CoderZ at Home – Parent Guide

Codyssey World with LEGO® Education SPIKE™Prime

>>Intro to Using this Guide

The courses in Codyssey World are flexibly designed so parents can choose the level of involvement in accompanying their child's learning - from hands-on to completely hands-off. Parents who choose to take a more active role can refer to this guide regularly or as needed to help answer their child's questions and monitor their progress. Each course includes learning objectives and guidance, an introductory overview, reflection questions, and additional sample questions.

>> Overview: Codyssey World

This novice-level curriculum is designed for ages 8 through 11. It introduces kids to Blockly-based coding while integrating basic concepts from math, geometry, science, and engineering. In addition to the hard skills covered in the course materials, the curriculum also blends critical life skills including, problem solving, pattern finding, planning, and persistence!

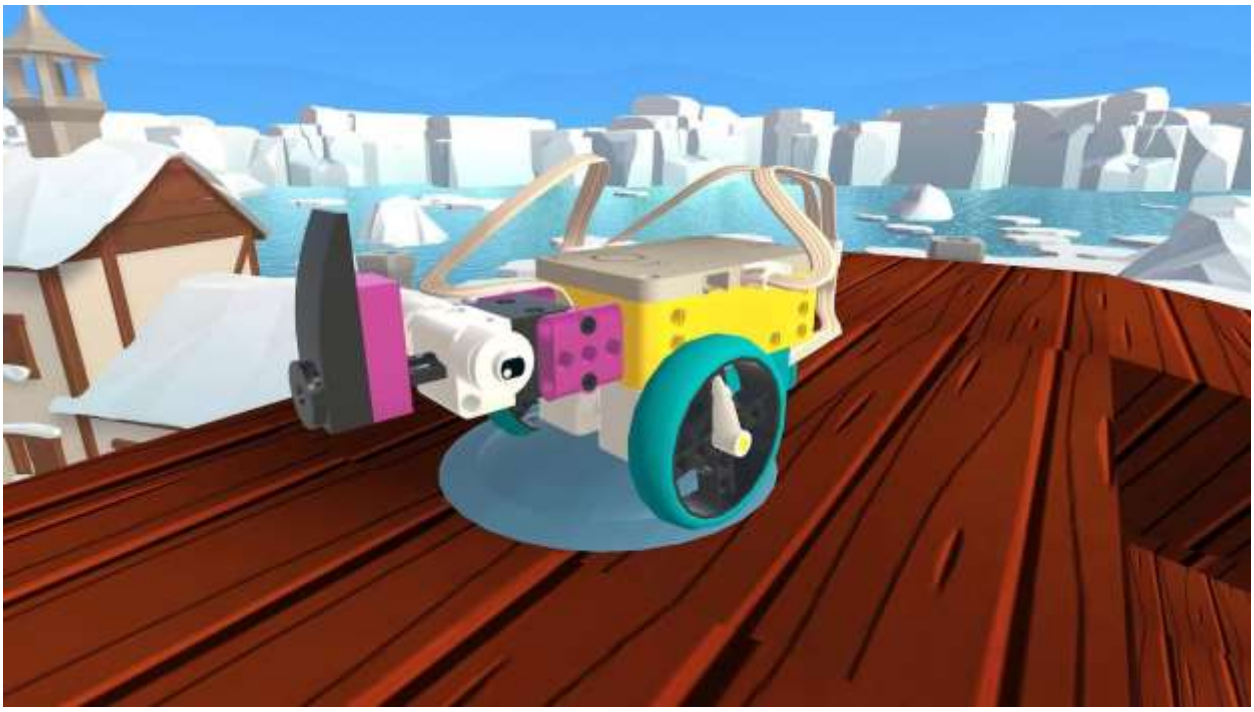
1

Course 1: Adventure Peak

How can code affect the physical world?

Kids are introduced to the CoderZ learning environment and learn basic navigation skills such as driving and turning using Drive and Turn blocks.

2



Course 2: Frozen Island

How can planning help to make better programs?

Kids learn the benefits of planning their algorithm before coding, collecting data on their robot's environment, and using pseudocode.

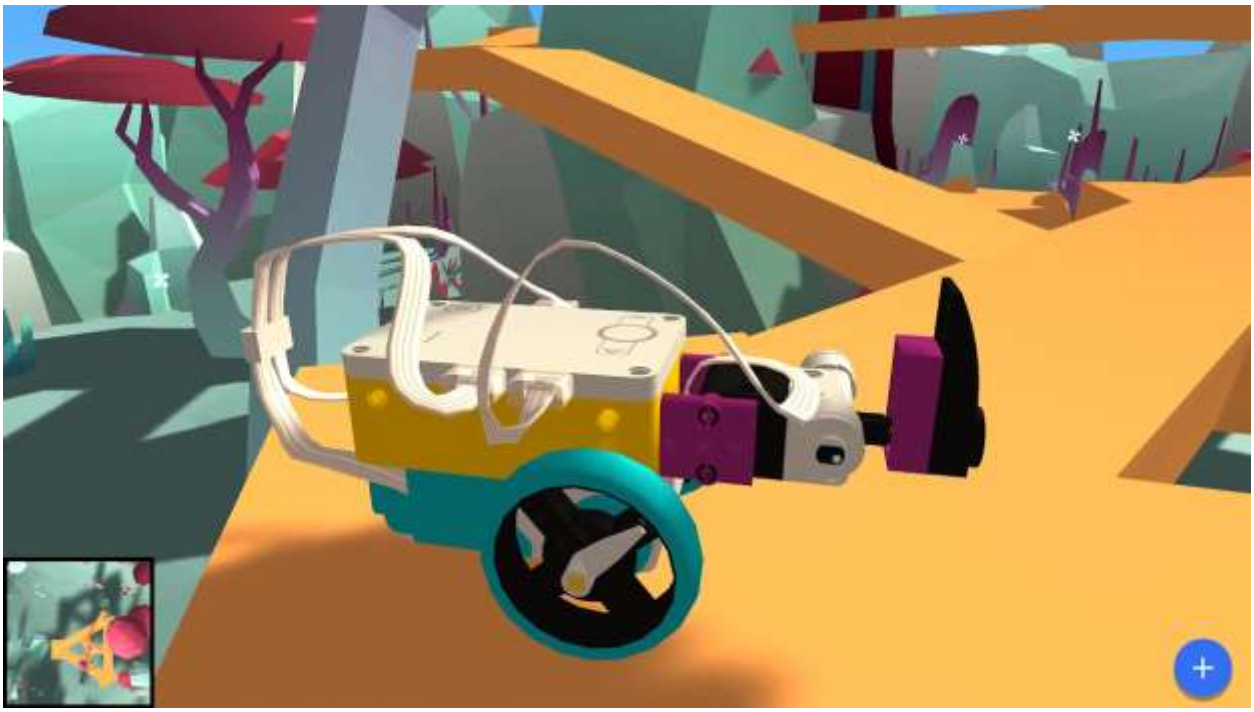
3

Course 3: The Lost City

How can loops make code easier to write and understand?

Kids learn how to utilize and execute loops in coding, using patterns to create complex code more easily.

4

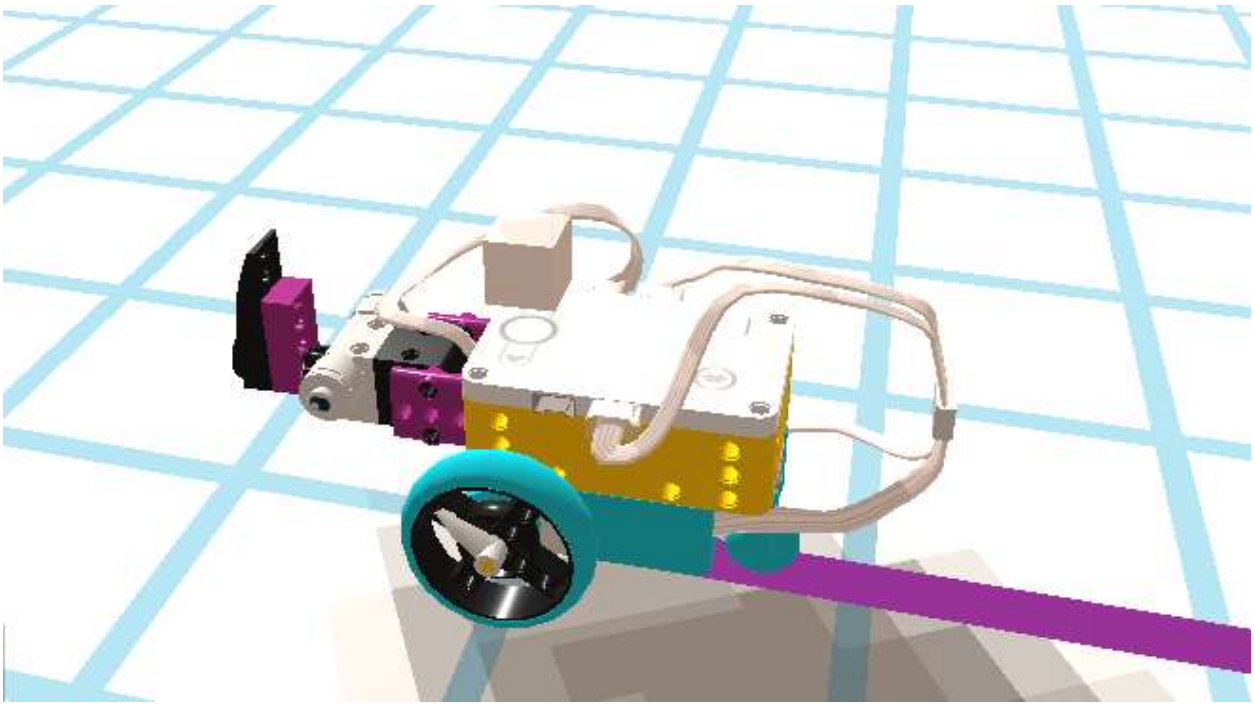


Course 4: Crystal Crater

How can I find and fix problems with a program?

Kids learn how to use parameters to change the behavior of a method as well as test and debug a program.

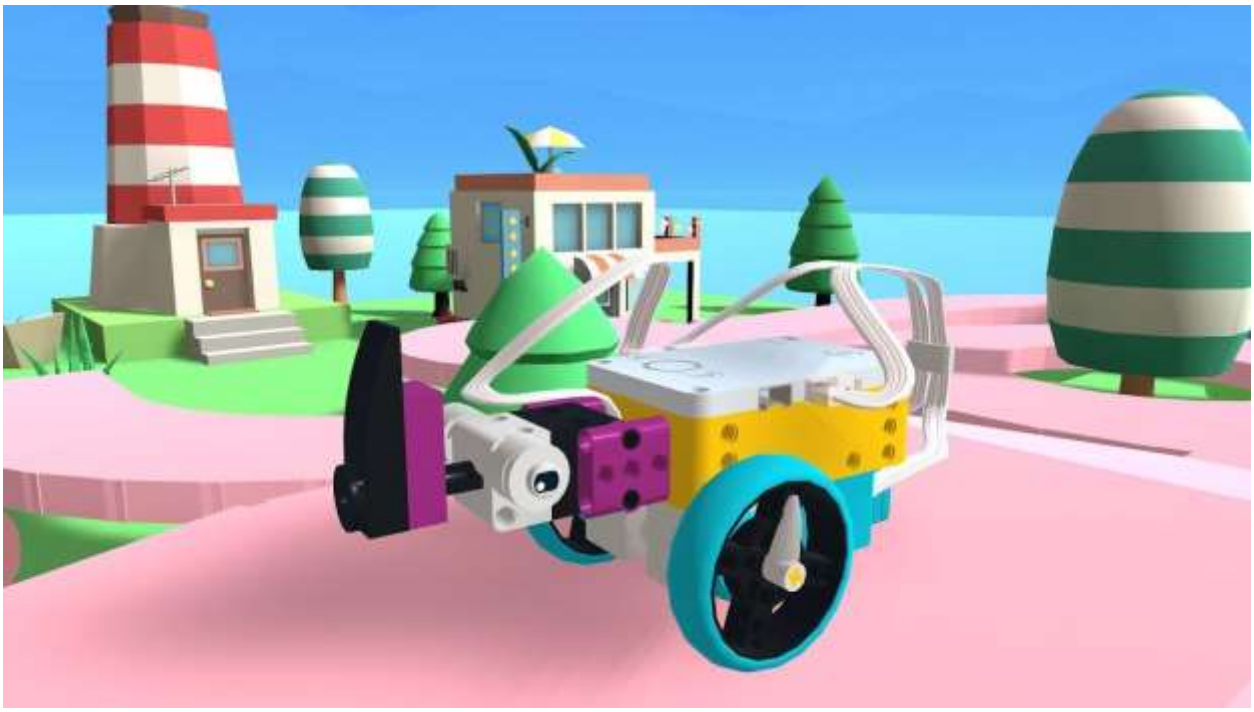
P



Project: The Dancing Robot

Kids use what they have learned to choreograph and code a dance for their robot. **(Optional)**

5

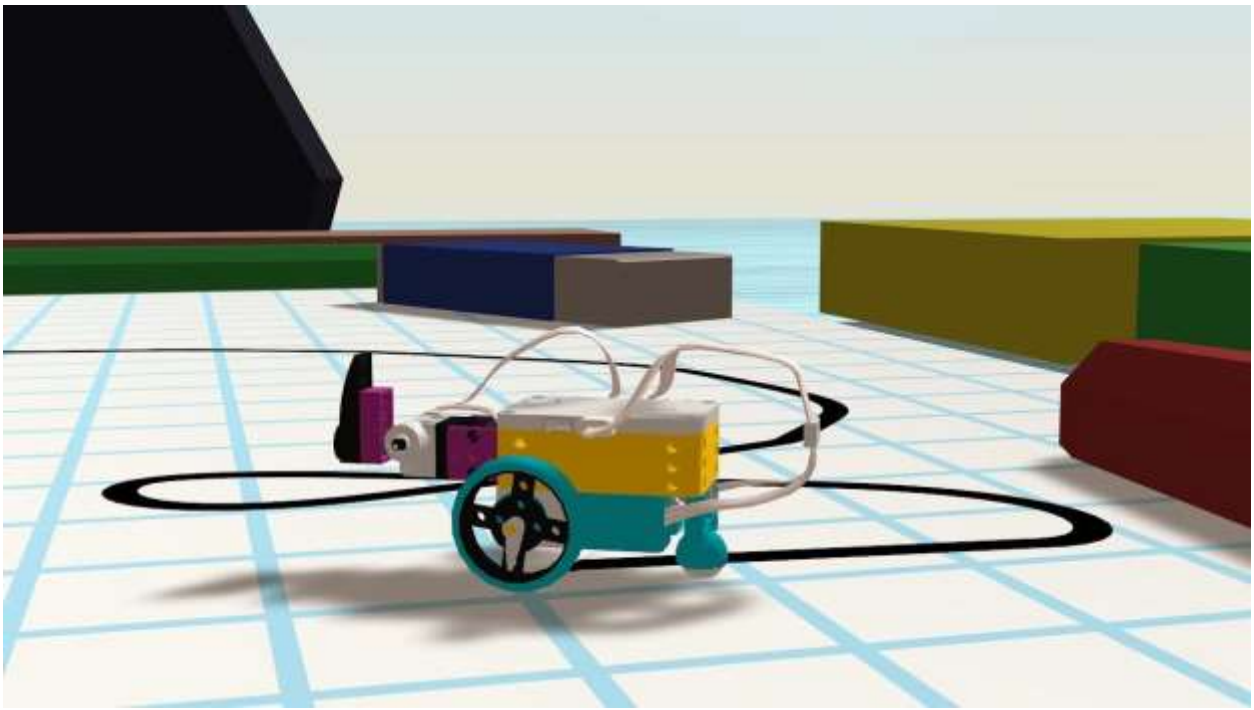


Course 5: Candy Town

What makes some ways of writing a program better than others?

