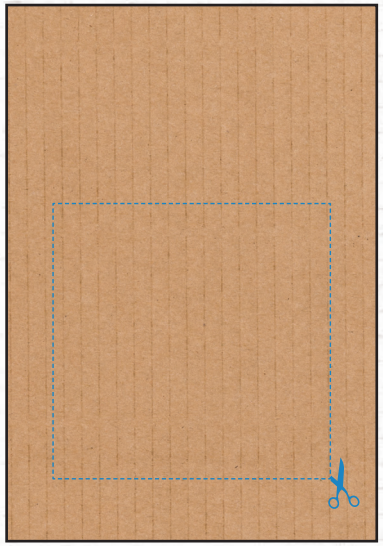
You can use cardboard, plastic, wood, or anything else that works well and you can get at home.

Try looking in your recycling box.

HERE'S WHAT WE USED:

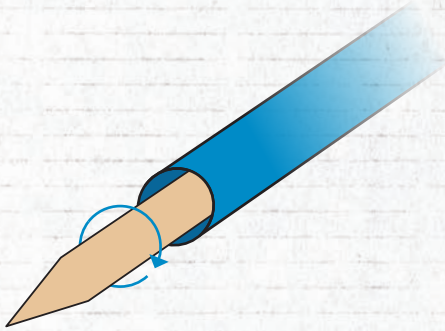
1. 2 CARDBOARD TUBES
2. CARDBOARD
3. STRAWS
4. EGG BOXES
5. WOODEN STICKS
6. MILK BOTTLE TOPS
7. FRIDGE MAGNET
8. SCISSORS
9. STRING
10. TAPE

INSTRUCTIONS



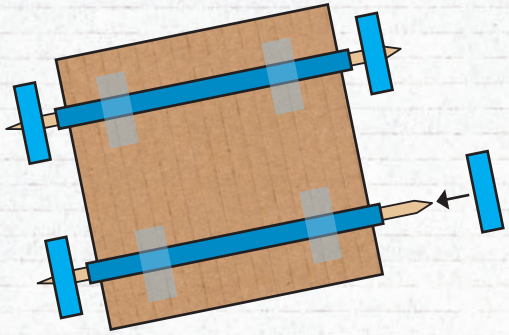
1.

Make a base for your robot by cutting out a cardboard square.



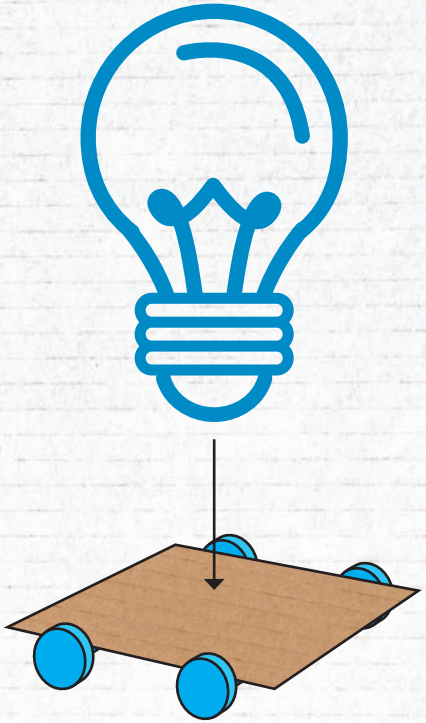
2.

Make your wheelbase, it's important that your "axle" can freely spin so thread a wooden stick through a straw which is stuck to the bottom of your base.



3.

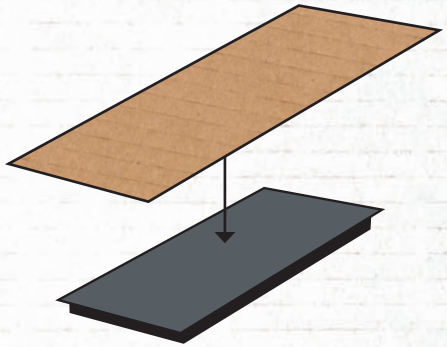
Attach your wheels to your axle – it's best to use something already round like bottle tops but if you are cutting them out of cardboard make sure they are as round as possible.



4.

