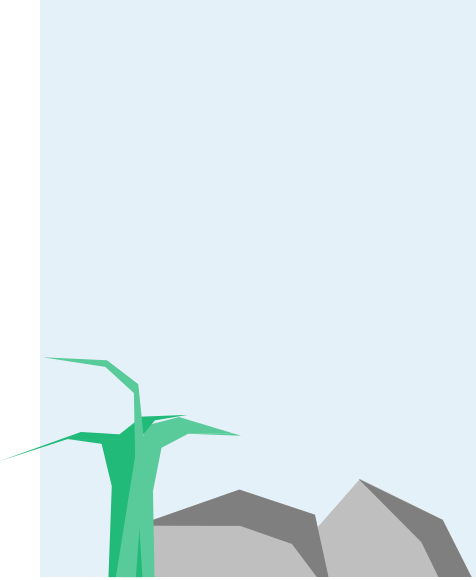
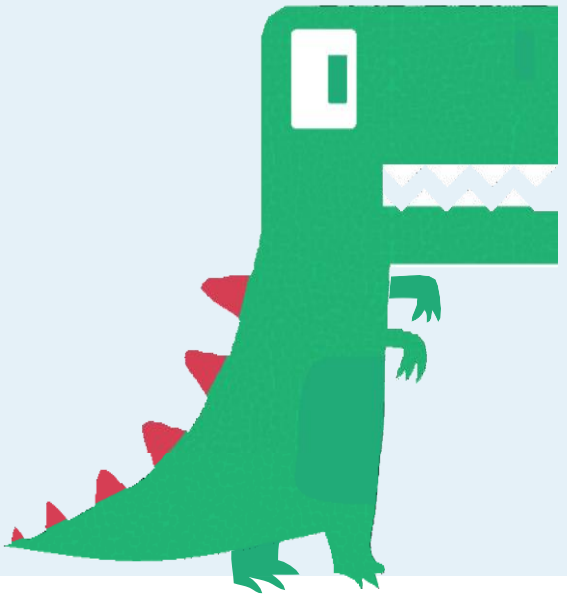
Student workbook

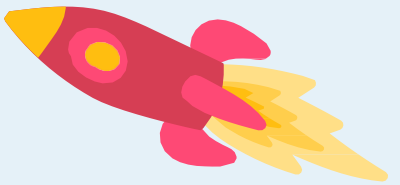
Iteration and selection with VEXcode VR

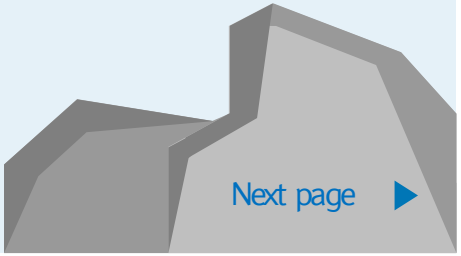
Code Playground

BARCLAYS









Next page ▶

Iteration and selection with VEXcode VR

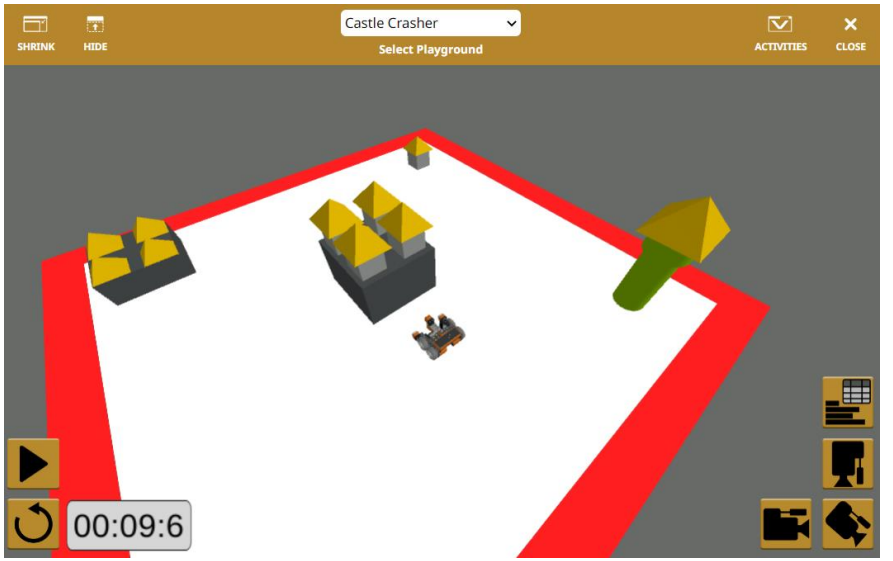
VEXcode VR project

In this session we will learn how to detect objects, using various sensors on the VEXcode VR robot, and to use if/else selection in their code and continue to explore iteration. By completing these challenges you will learn to –

- understand that the forever loop is a type of iteration
- use if/else selection
- know that Boolean data has two states – true and false

This project is designed for the VEXcode VR system, you can access it here - https://vr.vex.com/

Hope you enjoy the project!