Kids learn how to move their robot in new and interesting ways, then compare the advantages and disadvantages of different algorithms.

6

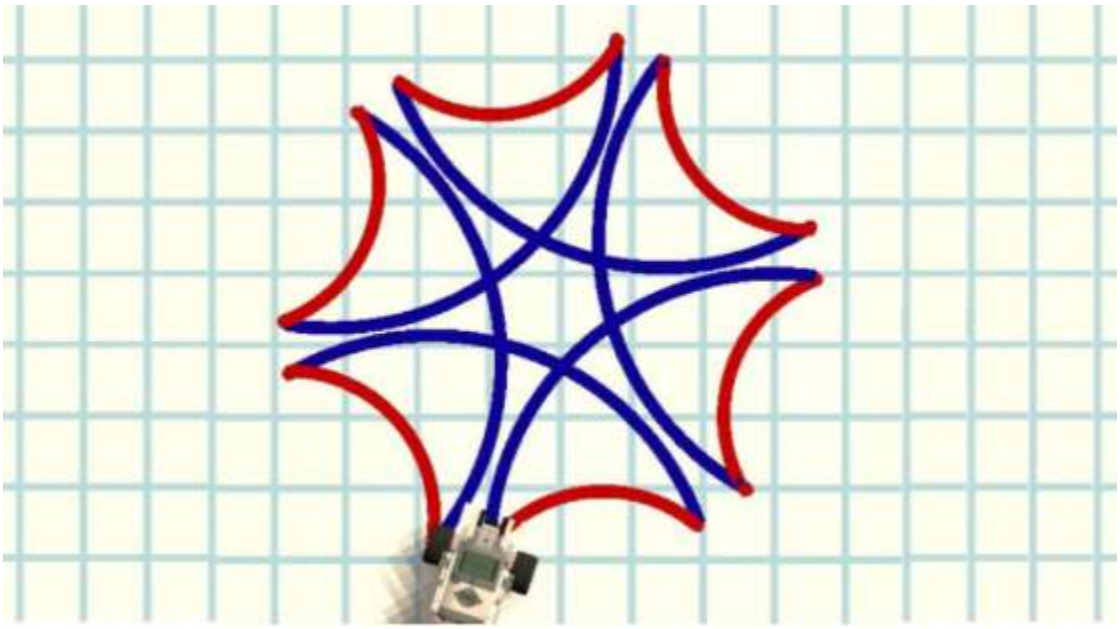


Course 6: Sketch It!

How can I solve problems using code?

Kids learn to decompose a problem into its component parts while they code their robot to draw various pictures and patterns.

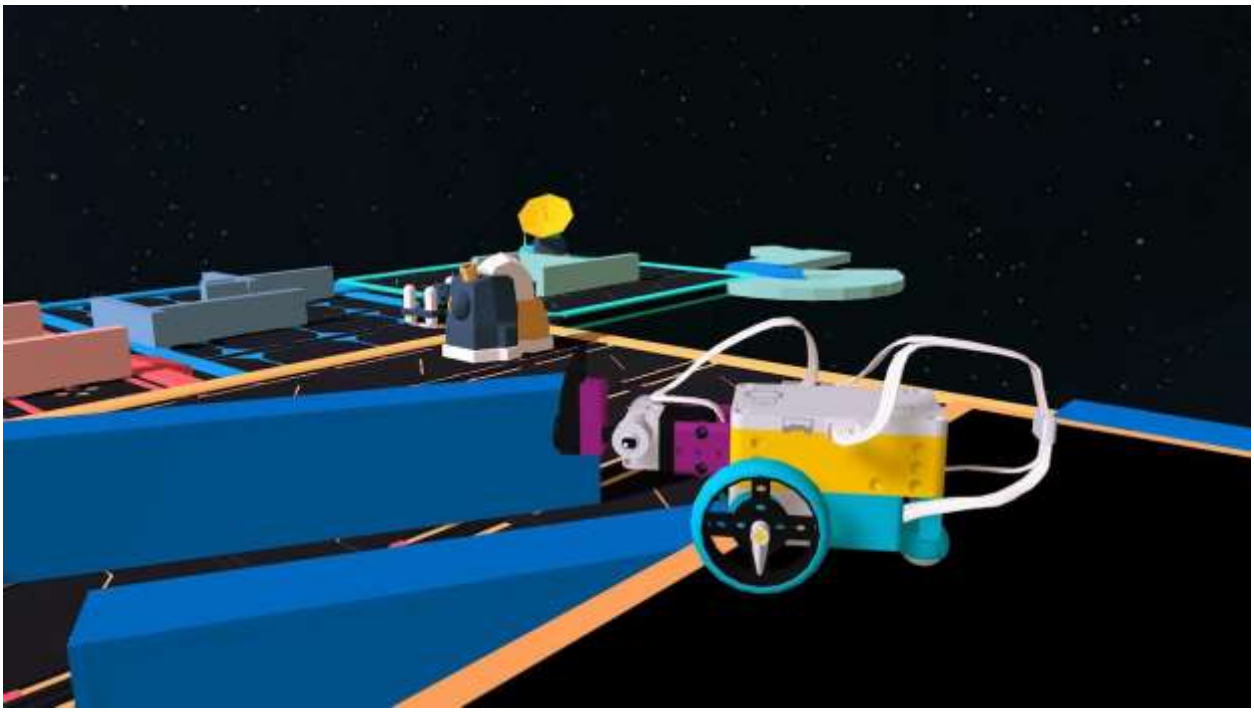
P



Project: Sketching Project

Kids use what they have learned to design and create a picture or pattern of their own. **(Optional)**

7



Course 7: The Milky Way

What practices and processes can make someone a better coder?

Kids learn to use a structured software development process and reflect on how they can learn to make better programs.

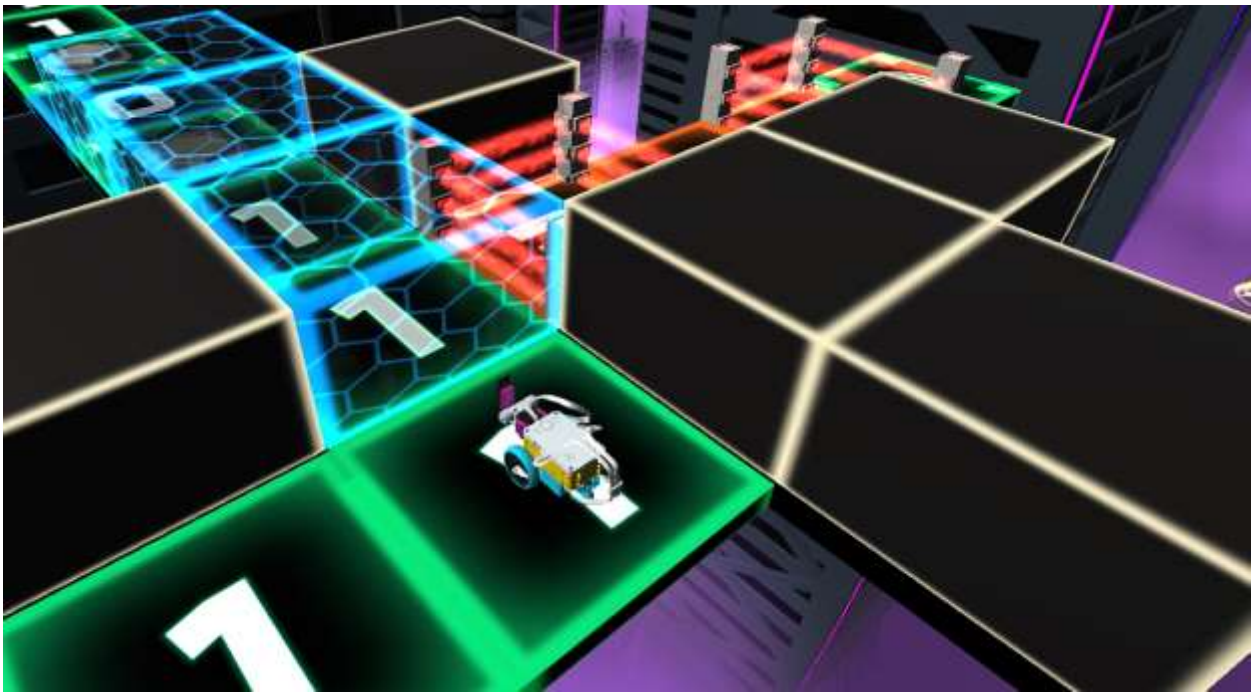
8

Course 8: Weebo's Manor

How can I change what a program does based on what it senses in its environment?

Kids use the robot's distance sensor to create programs that can adapt to unpredictable environments.

9



Course 9: Cyber World

How can I use a robot to affect its environment?

Kids use new types of output to move obstacles around the environment before sending the robot to the target.

10

Course 10: Arctic Lab

How can robots work together to solve a problem?

Kids use a coordinate system to help their robot and a helicopter drone work together to create a safe path and get the robot to the target.

>>Course 1: Adventure Peak

INTRODUCTION

Welcome to Codyssey World! Start by using the overview video to introduce the platform. Then enter Codyssey World and begin the first set of missions in Adventure Peak.

LEARNING OBJECTIVES & GUIDANCE

Use sequencing to create a program

Check that your child has completed Missions 4 and 5 successfully. You may also use Reflection Question 1 to make sure they understand the relationship between the order in which the blocks are configured and the order in which the robot completes the commands.

Describe how software commands affect hardware behavior

Use Reflection Question 2 to check your child's model of robot behavior, that robots (hardware) are acting out commands given to them by code (software).

REFLECTION QUESTIONS

Q1: Today, your robot was able to find its way through several mazes. How did it know which way to go?

A1: Answers may vary, but ensure that your child recognizes that robots are following instructions that come from a human programmer, that programmers use code to instruct the robots, and that the code must include the correct commands in the correct order.

Q2: Can you think of another type of robot? What kind of code would it need to work?

A1: Robots may be real robots that kids have encountered, or ones that they imagine for themselves. The code should have some relationship to the purpose of the robot and to the features that the robot has. Kids should be able to distinguish between the hardware and software of the robot, and describe how they are related.

Q3: What are two important things someone should remember when planning a program?

Q3: Your child may want to share their tips aloud with you. They can mention specific features of the UI, such as that the blocks are organized by color, tips specific to the programming language, such as that blocks are clicked together in the order that the robot will execute commands, or more general tips, such as that it's useful to look around the environment of the robot before beginning to program.

ADDITIONAL SAMPLE QUESTIONS

Q1: What actions did the robot perform in this session?

A1: The robot drove forward and backward, for set distances or until it reached the target. The robot turned left and right at 90-degree angles. The robot pressed buttons to complete bridges, raise platforms, or lower roadblocks.

Q2: How did we re-align the robot when we were facing the wrong way?

A2: By turning (either left or right) twice in a row.

Q3: How did we know what distance to program the robot to drive for each step of the code?

A3: By using Explore Mode to see the scene from above and checking the distances drawn on the road.

Q4: Why might a programmer want to drive the robot backward?

A4: In some cases the robot may be positioned precariously, like right on the edge of a ledge or a cliff, and if we try to turn we might fall off. Driving backward is safer. Even when there is no danger from turning, turning is another 2 lines of code (2 Turn blocks in a row, either left or right). Driving backward is just 1 line of code - it is more efficient, and when writing code we want to be as efficient as possible.

>>Course 2: Frozen Island

LEARNING OBJECTIVES & GUIDANCE

Describe the benefits of planning an algorithm before beginning to code

In Reflection Questions 1 and 2, ensure your child understands that planning can help them to understand what the code is supposed to do before coding, saving time and frustration that can come from a "guess and check" method.