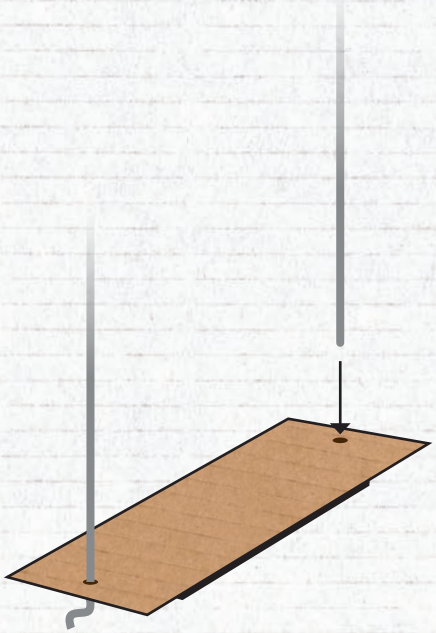
Time to make the body now, use your imagination! Make it look however you like: the mars rover; humanoid; an animal.

INSTRUCTIONS



5.

If you're using a winch to pick up objects, then attach your magnet to a strip of cardboard.



6.

Attach string to the end of the cardboard (so it looks like a swing).

7.

Connect the string to a stick that is that is attached to your robot (maybe using a straw like the wheels) which can freely turn when you want to wind your magnet up or down.

PROGRAMMING CHALLENGE

Getting started with VEXcode VR

VEXcode VR runs in your web browser and will work on almost any device – PC, Mac, tablets, Chromebooks etc.

The following browsers have been tested with VEXcode VR:

Windows:

Google Chrome	✓
Mozilla Firefox	✓
Microsoft Edge (Chromium only)	✓

MacOS:

Apple Safari	✓
Google Chrome	✓
Mozilla Firefox	✓

iOS:

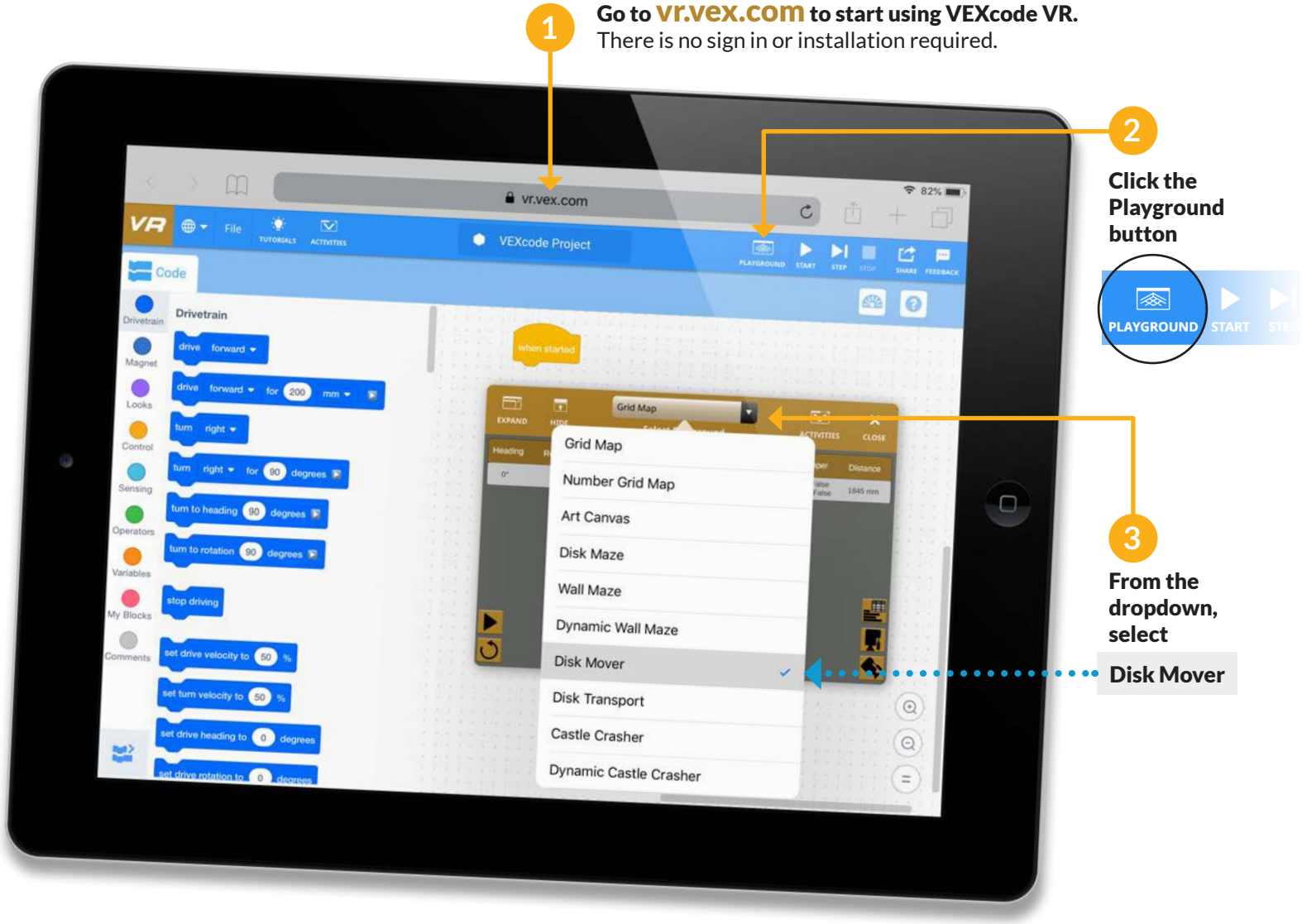
Apple Safari	✓

Android:

Google Chrome	✓

ChromeOS:

Google Chrome	✓



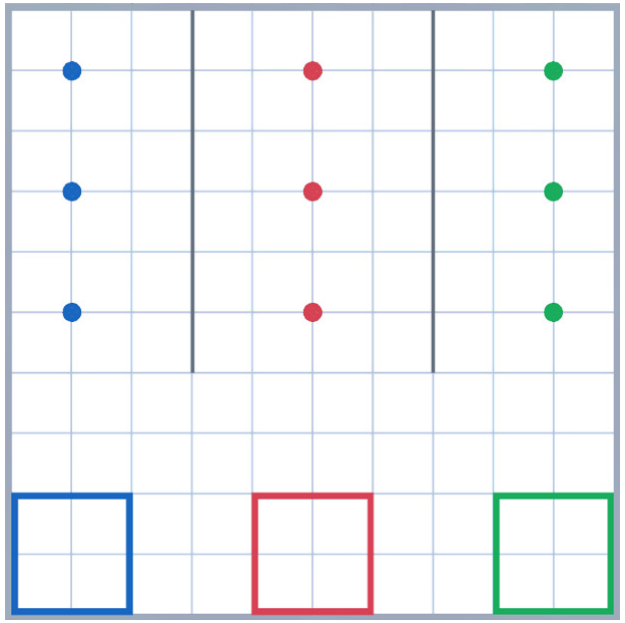
PROGRAMMING CHALLENGE

Objective

Using the VEXcode VR virtual robot, collect the pieces of recycling and place them into the recycling bins.

There are 9 pieces of recycling to collect

3 BLUE **3 RED** **3 GREEN**



There are 3 recycling bins

1 BLUE **1 RED** **1 GREEN**

Robot Recycle
VEXcode VR Challenge

Each recycling bin is divided into 4 compartments

1	2
3	4

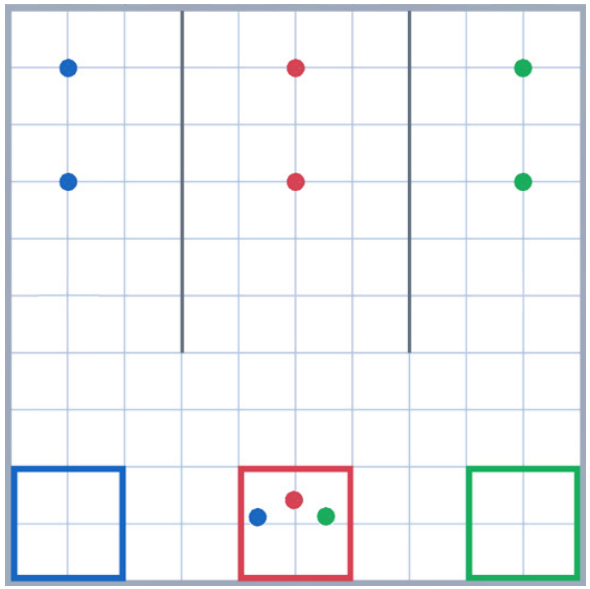
An introduction to using VEXcode VR for The Smallpeice Trust Engineering@Home Robot Challenge can be found at:

https://youtu.be/UfDIgxTExB8

PROGRAMMING CHALLENGE

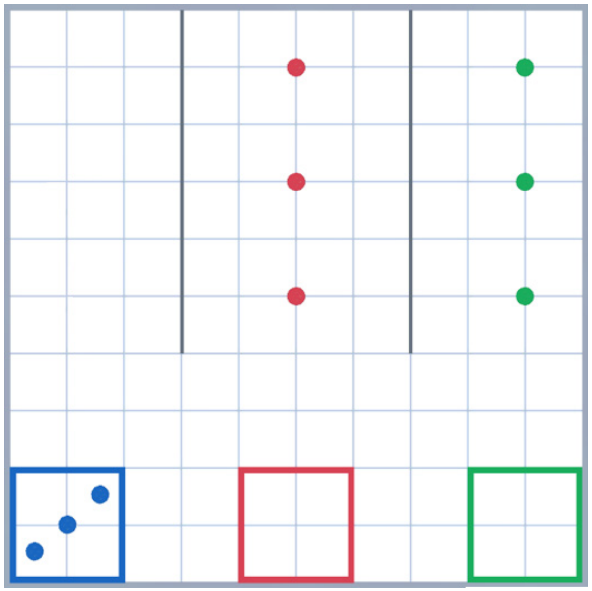
Tasks

There are 3 tasks to complete which get progressively more challenging:



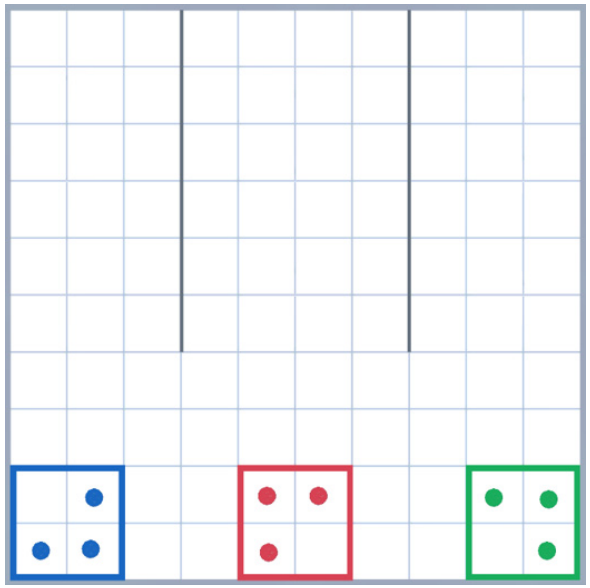
Level 1

Collect three pieces of recycling of any colour and place them into any of the recycling bins. You do not need to put the recycling in the bin of the same colour.



Level 2

Collect three pieces of recycling of the same colour and place them in the correct colour recycling bin.

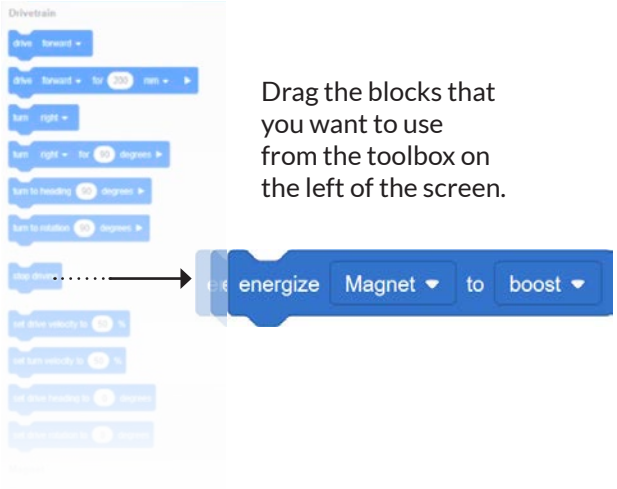


Level 3

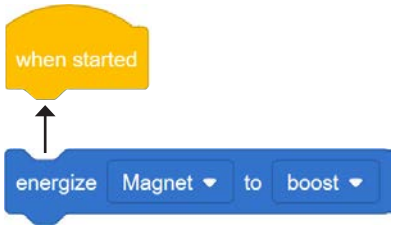
Collect all nine pieces of recycling and place them into the correct coloured recycling bin. Each recycling bin should only have one piece of recycling per compartment.