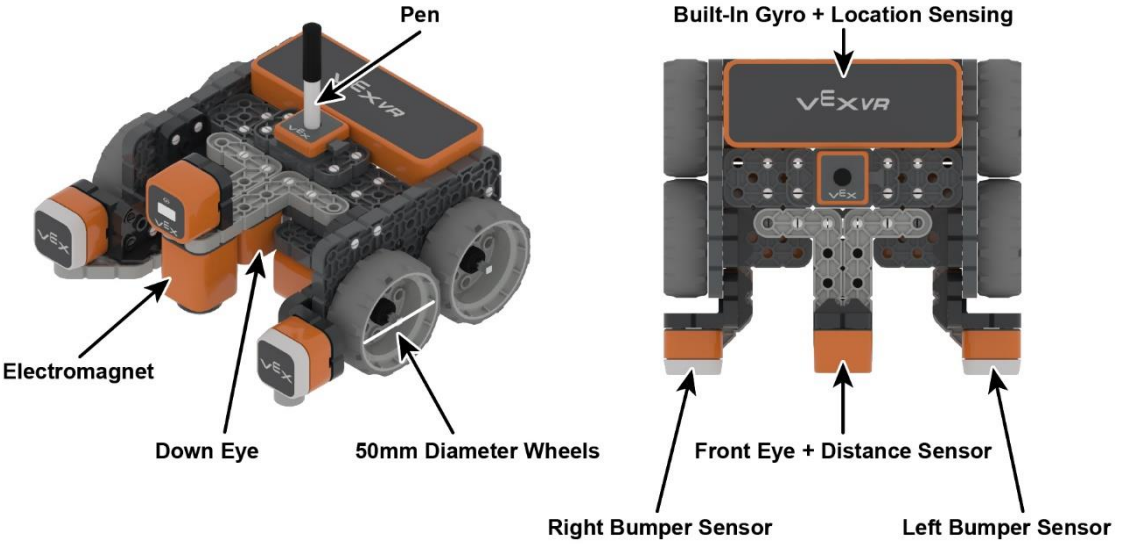
Iteration and selection with VEXcode VR

VEXcode VR project

Just like a real robot, the VEXcode VR robot has various sensors to help it detect objects or help it navigate around the playgrounds.

The Front Eye Sensor can detect objects in front of the robot, tell you how far away from them it is and what colour the objects are.

In this session, you will use the Front Eye Sensor to find objects in the Playground. Some robots will use their sensors to avoid obstacles but, in this task, you will need to crash into them!



Iteration and selection with VEXcode VR

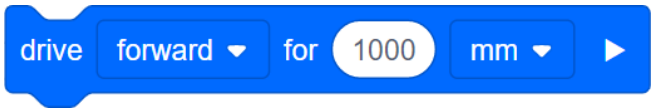
VEXcode VR project

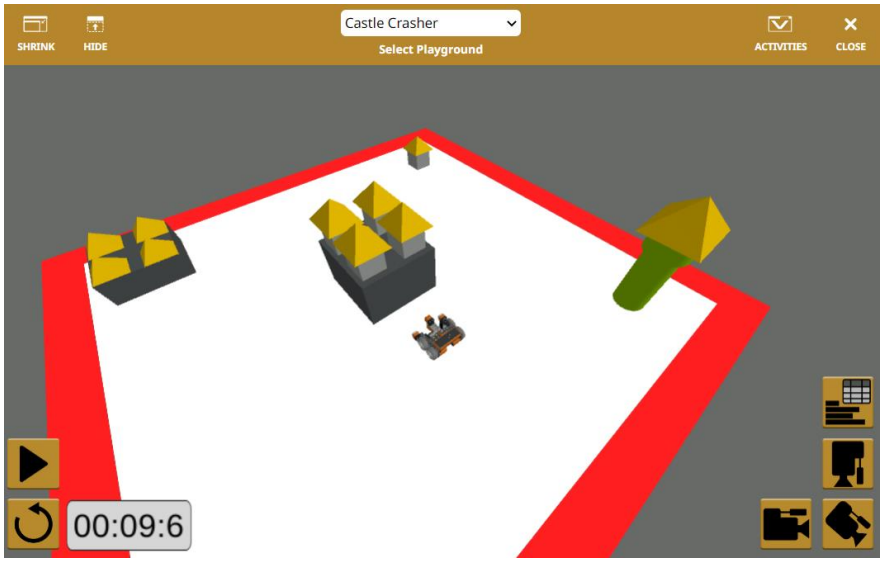
Step 1

Open the Castle Crasher Playground

You are going to code your robot to knock over as many of the pieces of the Castle as possible! Let's start by finding out what happens when you drive into them.