Use pseudocode to plan a program

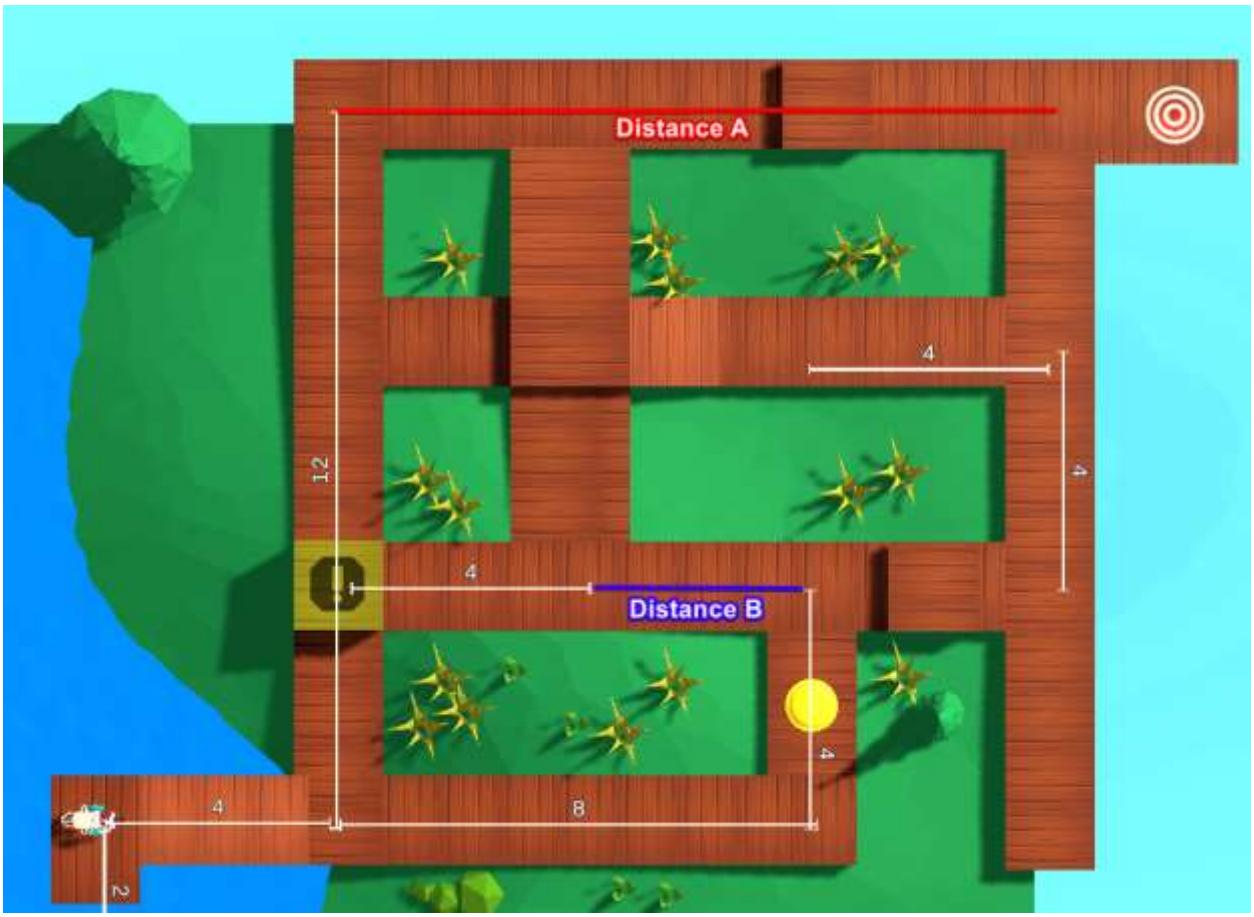
Check that your child has used pseudocode to plan at least two of their programs. Use Reflection Question 3 to ensure they understand that pseudocode will help them to plan a program without worrying about which specific blocks they will need.

Use data to create a model for a program's behavior

Check their success on Mission 6. You may also want to use their pseudocode as evidence of modelling the program's behavior.

OVERVIEW

This lesson makes explicit some of the lessons that were implicit in the final mission of Adventure Peak. To make sure your child is ready for the kind of reasoning that this lesson requires, start with a short review. Display the Explore Mode view of Mission 7 from Adventure Peak.



Ask your child for the length of **Distance A**. This is not a measurement they used directly in the previous solution, but they should be able to figure it out.

Once they have answered that the length is 12 units (meters) ask them to explain how we know that it is 12. They should be able to tell you that the 12 comes from adding the known length of 8 that is given on the bottom path and the 4 from the horizontal path second from the top.

Now ask the same questions about **Distance B**. This time the length of 4 comes from subtracting the 4 from the path second from the bottom from the bottom length of 8. Explain that in today's missions it will be necessary to use similar reasoning to calculate the various distances the robot needs to drive.

Kids should proceed through the missions in the Frozen Island Pack at their own pace. The missions become more challenging as they progress, but there aren't any new major concepts introduced in the sequence.

Give your child the following pieces of advice before they begin:

Using Explore Mode is mandatory in these missions. All of them have measurements marked on the roads that are not visible in Scene Mode.

View the tours for each mission. In most cases they explain exactly what to do.

If your child skipped the tour or wishes to view it again, have them restart the mission by clicking the button by the mission menu.

In Mission 5, distances are marked with ratio notation, as in 16:4 or 8:2. This is simply another way to represent a fraction; 16:4 means exactly the same thing as 16/4 or "sixteen over four". Point this out before they begin and remind them as necessary once they reach Mission 5.

REFLECTION QUESTIONS

Q1: What are two important things to remember when planning a program?

A1: Your child may want to share their tips with you. This discussion can be used as a chance to review the planning process (gather data, explain the plan, code the plan, test the plan). They may also mention the benefits of planning.

Q2: How can taking time to plan make coding faster?

A2: Planning can reduce the time needed to guess and check, or reduce the time spent finding problems in code. Planning also helps kids have a purpose when they begin to code, preventing them from being overwhelmed when faced with an empty code area.

Q3: How did pseudocode help you make your plan specific enough to code?

A3: Your child should mention specific instances when they used pseudocode to make a plan more specific. Check that their pseudocode includes specific distances and directions that correspond to coding options in the platform.

ADDITIONAL SAMPLE QUESTIONS

Q1: What data did we gather in Explore Mode, and what for?

A1: The lengths of various parts of the road, their intersections and mid-points. Using Explore Mode we could plan out the robot's route with minimum attempts.

Q2: Why did we pay special attention to markers placed in mid-points in some of the lengths of road?

A2: These markers were the half-way or third-way point in a stretch of road; finding it allows us to calculate the necessary drive length by dividing the total length by two or by three.

Q3: In the final mission of the pack, how did we know the correct distances to drive when some parts of the route were not labelled?

A3: By using Explore Mode to look at the scene from above and calculating with the data that was provided and inferring the length by comparing it to parallel parts of the route.

Course 3: The Lost City >>

LEARNING OBJECTIVES & GUIDANCE

Explain the benefits of using loops in programs

In the reflection questions, make sure your child has noted that loops make code both shorter and easier to read.

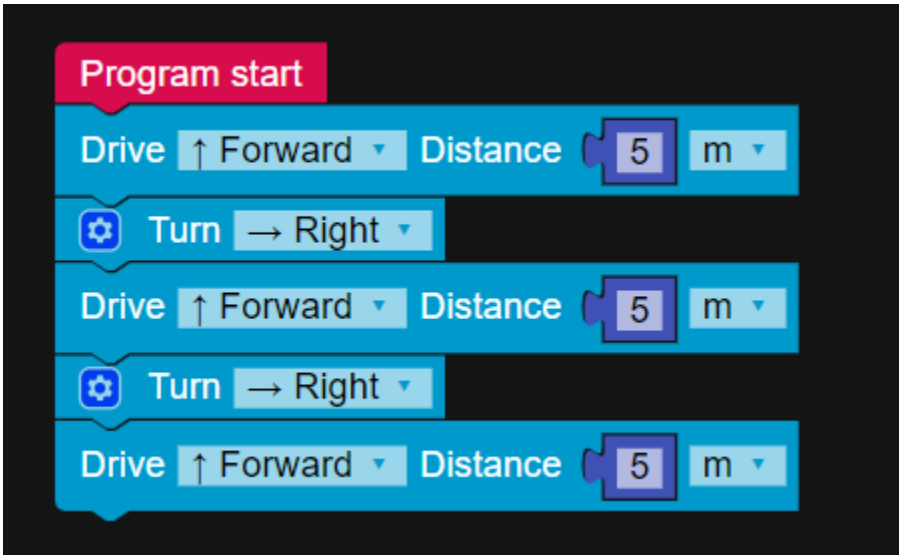
Use loops to execute a program

Check that your child has both completed Mission 6 successfully and has used a repeat block in their solution.

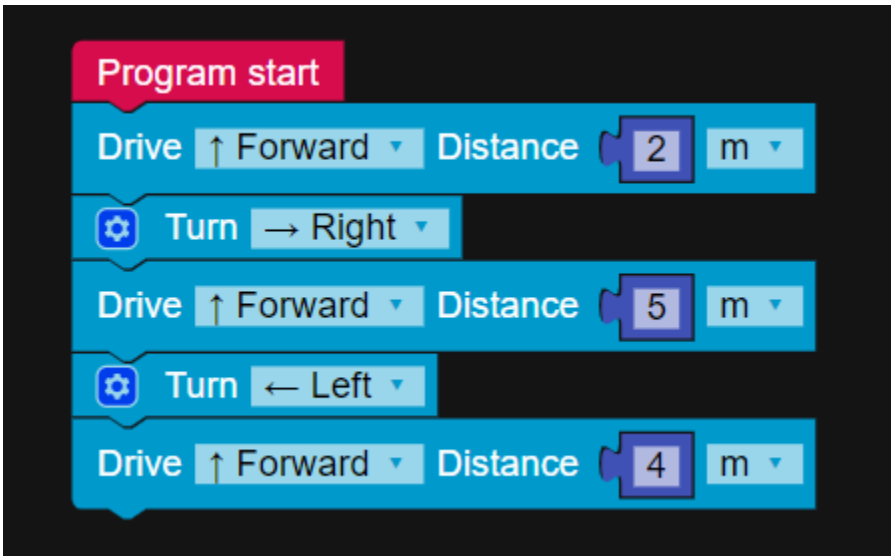
OVERVIEW

Show the following programs and ask your child to make a sketch of a course that each program would solve. The sketches should be from a top down perspective, as in Explore Mode.

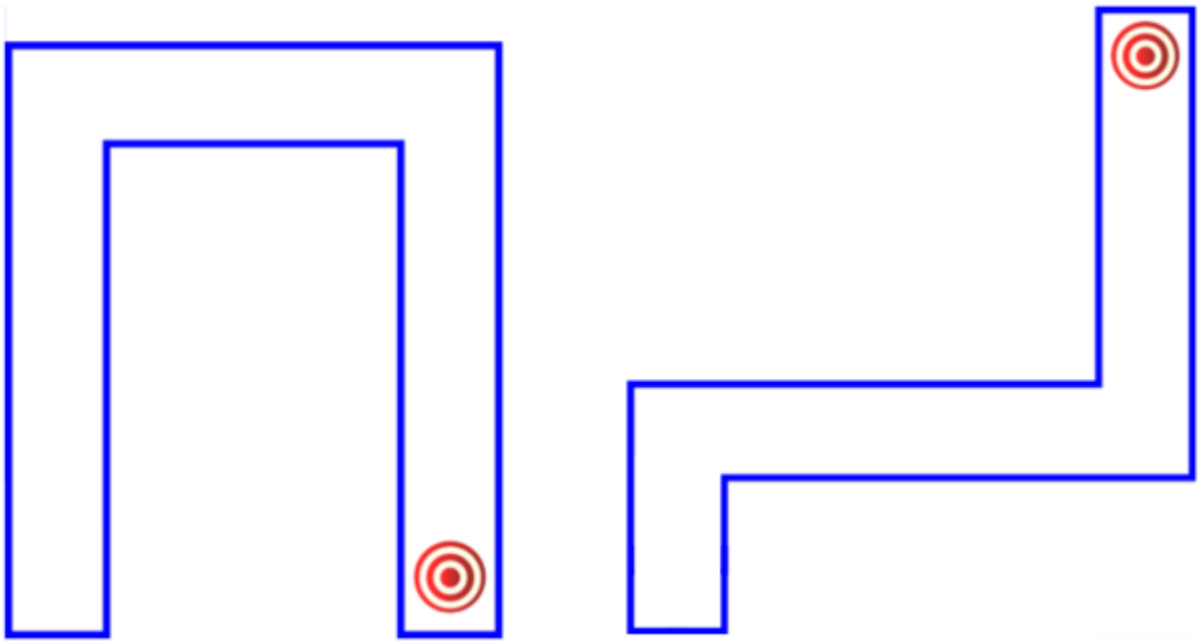
Program A



Program B



Give your child two or three minutes to complete their sketches before sharing. Sketches should look roughly like the ones below.



Ask your child to explain the reasoning they used to interpret the program and make their sketch.

MISSION BREAKDOWNS

Click through to get a mission-by-mission breakdown, with in-depth explanations, tips, helpful advice and more!

REFLECTION QUESTIONS

Q1: What are two ways that loops are useful in programming?

A1: While your child may think of various benefits of loops, ensure that they understand that loops make code shorter and more readable. Be sure to address any misconceptions, such as the belief that the loops make the code faster or easier for the computer or robot to execute.

Q2: What are three tips for someone who wants to use loops in their program?

A2: Your child may want to share their tips with you. Some potential tips are to try to find patterns of repeated code, to write the loops into the pseudocode in a particular way, or to keep track of which way the robot is turned after each iteration of the loop.

ADDITIONAL SAMPLE QUESTIONS

Q1: Why do we have to use the Wait Block in our code?

A1: Because it takes the ramps and gap-fillers several seconds to move into place after the robot presses the button.

Q2: Should the robot press a button, wait, and then turn; or press a button, turn, and then wait?

A2: It doesn't matter – as long as the robot completes these 3 steps before it starts driving forward, the gaps will be filled.

Q3. What does a Repeat Loop do to the code inside it?

A3. It repeats the code however many times it is set to run.

~~QUIZ ANSWER KEY~~

~~In addition to the more open-ended and exploratory course questions, quizzes are another tool we offer for you to assess your child's learning. They're not required in order to complete the curriculum, so feel free to skip this part if quizzes aren't your thing.~~

Course 4: Crystal Crater >>

LEARNING OBJECTIVES & GUIDANCE

Use parameters to change the behavior of a method

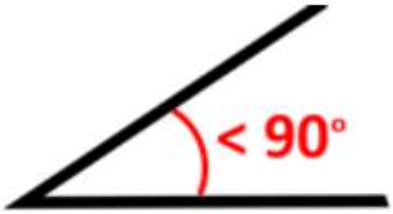
Check that your child has completed Mission 8 successfully.