PROGRAMMING CHALLENGE

Creating a program



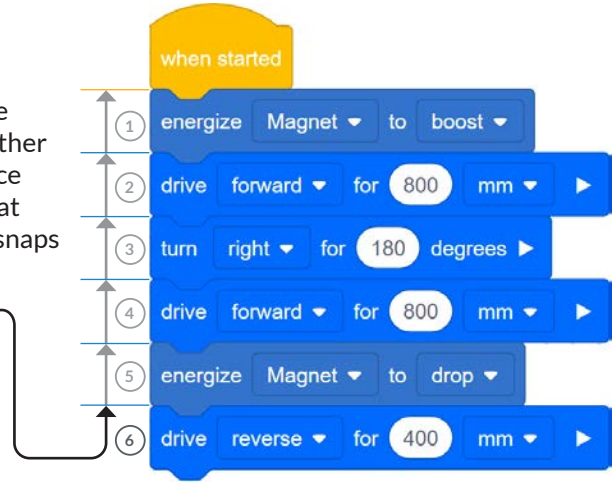
Drag the blocks that you want to use from the toolbox on the left of the screen.



Make sure the first block in your program is connected to the yellow '**when started**' block.

EXAMPLE PROGRAM

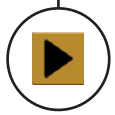
Connect the blocks together in a sequence ensuring that each block snaps to the one before it.



Running a program

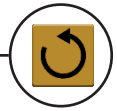
Click the **START** button in the blue band at the top of the screen

OR



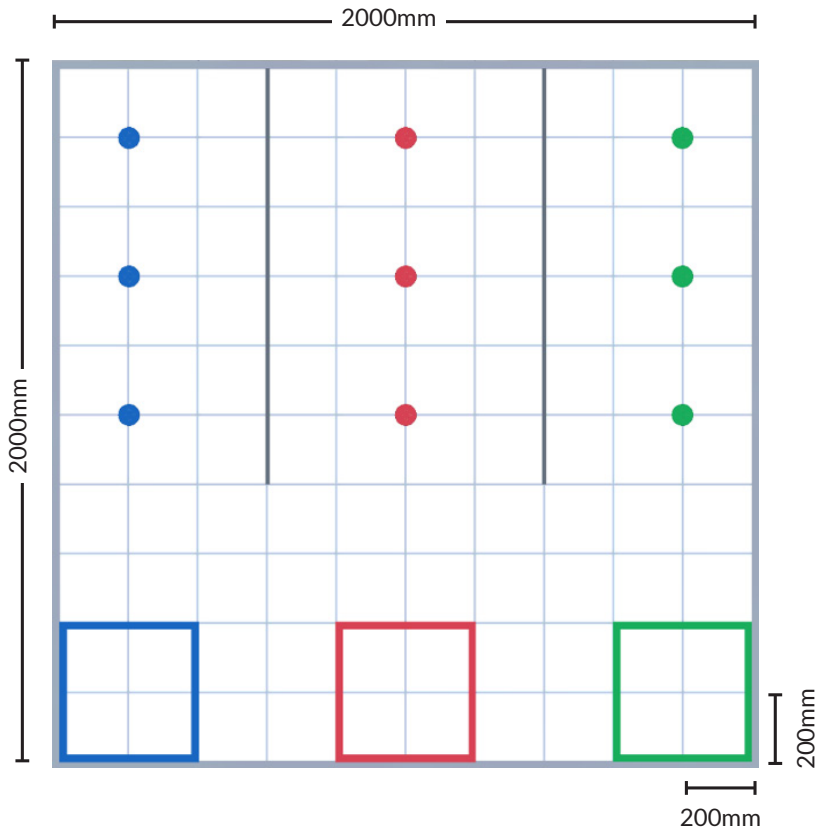
within the Playground window to run the program

To **RESET** the Playground to try again, click the reset button



PROGRAMMING CHALLENGE

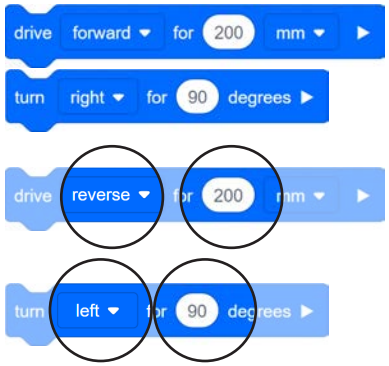
Tips



The Playground is 2000 x 2000mm and is divided into a grid of 200mm squares.
You can use these gridlines to help you work out how far your robot needs to travel.

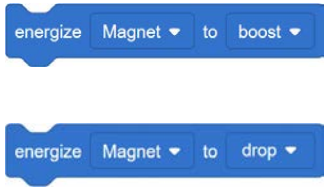
The easiest way to make your robot move is use the drive forward for 200 mm and the turn right for 90 degrees blocks.