Use the drive forward for block to drive your robot forward 1000mm so it crashes through the Castle. What happens?





Iteration and selection with VEXcode VR

VEXcode VR project

Step 2

Now you need to be able to detect more pieces of the Castle to be able to crash into the next one. For this, you will use the Distance Found an Object block which you can find in the light blue Sensing menu in the toolbox.

This shape block is called a Boolean. This means the answer to the question "has the distance sensor found an object?" can be one of two things:

True – it has found an object
False – it hasn't found an object

Iteration and selection with VEXcode VR

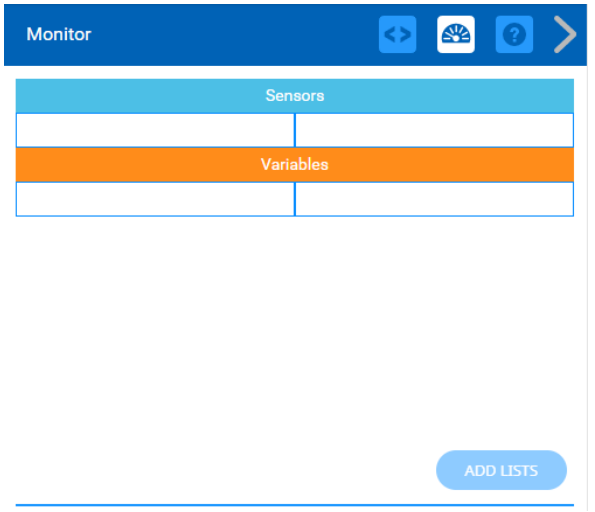
VEXcode VR project

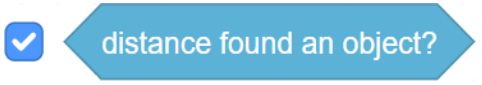
Step 3

To see more about what the robot is seeing, you can use the Monitor. To open the Monitor, click on the speedometer icon in the light blue ribbon.

The Monitor lets us see data from the robot. Go back to the Sensing menu of the toolbox and tick the box next to Distance Found an Object?. The data from this sensor will now show in the monitor.

The sensor is currently returning true because it can see the Castle directly in front of it.





Sensors	
Distance found an object?	true
Variables	

**Top tip – try turning your robot 45 degrees to the right using the turn for block. This makes the sensor return false because it can no longer see an object.

Iteration and selection with VEXcode VR

VEXcode VR project

Step 4

Now we need to tell the robot what to do when it finds an object and also what to do when it doesn't find an object. This is called selection because we are selecting what to do

Go to the yellow Control menu in the toolbox and find the block that has if then else.

You can place the Distance Found an Object inside the six sided hole in the if then else block.

Now get the Forever loop from the yellow Control menu and put the if then else inside it.

This code says:

If the robot finds an object, then

do something

otherwise (else)

do something else.

We put it in a forever loop so the robot keeps checking for objects all the time.

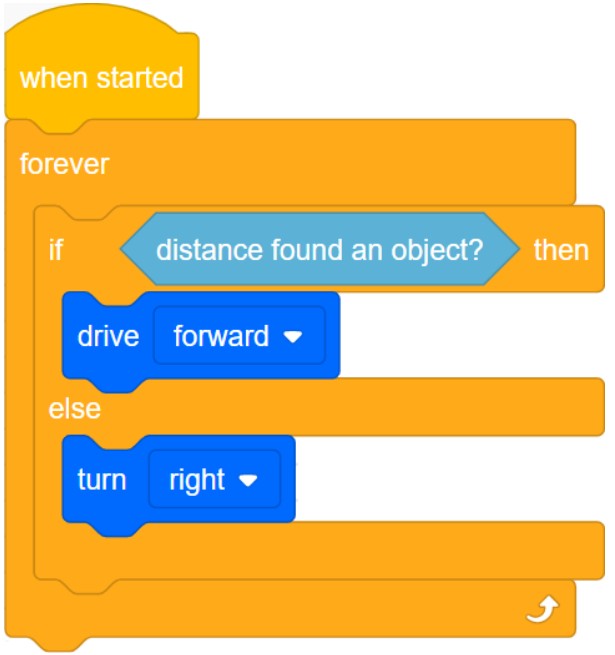
Iteration and selection with VEXcode VR

VEXcode VR project

Step 5

To finish the code, you will need to use the drive forward and turn right blocks.

These are different to the ones you have used before because they do not have any parameters. This means the robot keeps driving until we tell it to do something else.



Iteration and selection with VEXcode VR

VEXcode VR project

Level up!

- Could you add velocity blocks to make the robot go faster or slower? What impact do you think that will have?
- Can you make the robot stop before crashing into the castle instead?
- Can you track the movement of your robot using the pen function?

Notes