Test and debug a program

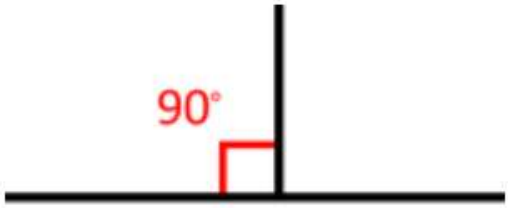
Check the answers to the second and third reflection questions to see whether your child is describing a reasonable debugging process.

OVERVIEW

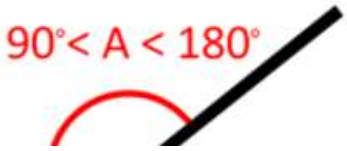
Start the lesson with a review of basic angle vocabulary. You child should at least know these terms:



Acute Angle: An angle whose measure is between 0 and 90 degrees.



Right angle: An angle whose measure is exactly 90 degrees.



Obtuse angle: An angle whose measure is between 90 and 180 degrees.

Then take a quick look at the concept of interior and exterior angles of a polygon. Show the following image. Point out that the triangle is an equilateral triangle and that as expected each angle is 60 degrees. Explain that these are the interior angles of the triangle and they will always add up to a total of 180 degrees.

We can also talk about the exterior angles of a polygon. These are formed by extending one of the sides of the polygon as seen in the picture.

Ask your child what they notice about the measures of the interior and exterior angle seen here. They should realize that since together they make a straight angle, they add up to 180 degrees. Your child may already know that the term for two angles whose measures add up to 180 is supplementary angles.

Explain that the reason we care about exterior angles is that if we are programming the robot to drive around a shape, we must use the exterior angles of the shape, rather than the interior. If you have the time and space, you can illustrate this by having your child walk around a marked-out equilateral triangle while holding their arm out in front of them. It will be easy to see that at each vertex their arm turns through an obtuse angle (120 degrees) not an acute one (60 degrees).

Most of the missions in this lesson are fairly simple from a coding standpoint. The challenge here is more centered on the underlying angle concepts, some of which may be new to your child.

By and large your child can do this entire lesson at their own pace. Assist if they get stuck on a particular concept.

MISSION BREAKDOWNS

Click through to get a mission-by-mission breakdown, with in-depth explanations, tips, helpful advice and more!

REFLECTION QUESTIONS

Q1: How do parameters make it easier to write programs?

A1: Your child should recognize that parameters allow the same smart block to do slightly different things, such as turn in different directions or drive different distances. This gives the programmer more control over exactly how a command is carried out by having the general instruction reflected in the block and the specific details reflected in the parameter.

Q2: What is one bug that you found in a program, and how did you find the bug and how did you fix it?

A2: Check that your child is describing a reasonable debugging process that includes trying to understand what in the code is causing the bug before changing the code to fix it.

Q3: What are two tips for someone who is trying to debug their program?

A3: Potential tips may include running the code and checking bit by bit until something unexpected happens, explaining the code step by step to a friend, or changing a small bit of code to see its impact.

Q4: Do you think it's easier to debug your own code or someone else's code? Why?

A4: Kids' perspectives may vary, and they may want to discuss their various ideas with you. While there is no 'right' answer to this problem, your child may recognize that while someone's own code may be easier for them to understand, another person may be able to see something that the original programmer overlooked, either because they have different knowledge and

assumptions, or because the act of trying to understand a piece of code can be an important part of debugging it.

ADDITIONAL SAMPLE QUESTIONS

Q1.: What is the difference between an acute angle and an obtuse angle?

A1: An acute angle is less than 90 degrees. An obtuse angle is more than 90 degrees but less than 180 degrees.

Q2.: What are corresponding angles? Are they always equal?

A2: When two lines are crossed by another line (the transversal), the angles in matching corners are called corresponding angles. Corresponding angles are only equal if the two lines crossed by the transversal are parallel lines.

Q3: Why is it important to know if three given lines are two parallels and a transversal?

A3: Because two parallel lines and one transversal give us several pairs of angles. Once we have the degrees of one of these angles, it is possible to figure out the rest of the angles with a few easy calculations.

~~QUIZ ANSWER KEY~~

~~In addition to the more open-ended and exploratory course questions, quizzes are another tool we offer for you to assess your child's learning. They're not required in order to complete the curriculum, so feel free to skip this part if quizzes aren't your thing.~~

Course 5: Candy Town >>

LEARNING OBJECTIVES & GUIDANCE

Compare the advantages and disadvantages of similar algorithms.

In the reflection questions, check that your child has named one advantage each for a screw and smooth turn, and has named three criteria for determining which algorithm is better. Advantages may include ease of coding, efficiency of the algorithm, or precision of the algorithm.

OVERVIEW

Begin the lesson with a small demonstration. You will need a small clear area where you can move without obstacles.

Have your child stand in the middle of the clear area and act out how the robot makes a 90-degree right turn, just as if it is executing the following block:

Note: your child should turn in place, without advancing in any direction. (This is called a 'screw turn').

Now demonstrate yourself how a car makes a right turn. Repeat the demonstration a couple of times until you both agree that you've seen a good model of what a car turning looks like.

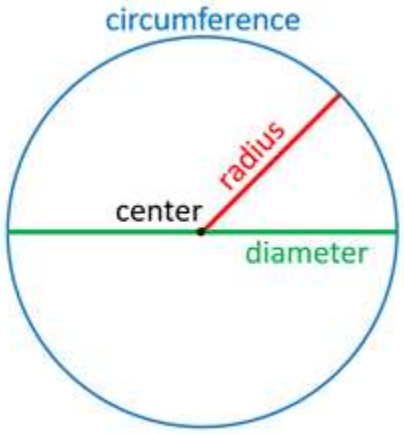
How is the robot's turn different from the car's turn? Possible answers might be that the robot stays in place while the car moves as it turns, or that the car's movement is more of a curve (this is called a 'smooth turn'). After the discussion, explain that it is possible to make the robot make turns like a car, and that that is what today's lesson is about.

Explain the following features of a circle:

Circumference – the distance once around a circle (its outline).

Diameter – a line that goes straight across a circle, through its center. It is twice the circle's radius.

Radius – the distance from the center of a circle to any point on its circumference. It is half of the circle's diameter.



Go through Mission 1 with your child to make sure they know how to configure and use the Radius parameter on the Turn block. Mission 8 is also a good place to prepare together by looking at a new geometric relationship that will be used on the last three missions.

MISSION BREAKDOWNS

Click through to get a mission-by-mission breakdown, with in-depth explanations, tips, helpful advice and more!

REFLECTION QUESTIONS

Q1: What is one benefit of a screw turn?

A1: Ensure your child understands that the ‘screw turn’ turns the robot in place. This type of turn is appropriate for corners or times when there is no space for the robot to drive in a curve.

Q2: What is one advantage of a smooth turn?

A2: Ensure your child understands that the smooth turn moves the robot in a curve. This is more appropriate for a curved path or in an open space avoiding obstacles, because it takes less time to curve around an obstacle than to make several screw turns to drive around it.

Q3: **What are three ways that an algorithm could be better than another?**

A3: Kids should understand that algorithms can be judged by various criteria, such as how easy they are to understand, how much time they take the robot to execute, or how well they meet the goals of the program itself.

ADDITIONAL SAMPLE QUESTIONS

Q1: What is the difference between a radius and a diameter?

A1: A diameter is a line that goes straight across a circle, through the center. It is twice the circle's radius. A radius is the distance from the center of a circle to any point on its circumference. It is half of the circle's diameter.

Q2: What is the difference between a screw turn and a smooth turn?

A2: A screw turn is a turn on the spot in which the robot does not drive anywhere (does not advance). It is executed by turning both wheels at the same speed in opposite directions. A smooth turn is an arc, meaning that the robot advances and moves forward or backward in the scene and does not stay in the same place as where it started.

Q3: What is the advantage of smooth turns? Why not simply drive forward, turn [+/-10] degrees, drive forward, repeat?

A3: Driving in a smooth turn (an arc) is more efficient in many ways. It is only one block of code instead of several. It is shorter driving time -- after every single block there is a pause, and the more blocks there are, the more pauses. It is more precise – easier to set the exact angle and radius instead of adding up several angle-turns. With several short drives there is more room for error.

Q4: Advanced: How do you think the robot manages to drive in smooth turns according to the set radius?

A4: The robot has 2 motors. When executing a smooth turn, both wheels turn in the same direction, at different speeds. The speeds are determined so that the ratio

between them is the same ratio as the distance from the center of the circle to each wheel. For example: if the robot is turning right, then the right wheel is the 'inner' one, the wheel turning slower, and the left wheel is on the 'outside' of the arc.

Course 6: Sketch It! >>

LEARNING OBJECTIVES & GUIDANCE

Decompose a problem into its component parts.

Check your child's work to see how they have broken down the letters or pictures into parts, and use the second and third reflection questions to check that they can list examples of decomposition.

Model programs that use hardware to solve problems.

Check that most missions have been completed successfully.

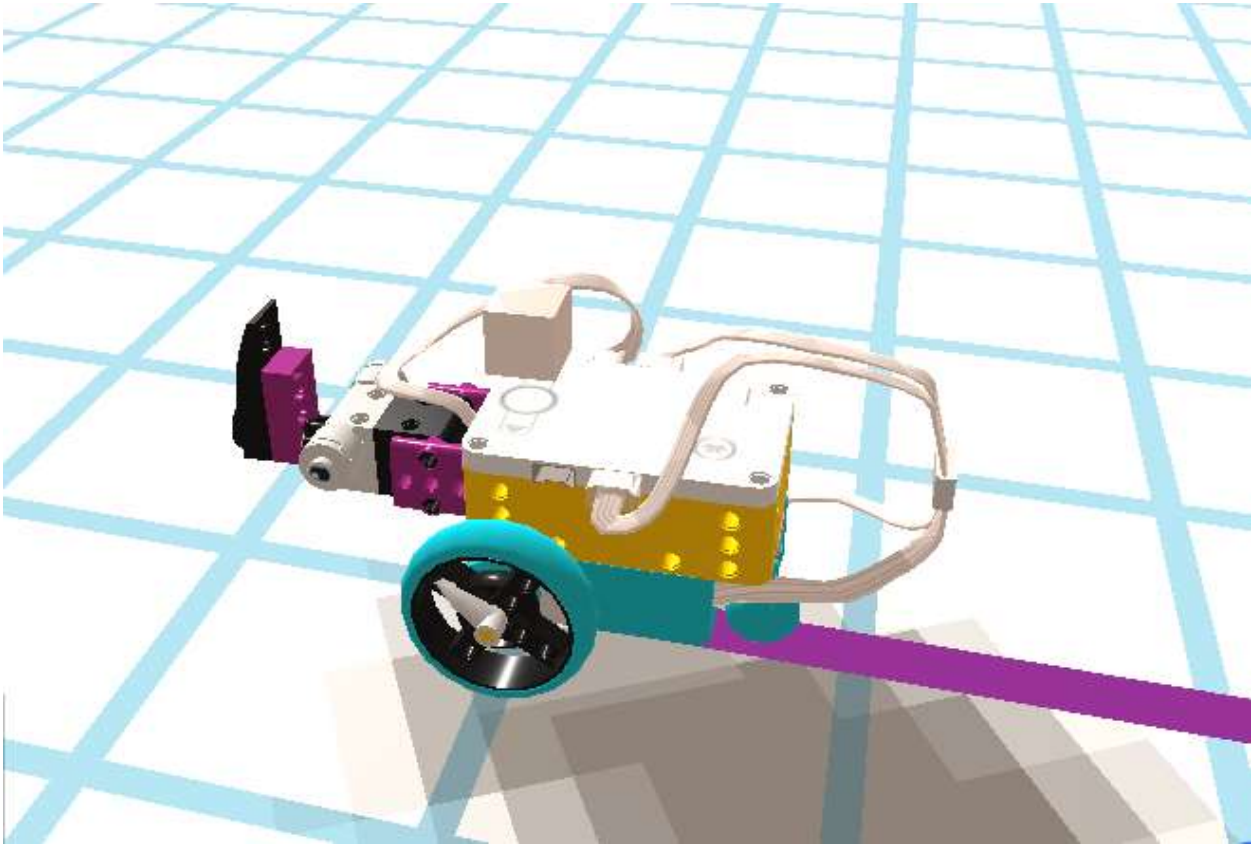
OVERVIEW

The missions in this course quite different from the others. There is no target to reach and kids will not receive a "you have completed the mission" feedback at any point. But that doesn't mean there are no goals to achieve.

In these missions the robot is equipped with a set of trail drawers (painters) that allow it to leave a colored trail as it moves through the environment.

Each mission will give kids a particular drawing task.

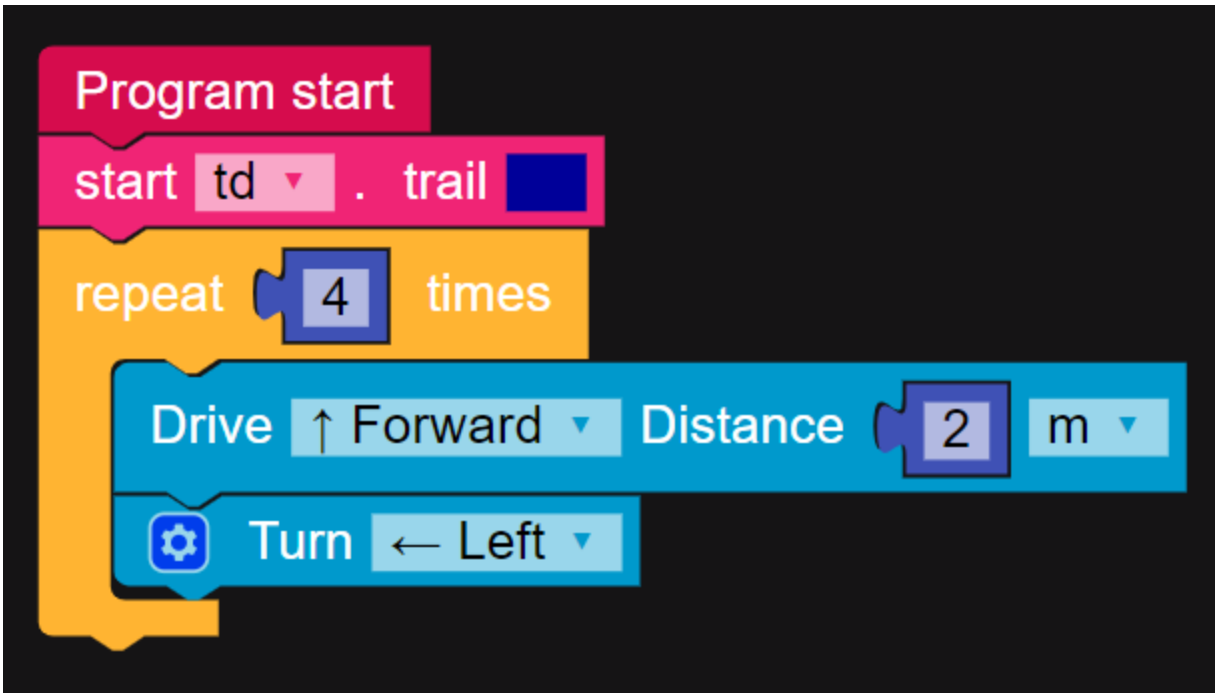
To get kids up to speed on these new tools before they begin work on the missions proper, have them enter Mission 1 of the Sketch It! Pack (Freestyle). While they will ultimately be working towards the goal of this mission (drawing their name or initials), in this first part of the lesson they should view the tour to get acquainted with the drawing tools in the Gadgets menu by doing a little free form coding.