You can use the dropdown to change from forward to reverse or from right to left. You can also enter any value for the distance you want to travel or the number of degrees you want to turn



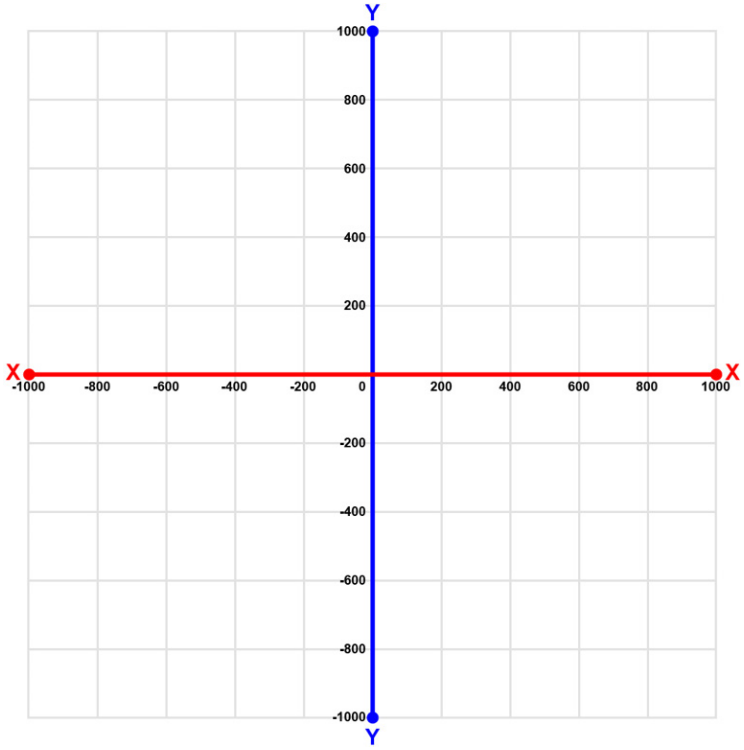
To collect a piece of recycling, use the energize magnet to boost command before you drive over the item that you want to collect.

To release it, use the energize magnet to drop command.



PROGRAMMING CHALLENGE

Tips



You can use coordinates to improve the accuracy of your navigation.
The centre of the Playground is coordinates **X0** and **Y0**

The **position X in mm** and **position Y in mm** sensing blocks can be used to help you navigate to specific positions on the field. Because this means you can use variables or lists to store coordinates of the locations you need to travel to, you can make your code more efficient by reusing certain algorithms.

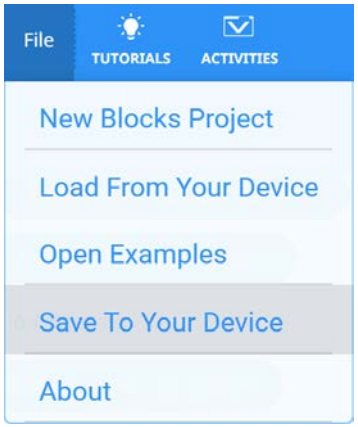
PROGRAMMING CHALLENGE

Tips

Save your program regularly to ensure you don't lose your code.

Name your program using the box in the top centre of the screen:

Save and load programs using the file menu:



NEED A CHALLENGE?

If you complete your robot and want to challenge yourself further:

1. Decorate your robot in the most imaginative way possible
2. Complete all three challenges on the coding side
3. Create a video about why you made design choices and what you completed on the VEX VR website
4. Try creating a different kind of pick-up mechanism
5. Film a video of your robot in action and send it to us!

Once you've got your robot performing at its optimum, film it in action and share your video on:

www.facebook.com/TheSmallpeiceTrust

www.twitter.com/SmallpeiceTrust
Use the hashtag **#EngineeringAtHome**

www.instagram.com/TheSmallpeiceTrust

COMPETITION TIME Click to view T&C

Best Design:

Send a picture/video to **@SmallpeiceTrust** with **#EngineeringAtHome**

Solving the Problem:

Email your code completing all 3 of the VEX VR challenges to **ukcode@vexrobotics.com**

Best Video:

Create a video showing us your design, telling us why you chose to make it like that and telling us what you achieved and enjoyed in the VEX VR challenge. Tweet it to **@SmallpeiceTrust**

Computing Programmes of Study Curriculum Links

KEY STAGE 2

Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts

Use sequence, selection, and repetition in programs; work with variables and various forms of input and output

Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs

KEY STAGE 3

Design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems

Understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem

Use two or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions

Understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming; understand how numbers can be represented in binary, and be able to carry out simple operations on binary numbers [for example, binary addition, and conversion between binary and decimal]