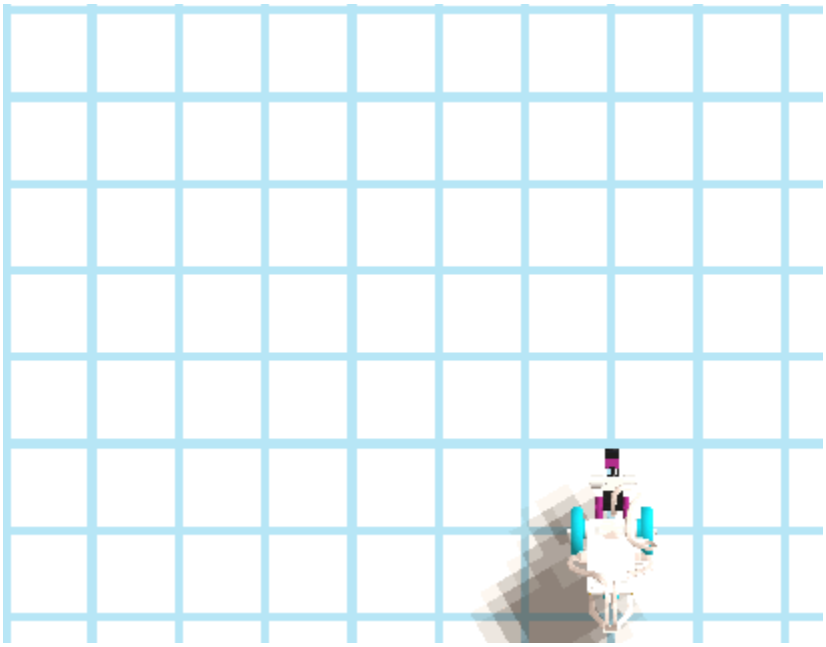


Once your child feels like they have a handle on how the tools work, have them demonstrate their mastery by completing the following three tasks. Once these tasks are completed they can move on to the missions. Solutions programs are included.

TASK 1

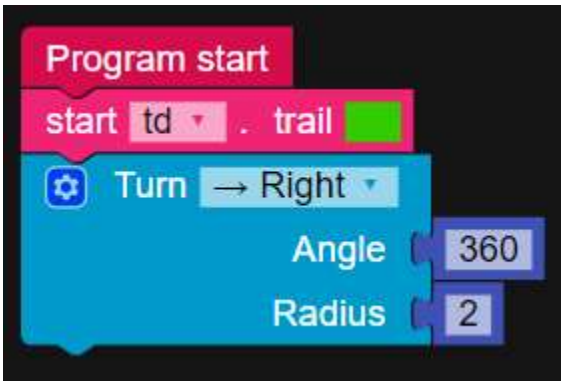
Draw a blue square

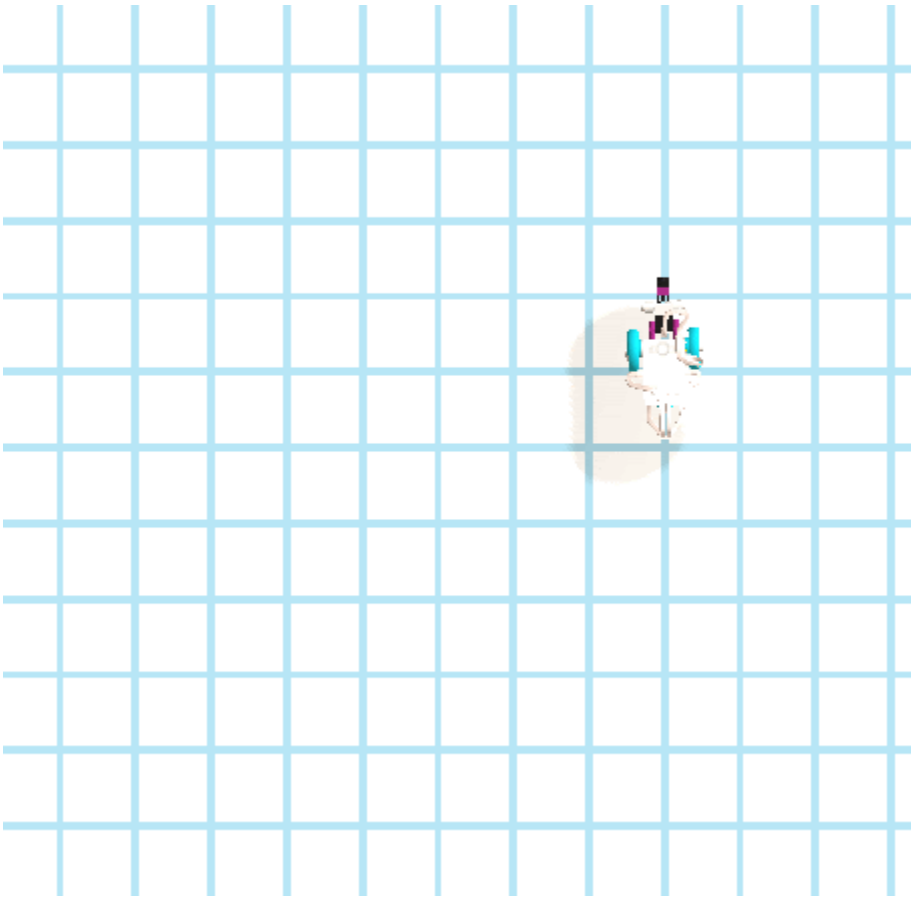




TASK 2

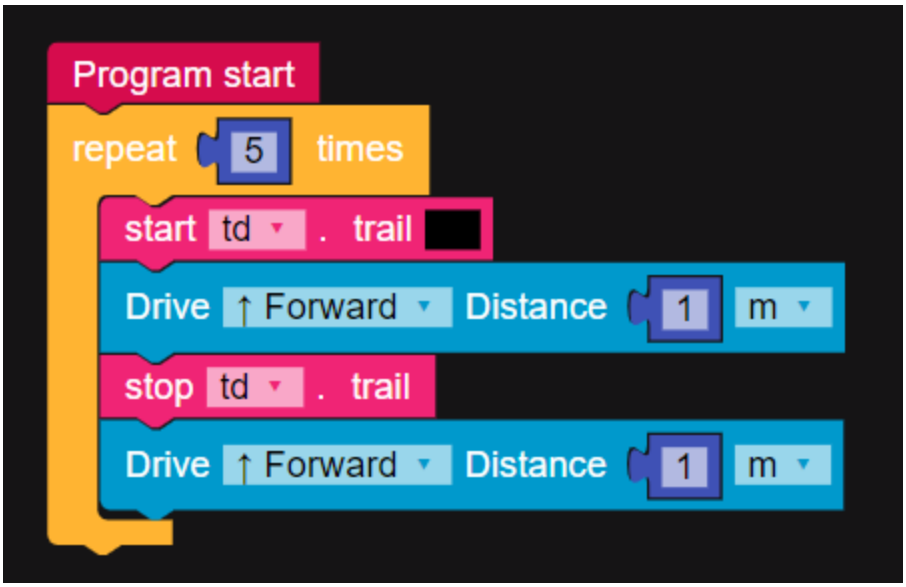
Draw a green circle

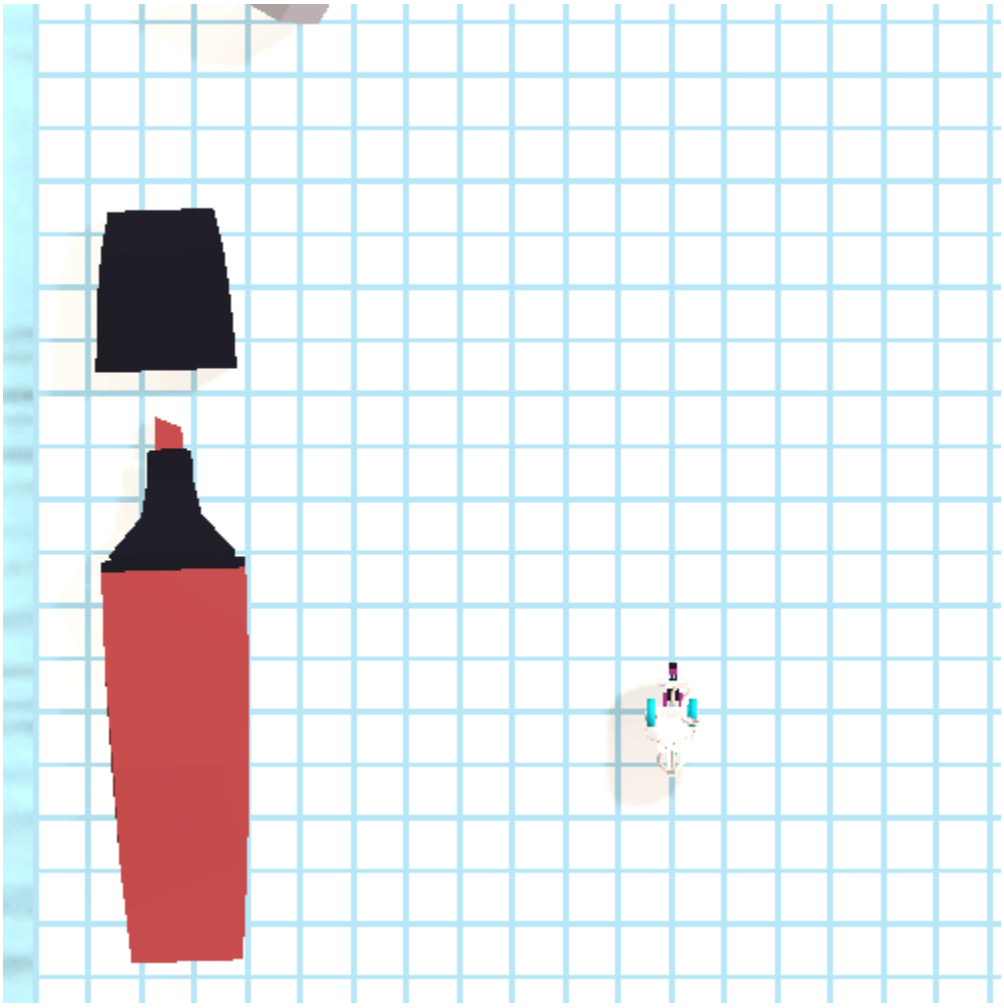




TASK 3

Draw a dashed line with at least 5 dashes





Because the missions in this pack are open ended and involve a level of creative expression, a computer cannot judge the level of success. So for each of these tasks, you may choose to evaluate

your child based on their completion of the respective tasks or choose to tailor the missions with your own requirements.

REFLECTION QUESTIONS

Q1: What are three tips for someone who is trying to draw a complicated picture with code?

A1: Kids may suggest tips such as drawing the picture by hand ahead of time, breaking down the picture into smaller pieces, identifying patterns, or coding and testing a bit at a time. While there are no specific 'right' and 'wrong' answers, ensure that your child understands the role of decomposition in addressing complex problems.

Q2: What are two ways that breaking down a big problem into smaller problems can help?

A2: Kids should understand that breaking down problems allows someone to address a large, complex problem a bit at a time, treating it as a series of smaller problems, then connecting the solutions. Kids may also identify other advantages, such as making it easier to identify patterns, or even dividing up work when part of a group.

Q3: Give an example of when you broke down a big problem into smaller parts today. What was the big problem, and what was one smaller part?

A3: Check that your child's answers make sense and that the smaller part of the problem is in fact part of the larger problem and can be solved independently of other parts.

Course 7: The Milky Way >>

LEARNING OBJECTIVES & GUIDANCE

Use a structured software development process

In the first reflection question, ensure that your child has described a process that includes understanding a problem (e.g. through data collection in the environment), planning for a solution (e.g. planning an algorithm with pseudocode), coding the solution, testing the code, and repeating the process until the problem is solved.

OVERVIEW

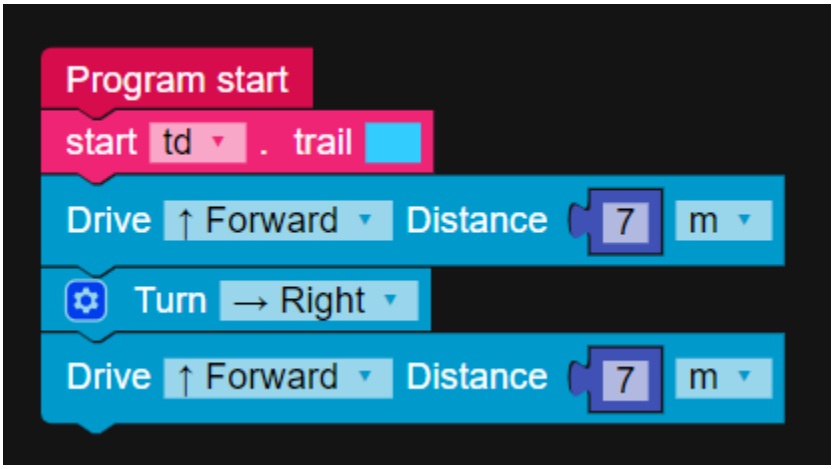
Log into Codyssey World and go to Mission 1 of the Sketch It! pack. If your child's code from the previous lesson is still there, they should simply detach it from the Program Start block.

The goal of this exercise is to have kids explore two short programs that move the robot to the same position and compare how long each program takes to execute. They will need to have access to some kind of stopwatch; there are many available online and in the app stores for the different kinds of smartphones.

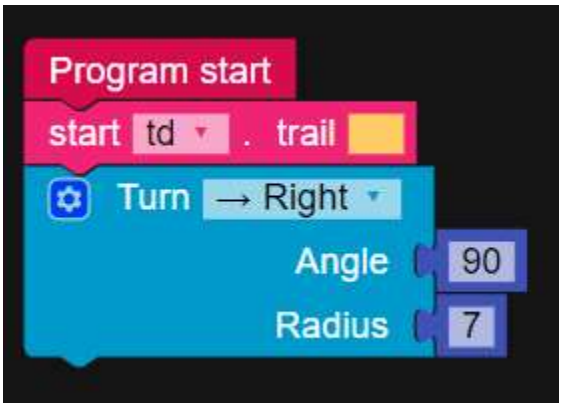
Have your child use two different trail colors for each program so that they can easily follow the robot's trajectory.

These are the two programs. Let your child first predict which one they think will be fastest, then test and time each one to see if they are right. If you're following along with your child or teaching more than one child, this would be an ideal activity to work on as a pair.

Project A

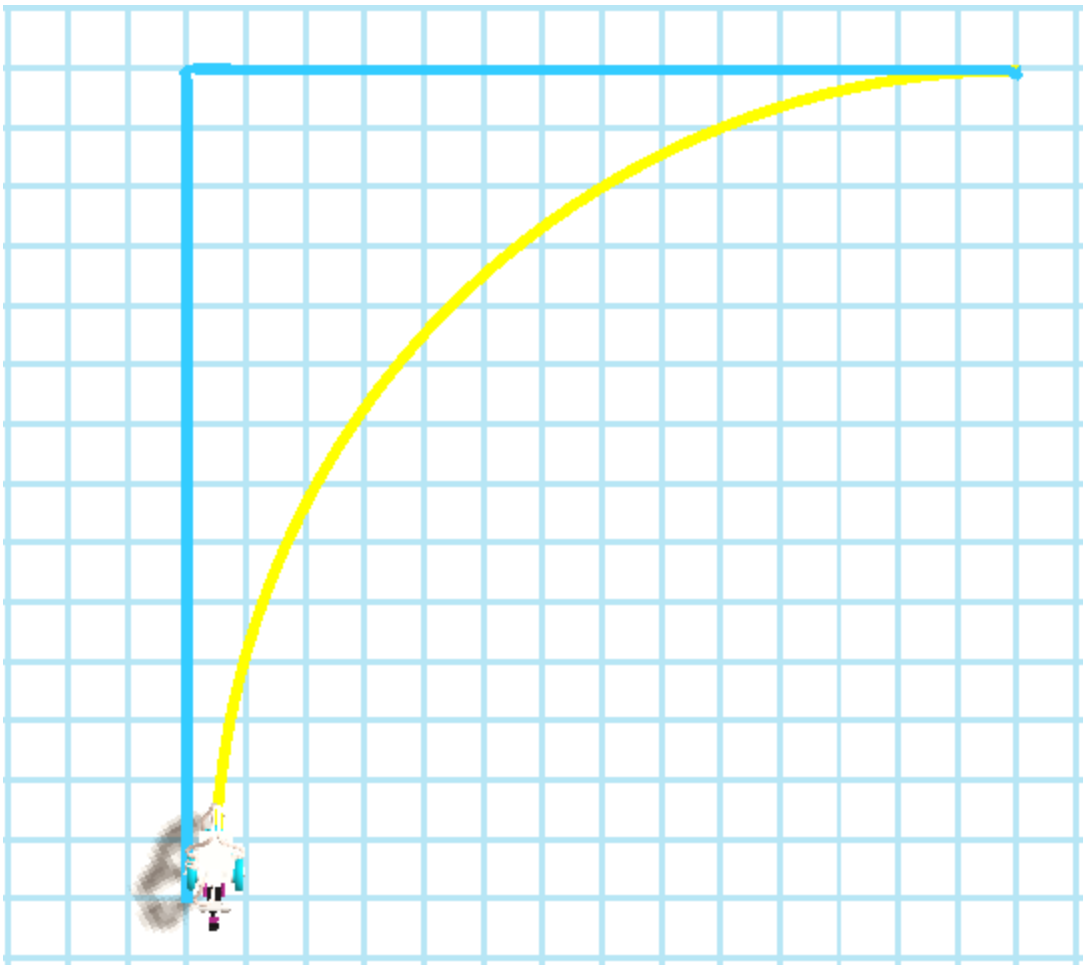


Project B



After your child has conducted their tests, discuss the results. Which program was faster? Why? You can show them the image below to reinforce that the two programs do in fact accomplish the same thing in terms of where they move the robot.

Explain that the reason for this exercise is that in some of the missions in this lesson there will be a time limit that will force them to think not just about whether or not their code works but how fast it gets the job done.



REFLECTION QUESTIONS

Q1: What steps are important when you are solving a problem with code?

A1: Kids should make a connection to the four steps listed earlier in the course (gathering data, explaining the program, coding the program, testing the program). It is not necessary that they use the exact vocabulary, but they should understand that solving a problem involves understanding, planning for, and testing a solution.

Q2: What are three skills people need to be an effective coder?

A2: Answers may vary, but can include skills such as paying attention to details, planning, creativity, persistence, and willingness to try things out. Ensure that your child takes the time to consider these broad cross-cutting practices and are not overly focused on technical skills particular to the CoderZ platform.

Q3: What was the most fun part of these challenges?

A3: Allow your child to share what they enjoyed most about the challenges with the class. Encourage them to think about what made things fun for them, especially when their enjoyment helped them to overcome challenges or, conversely, overcoming challenges made the activity more fun.

ADDITIONAL SAMPLE QUESTIONS

Q1: When should we drive in a straight line, and when should we drive in curves and arcs using smooth turns?

A1: It depends on the route and on whether there are obstacles in the robot's path. If there are no obstacles, a straight line is always quickest. If we have to maneuver around obstacles or walls, then driving in a curve will save time since it is only one block of code.

Q2: Is driving forward always the best strategy?

A2: No. If the robot is facing 'the wrong way' and we code it to turn around, we are wasting precious time. It is better to drive backward than to spend time and write a longer program to make the robot turn around.

Q3: What is another way to shorten our program, other than choosing the correct driving path?

A3: Using repeat loops.

Course 8: Weebo's Manor >>

LEARNING OBJECTIVES & GUIDANCE

By the end of this lesson kids will:

- Use the distance sensor to gather information from the environment
    - Check for mission completion in "Platform Hopper" and "Popcorn Puzzle".
- Change what a program does based on what it senses in its environment
    - Check for mission completion in "Platform Hopper" and "Popcorn Puzzle". You may also use the answers to the reflection questions, checking that your child makes connections between different sensor values and different commands the robot should follow.

OVERVIEW

Review the software development process from the last lesson. Discuss need to use data in the environment to change a program as it's running.

REFLECTION QUESTIONS

Q1: What makes it possible to plan an algorithm even though an environment might change as the program is running?

A1: Answers should mention gathering information from the sensors as the program is running as well as rules that connect the sensor values to desired robot behavior.

Q2: In these missions, you used the "Wait Until" and "Drive Until" commands to tell the robot when it should move in different ways. Think of a different way you might want to use the information your robot senses in your environment. What type of instructions might you need, and how would you use them?

A2: Answers may include using the sensor values to tell the robot exactly how far to go, or in what direction to turn.

Q3: Can you think of a real life robot that might use a distance sensor? How would it use the information from the sensor to change its behavior?

A3: Self-driving cars and almost all robots that move about in their environment include distance sensors. Encourage your child to think of specific examples they may have seen or heard of and why those sensors are important to that particular robot.

ADDITIONAL SAMPLE QUESTIONS

Q1: How do you measure the distance of an object behind the robot?

A1: You must turn the robot around because the distance sensor is at the front of the robot.

Q2: What is the furthest away that the distance sensor can measure?

A2: 10 meters.

Q3: When is it better to use the Wait until Distance block and when is it better to use the Drive until Distance block?

A3: The Wait Until Distance block is used when the robot needs to **stay still** until the distance from it to an object is correct. The Drive Until block is used when the robot needs to move to the correct place by **driving** (forward or backward **until** the distance from it to an object is correct.

Course 9: Cyber World >>

LEARNING OBJECTIVES & GUIDANCE

By the end of this lesson kids will:

- Use the Push Block command to move cyber blocks within the environment
    - Check for mission success in any mission in the latter half of the pack. Kids do not need to complete all of the missions to demonstrate proficiency.
- Use multiple problem solving strategies to develop algorithms to complete a task
    - Check your child's responses to reflection questions 1 and 2. Answers may vary, but ensure that they are describing reasonable strategies to solve the problem, such as gathering data from the environment, decomposing the problem into smaller parts, or explaining the algorithm in pseudocode or natural language.

OVERVIEW

In Cyber World, kids use a ‘Push block’ feature of the robot to move obstacles around the environment before sending the robot to the target. The ability to affect the robot’s environment, as well as navigate through it, adds a new dimension to the planning process and extends the input-output model of how robots interact with their environments.

This lesson also serves as an overall review and challenging sequence for kids to use the problem solving, software development, and debugging skills that they have learned throughout the course.

Warm-Up Activity

Give your child some time to write down everything they have learned about the following topics: debugging, planning a program, and solving problems with code.

Remind them that so far in all of their environments, they have been navigating around the obstacles. In the upcoming missions, the robot will be able to push the obstacles out of the way.

What might have to change about how they use their skill now that they can move obstacles around? Will it make the missions simpler or more complicated?

Remind them of the practice of Pseudocode, which can be very useful in planning the robot's route, and will save them time in the long run.

REFLECTION QUESTIONS

Q1: How did you figure out what the numbers on the different floor tiles meant, and how did you use that knowledge to help you succeed in your missions?

A1: Kids should indicate that they needed to run different pieces of code to figure out the meaning of the numbers on the floor tiles. Gathering data about the environment and how it works is one step in planning and developing a program.

Q2: What was the most challenging mission in this pack and two different methods you used to succeed in that mission?

A2: Answers may vary, but make sure your child describes distinct strategies such as guessing and checking, running small bits of code at a time, explaining their algorithm in plain English or pseudocode, or looking at previous, similar missions for guidance.

Q3: Did having the ability to push the Cyber Blocks around make the missions simpler or more complicated? Why?

A3:Look at your child's reasoning for evidence of their understanding that while having more options can be helpful, it also can complicate the planning and development process.

ADDITIONAL SAMPLE QUESTIONS

Q1: What is the parameter in the Push Block command used for?

A1: The parameter is used to tell the robot how many power pulses to send to the Cyber Block. More power pulses can push the block across a higher number of squares.

Q2: What is the distance between the center of one floor tile to the center of the floor tile next to it?

A2: The distance between the center of one floor tile and the next is 2 meters. That is also the length of each of the sides of the tile.

Q3: What is the meaning of the numbers on the floor tiles?

A3: The numbers on the floor tiles indicate the level of friction each tile has, which is also the number of power pulses it takes to move a Cyber Block off of the square using a Push Block command.

Course 10: The Milky Way >>

LEARNING OBJECTIVES & GUIDANCE

By the end of this lesson kids will:

- Use a coordinate system to specify a location
    - Check for mission success in any mission in the latter half of the pack. Kids do not need to complete all of the missions to demonstrate proficiency.
- Coordinate the behavior of multiple robots to solve a problem
    - Check for mission success in the latter half of the course, and review the answer to reflection question 1 to ensure that your child can identify information needed to coordinate a solution to a problem.

OVERVIEW

- Give your child the following prompt:

    - We've spent a lot of missions explaining to the robot how to get to a target. In our everyday lives, how do we find our way to where we want to go?
- Allow your child to share their ideas. They may speak of ideas such as giving directions, cross streets, addresses, or GPS.
- Reinforce the idea that when we communicate with others about location, we need to have a shared understanding of where things are. That can be addresses, street grids, or a GPS system.
- Review the idea of a coordinate system, and explain that this new pack will introduce a new helicopter robot that uses a coordinate system rather than specific directions to tell it where to go.
- Ask your child to brainstorm advantages and disadvantages for each system and create a collective list.

REFLECTION QUESTIONS

Q1: Think of a real world situation in which two robots might need to work together to complete a task or solve a problem. What information will they need to share with each other?

A1: Answers may vary, but check that your child describes information that is needed to complete the task.

Q2: Why is it necessary to tell the robot the exact path to take, but the helicopter can just be given coordinates?

A2: Answers may vary, but your child should recognize that because it moves on the ground, the robot must be programmed to avoid obstacles and stay on the platform, whereas the helicopter does not need to worry about these things when flying.

Q3: **With the current code, the ice block must be on a particular platform for the helicopter to pick it up. Design a new block that will allow the helicopter to pick up a block from anywhere on the grid and drop it off anywhere on the grid. What parameters will it need?**

A3: Blocks parameters should include x and y coordinates for the pickup location as well as x and y coordinates for the drop off location.

ADDITIONAL SAMPLE QUESTIONS

Q1: How are the two parameters of the 'Move Block to Position' command used to tell the helicopter the exact place to drop the ice block?

A1: The X parameter gives the left-to-right location of the drop off spot, and the Y parameter gives the front-to-back (or up-and-down, depending on perspective) location of the drop off spot.

Q2: How does the helicopter know where to go to pick up an ice block?

A2: The helicopter always picks up an ice block from the designated pick-up platform.

Q3: Is it always better to use coordinates rather than specific directions?

A3: No, specific directions are important when the robot should take a specific path to a location.

FAQs

1. For what age groups is this curriculum suitable?

Kids as young as 8 years of age can complete the courses found in Codyssey World, depending on their existing level of STEM proficiency. But the self-paced structure makes it suitable for a fairly broad age range.

2. Can kids complete the courses independently without parental guidance?

CoderZ courses are flexibly designed to be adapt to both independent and parent-led learning. The parent-led experience offers more opportunity for enrichment, but is not required to successfully complete the course.

3. Where can I manage my subscription?

All your account details can be updated under My Account >> Manage Subscription.

4. Where can I find support?

Parents have full access to mission solutions, support articles, and can contact us at Parents@gocoderz.com for any type of problem. Open Office hours for personal online support from one of our friendly pedagogical guides can be scheduled through the membership subscription area, after logging in.

5. Do you have any additional curriculum?

Yes! We have 2 additional full worlds for you to explore: Codabunga, and RoboRover, with +20 more fun courses and ~200 missions. And we are always updating our content, so have a look at the website!

6. What should I do before the first lesson?

Make sure the computers that will be used meet the minimum requirements to run CoderZ.

Review this Teacher's Guide and complete the relevant missions for the first session.

7. Do we need a special computer or software?

CoderZ is only compatible with Chrome browser. It can be downloaded here.

Users should make sure webGL is enabled. If it is not enabled, here is how to enable it.

8. Does this curriculum align to any state or national standards?

Yes. You can see the related standards by logging into the relevant course under Session Plans and and clicking on the Alignment to Standards tab.

Codabunga World

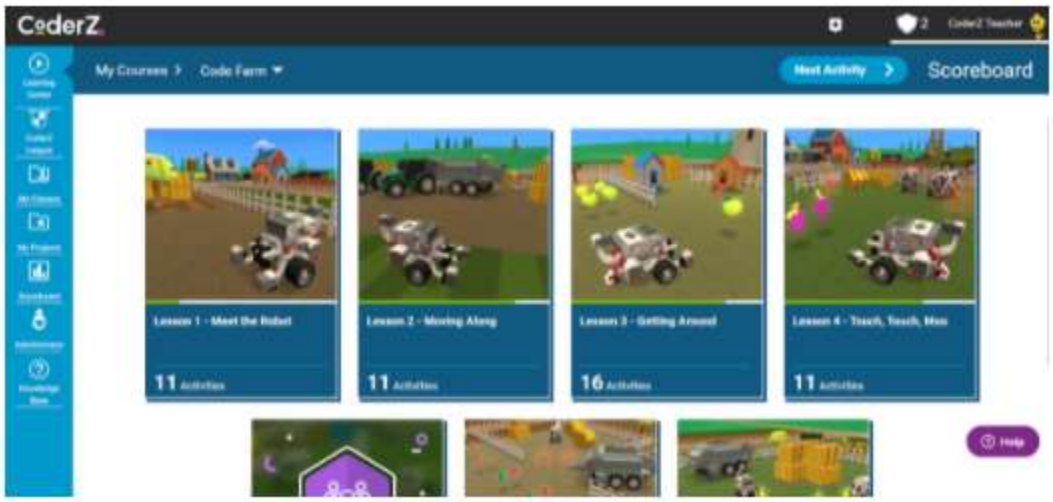
>>Intro to Using this Guide

The courses in Codabunga World are flexibly designed so parents can choose the level of involvement in accompanying their child's learning - from hands-on to completely hands-off. Parents who choose to take a more active role can refer to this guide regularly or as needed to help answer their child's questions and monitor their progress. Each course includes: an introductory overview, opportunities for evaluation and guidance, reflection questions, and additional sample questions.

>>Overview: Codabunga World

This beginner-level curriculum is designed for ages 9 through 12. It teaches kids Blockly-based coding while supporting foundational concepts in math, geometry, formal logic, and physics. In addition to the hard skills covered in the course materials, the curriculum also blends critical life skills including breaking down big problems into manageable parts and considering different solutions to the same problem.

0

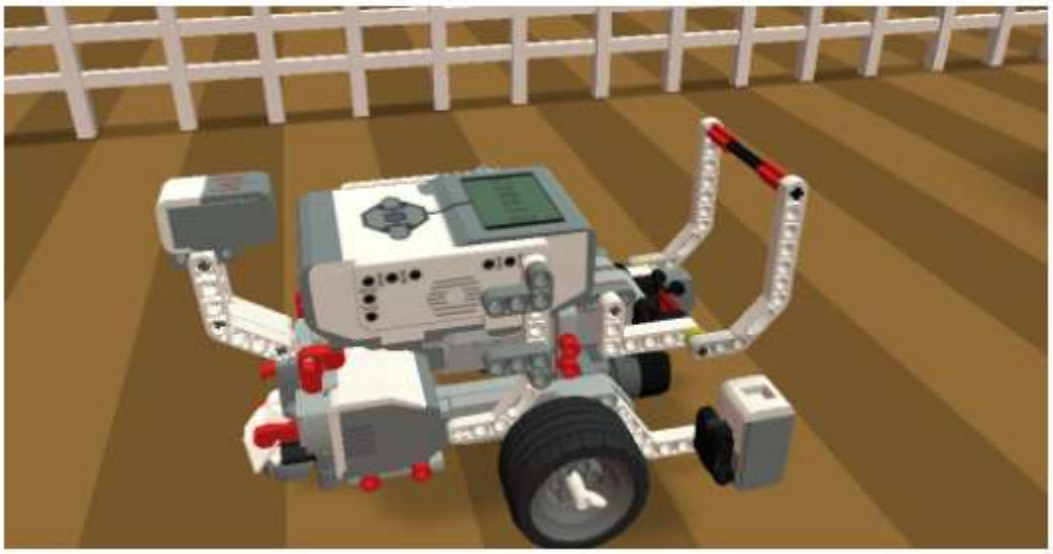


Course 0: Technology Detective

What can we do when technology doesn't work as expected?

Kids are introduced to the CoderZ platform and practice basic troubleshooting skills while logging into their accounts for the first time.

1



Course 1: Code Control

What is a robot?

Kids are introduced to the CoderZ courses and missions, and learn basic navigation skills such as driving the robot back and forth.

2

Course 2: Moove It!

How can I fix problems in code?

Kids learn the concept of 'bugs' and 'debugging' their program, and work on sequencing their code.

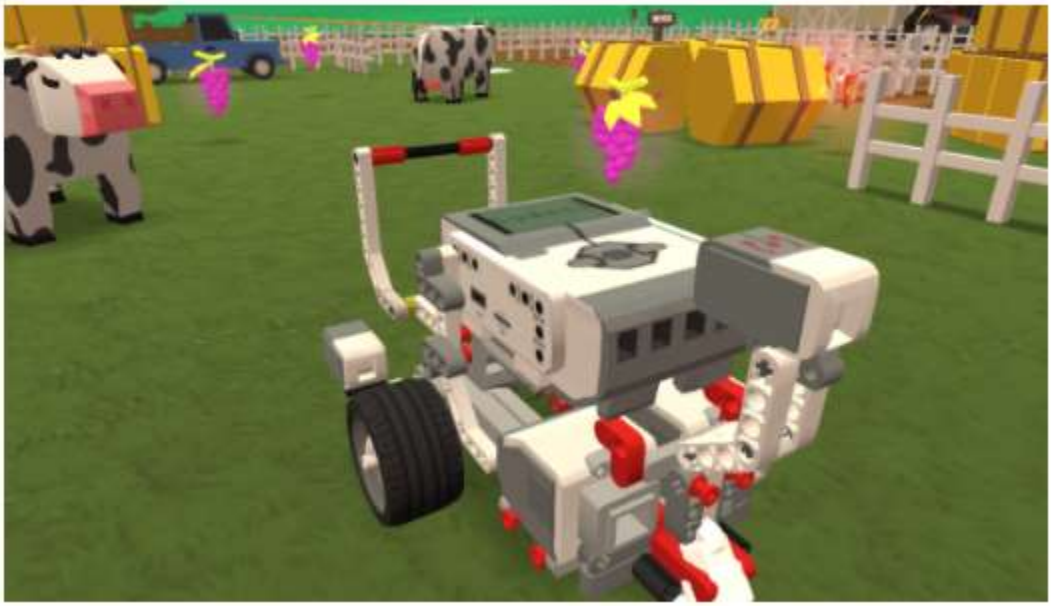
3

Course 3: Spinning Around

How can planning make better programs?

Kids learn the benefits of planning their algorithm before coding, as well as how to perform time-based screw turns.

4

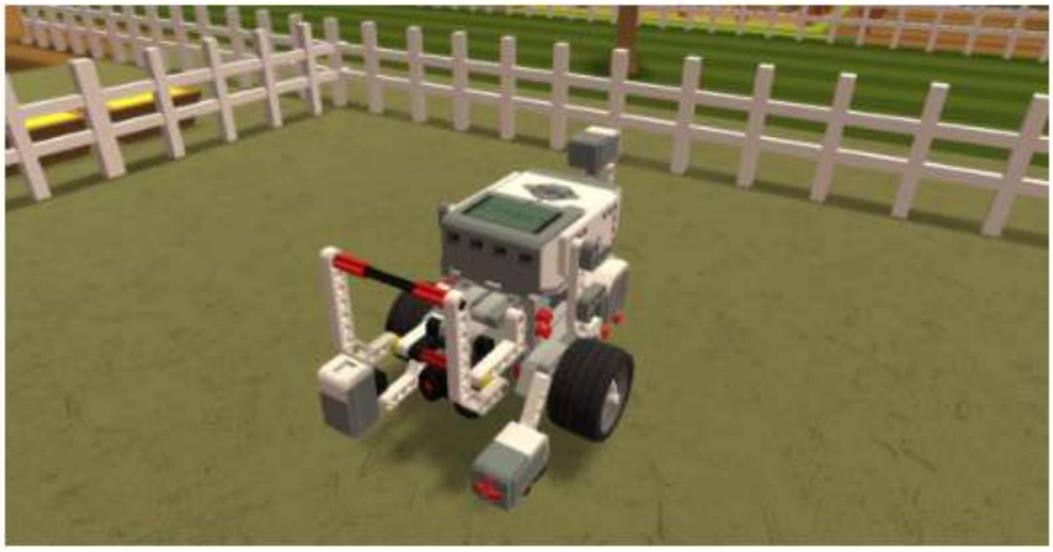


Course 4: Touch, Touch, Moo

How can I use the robot to get information about the environment?

Kids are introduced to the Touch Sensor, and learn to navigate their environment based on physical cues around the robot.

P



Project: Build a Better Robot

How can we make robots work better for everyone?

In this offline project, kids design a robot that they would like to have.

5

Course 5: Gyro's Turn

How can sensors make it easier to code?

Kids are introduced to the Gyro Sensor and learn how to turn based on sensor readings rather than time, and draw on the floor using the Trail blocks.

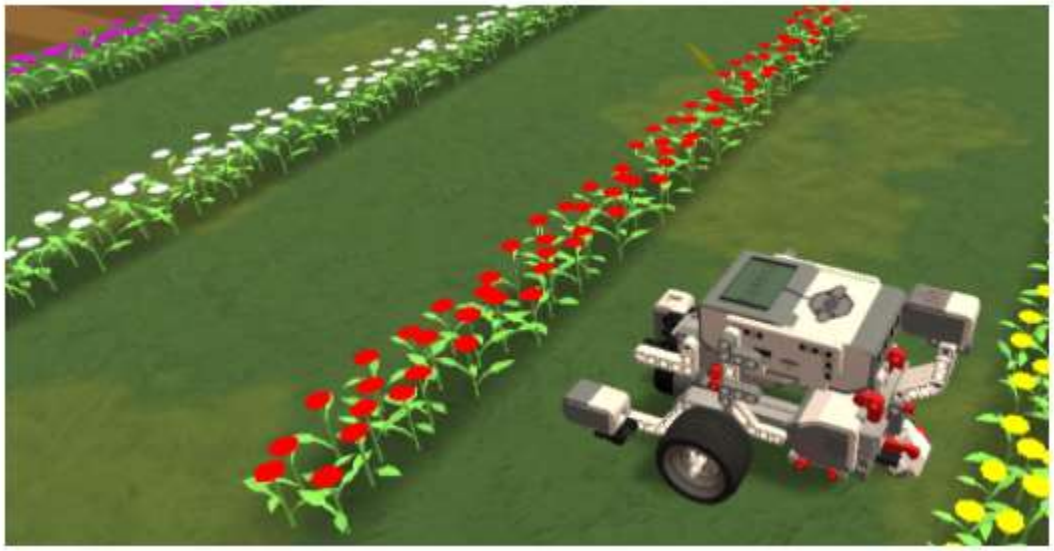
6

Course 6: Loop de Loop

What makes one algorithm better than another?

Kids learn to use Repeat loops and Print statements, thus making their code more efficient and practicing more debugging strategies.

7

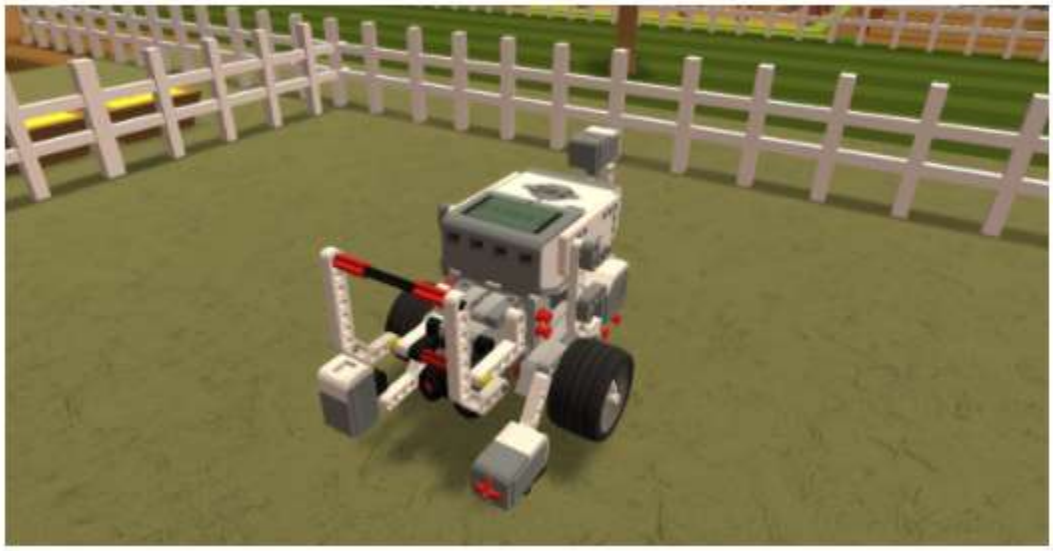


Course 7: Codachrome

How can I change what my robot does based on what it senses in its environment?

Kids learn to use conditionals (If statements) and navigate missions based on the Color sensor.

P



Project: Robots, Unite!

How can robots work together to solve problems?

In this offline project, kids investigate teams of robots that work together to achieve a goal.

>>Course 0: Technology Detective

OVERVIEW

Kids are introduced to the CoderZ platform and practice basic troubleshooting skills while logging into their accounts for the first time and reviewing how to ensure their account security.

>>Course 1: Code Control

OVERVIEW

This lesson introduces kids to basic concepts in robotics and the various features of the CoderZ Cyber Robotics platform.

We begin the lesson by drawing and describing robots and what they can do, then coming up with a working definition of a robot. We then watch a short instructional video about robotics before logging into the CoderZ platform. Kids control the virtual robot using manual control, then practice using the coding interface to control the robot. They then reflect on what they have learned about how hardware and software interact in the virtual robots, as well as the robots they described in the beginning of the lesson.

LEARNING EVALUATION AND GUIDANCE

Define a robot as a machine controlled by a computer

In the last reflection question ("Why do robots need both hardware and software?"), ensure that your child has explained that the robot's software controls what the hardware is doing.

Use arguments to change the way a command runs

Check for mission completion in Missions 4 and 5 of the course, which ask kids to edit and add arguments, respectively. In the second reflection question, ensure that your child has described that arguments change the specific details of a block command.

REFLECTION QUESTIONS

Q1: Think of a robot-ike device you know. What are two code blocks you would need to make it work?

Q2: How do arguments make coding more convenient?

Q3: Why do robots need both hardware and software?

ADDITIONAL SAMPLE QUESTIONS

Q1: What is a robot?

A1: Robots are machines controlled by computers. Robots have moving parts, use input and output to interact with their environments.

Q2: What is the difference between hardware and software?

A2: Hardware is the physical parts of the robot (or computing device), and software is the code that makes it run.

Q3: Why do robots need both hardware and software?

A3: Robots need hardware to interact with the physical world. Hardware allows robots to both take in information (sensors, buttons) and send out information (motors, screens, speakers). Software allows robots to use the information they take in to make decisions about their output.

Q4: Where can you find the blocks you will need to create your program?

A4: The blocks are in the block library on the left side of the screen. Clicking on the different categories allows you to access different types of blocks.

Q5: How do you solve the problem of a block being grayed out and not run as part of the program?

A5: Attach it to the Program Start block (or to a block attached to the Program Start block)

Q6: How do you get rid of blocks that you don't need anymore?

A6: Drag them into the trash can in the lower right side of the Code Editor, or: Right-click the block and select 'Delete block', or: Click the block and then press the Delete key on the keyboard.

Q7: What happens when you use negative numbers for the arguments in the drive block?

A7: The robot moves backwards.

Course 2: Moove It! >>

OVERVIEW

This lesson focuses on developing debugging skills while teaching kids to control the distance that the robot moves forwards and backwards.

We begin by investigating code that contains the 'Wait For' block, then discuss how the code works and what changes we could make to improve the program. We then watch a video on debugging before moving to the CoderZ platform to use what we have learned on a series of missions, which reinforce the concepts of debugging and using arguments in code. The pack culminates in a challenge mission that requires using all these new skills. Lastly, we reflect on how to use what we have learned about debugging and arguments.

LEARNING EVALUATION AND GUIDANCE

Identify and fix errors in a program

In Reflection Question 1, check that your child has identified an actual bug in a program. In Reflection Questions 1 and 2, check that they have described a reasonable process or set of strategies for debugging programs, such as reading through the code, comparing the expected behavior of the robot to its actual behavior, or asking for help.

Use a 'Wait For' command to control program execution

Check for mission completion in the last mission. If your child is unable to complete the last mission, Missions 5 and 7 can be used as alternate assessment activities.

Use sequencing to control program flow

Check for mission completion in the last mission. If your child is unable to complete the last mission, the Parson's problem in Mission 8 can be used as an alternate assessment activity.

REFLECTION QUESTIONS

Q1: Describe a bug you fixed today, explaining how the problem with the code caused the robot to do something unexpected.

Q2: What are three tips for someone who is trying to debug a program?

Q3: Describe a situation where changing the order of a Wait For block changes what the robot does.

ADDITIONAL SAMPLE QUESTIONS

Q1: What does the Wait For block do?

A1: The Wait For block pauses the robot from executing the next commands in the code for the number of seconds or milliseconds dictated by the argument. It does not stop the robot from moving or performing any actions.

Q2: What is a "bug" in regards to code?

A2: A bug is a problem with the code that causes the program to run in a way that the programmer did not intend.

Q3: What is debugging?

A3: Debugging is finding and fixing problems in code.

Q4: Why does it matter that blocks are in the right order?

A4: The robot always executes the commands in the order that they are attached to the "Program Start" block. If they are in the wrong order, the robot will execute the orders as they are sequenced and will not complete the mission successfully.

Q5: Which two blocks are needed to make the robot drive forward for three seconds?

A5: The 'Set Wheel Velocity' block and the 'Wait For' block. The 'Set Wheel Velocity' block starts the robot moving forward, and the 'Wait For' block makes it wait for a certain amount of time before doing something different.

Course 3: Spinning Around >>

OVERVIEW

This lesson builds on kids' understanding of the two motor drives by introducing the screw turn, which turns the robot by moving the left and right wheels at the same speed in opposite directions. Kids also learn how to use comments and better plan their programs, a skill essential to managing the emerging complexity of their code.

We begin with a complex manual mission, then introduce the mechanics of the screw turn and encourage kids to use planning as they address the increasingly complex challenges in the online missions. The progression culminates in a challenging mission that mirrors the manual mission kids began with. We then reflect on the skills that allowed us to successfully complete the course.

LEARNING EVALUATION AND GUIDANCE

Use two-motor drive to move the robot in screw turns

In the last mission, check that your child has used a screw turn. In Missions 13, 14, and 15, screw turns are necessary to reach the target.

Use comments to explain and document code

Check that your child has used comments in Missions 13, 14, or 15. In Reflection Question 1, they should identify two advantages of comments. Answers may include communicating to other programmers on a team, reminding oneself of intentions and considerations taken while planning, or documenting the plan in the code space before coding.

Use a structured process to plan a program

In Reflection Question 3, check that your child has described the four steps of the planning process correctly, and that they have given good examples of how they can be used. Explanations may be relatively trivial ("I saw where the robot needed to go"), but they should distinctly correspond to each step.

REFLECTION QUESTIONS

Q1: What are two ways comments can help you write better programs?

Q2: What are some tips for helping someone plan their program

Q3: Describe a time in this lesson that you used the planning process, explaining how you used each of the four steps?

ADDITIONAL SAMPLE QUESTIONS

Q1: What are the four steps to the planning process?

A1: The four steps are gathering data, explaining the plan, turning the plan into code, and testing the plan.

Q2: What does it mean to comment code?

A2: Commenting code is putting short explanations of the code into the program. Usually comments explain the purpose of the few lines or blocks of code that come right after them.

Q3: Does the robot ever read and execute comments in the code?

A3: No, the comments in the code are just for people to read and use to understand the program.

Q4: What kind of arguments should you put into the Set Wheel Velocity block to make the robot do a screw turn?

A4: The arguments should be the same number, except one should be negative and the other positive.

Q5: Does it matter whether a programmer writes the program first and then puts in comments afterwards, or writes the comments first and then puts the code in afterward?

A5: No, writing the comments before the code can help plan the code, but writing the comments after the code is still helpful to make sure people who read the code later can understand what it does.

Course 4: Touch, Touch, Moo >>

OVERVIEW

In this lesson, kids are introduced to the idea of sensors, which detect information about the robot's environment so that it can be used during program execution.

Kids are first encouraged to reflect on how they use their own senses, and how sensors could help a robot navigate their environment. They then use Manual Control to explore situations in which the touch sensor returns information to the program. After investigating code that uses sensor data to control program flow, they debug and write their own programs that allow the robot to use information from its environment to navigate to the target. After completing a final challenge mission, kids can reflect on how sensors and sensor input allow them to create more complex and flexible programs.

LEARNING EVALUATION AND GUIDANCE

Explain how sensors allow robots to collect information from their environment

In Reflection Question 3, ensure that students mention robots getting information from their environments or interacting with their environments in some way.

Use the touch sensor to determine when the robot is touching an object

Check for mission completion in any one of Missions 5, 14, or 15. Ensure that students have used the Wait Until Touch block rather than the Wait For block to control the program flow.

Use a 'Wait Until' block to control program execution

Check for mission completion in any one of Missions 5, 14, or 15. Ensure that students have used the Wait Until Touch block rather than the Wait For block to control the program flow.

REFLECTION QUESTIONS

Q1: Why is it good to use the sensor, even if you already know the exact path the robot should take?

Q2: Think of another sensor that would be useful for this robot. What sort of information would it collect and how could it use it?

Q3: Why is it important for robots to have sensors?

ADDITIONAL SAMPLE QUESTIONS

Q1: Where is the Touch Sensor located on the robot?

A1: The Touch Sensor is on the front of the robot.

Q2: How is a Wait For block different from a Wait Until block?

A2: The Wait For block pauses the robot from executing the code until a certain amount of time has passed, while the Wait Until block pauses the robot from executing the code until a specific event happens.

Q3: What will the program do if the event the Wait Until block is waiting for never happens?

A3: The program will continue to wait until the programmer or another person stops it from running.

Q4: What are the two values that the Touch Sensor can return?

A4: The touch sensor can return 'True' (when the sensor is pressed) or 'False' (whenever the sensor is not pressed).

Q5: Will the touch sensor always detect when the robot is touching something?

A5: No, the touch sensor will only detect when the sensor itself is pressed.

PROJECT 1: Build a Better Robot >>

OVERVIEW

Kids design a robot that they would like to have, and that they think would be useful to most people in their age range. They explain in what situations they would expect the robot to be used and why the robot is appropriate for that environment. They then read a set of user profiles and evaluate how well their robot would work for that person, and make improvements to the robot to increase usability and accessibility.

TIPS

Q1: Describe one piece of feedback you got and how it improved your design.

Q2: What were three things you needed to think about when making our robot more accessible for everyone?

Q3: How can criteria and constraints make it easier to make decisions about your design?

LEARNING EVALUATION AND GUIDANCE

Use a structured process to design a computing system

In Part 4 of the project guide, ensure that your child has identified key criteria for their robot, and explain how their designs meet those criteria.

Use feedback to improve a design's usability and accessibility

In Parts 2 and 3 of the project guide, ensure that your child has brainstormed changes to their robots that are associated with the given feedback and the identified accessibility concerns.

Document and present a product and its design process

In Part 5 of the project guide, check that your child's presentations include the required information in the presentation checklist.

Use media created by others in responsible and ethical ways

In Part 5 of the project guide, check that your child has identified the license for each of the images that they use. You may also use the exit ticket in Session 4 to ensure they understand that licenses give permission to use creative works.

Course 5: Gyro's Turn! >>

OVERVIEW

This lesson introduces kids to the Gyroscope, a sensor that measures rotational speed or rotational change.

The lesson begins with a review of timed turns, which require kids to calculate the amount of time each turn should take based on their own observations of how far the robot has turned. We then learn about the Gyro sensor and explore the types of values that it returns in a Manual Control mission. As we begin to code turns using the Gyro, we reflect on how the sensor makes turns easier to program. We practice various types of turns, exploring issues related to inequalities, hardware accuracy, and relationships between angles. Last, we are introduced to the Trail block, which allows us to draw on the ground in the Simulation Pane.

LEARNING EVALUATION AND GUIDANCE

Use the Gyroscope to turn the robot to a precise angle

In Mission 9, check for mission completion. This mission asks kids to code both left and right turns of various magnitudes.

Use Start Trail and Stop Trail blocks to draw on the ground in the simulation

In Missions 14 and 15, check that your child has drawn identifiable images on the ground in the simulation.

Use comparison operators to help make decisions in a program

In Mission 10, check for mission completion. You may also use the exit ticket after Mission 10 to ensure that your child understands when to use the 'less than' or 'greater than' operators.

REFLECTION QUESTIONS

Q1: How can the limitations of your hardware change the way you have to code?

Q2: Why does the gyro keep counting up forever, instead of resetting when it turns back to where it started?

Q3: What are two advantages of gyro turns over timed turns?

ADDITIONAL SAMPLE QUESTIONS

Q1: How do you turn off the robot's color trail?

A1: Use the Stop Trail block to turn off the robot's color trail.

Q2: If the robot turns right, will the Gyro sensor count up or down? What about if it turns to the left?

A2: The robot counts up when turning to the right (clockwise) and down when turning to the to the left (counterclockwise).

Q3: What is the difference between positive and negative angles when turning?

A3: Positive angles are measured from left to right (turning right/clockwise), and negative angles are measured from right to left (turning left/counterclockwise).

Q4: How can you change the color of the robot's trail?

A4: Use the Start Trail block and click the color area to change to the new color.

Q5: Why is it important to use an inequality operator (≥ or ≤) rather than an equality operator (=) when using the Wait Until Gyro Angle block?

A5: Because the Gyro might skip over the exact angle, and the program will wait forever.

Course 6: Loop de Loop >>

OVERVIEW

Kids reflect on what can make one algorithm better than another, then use the Repeat block to design, improve, and code algorithms that use repeated patterns.

The lesson starts with a discussion of the two types of turns, time-based and gyro-based, and we explore the advantages of each algorithm within the context of code and comments used, then discuss what an algorithm with repeating patterns would ideally look like. Afterwards, we are introduced to the Repeat block and practice using it to code increasingly complex patterns. We then use the Reset Gyro block to fully leverage the power of the Repeat loop in drawing their

designs, and use the Print block to get more insight into what the Repeat block is doing. Finally, we identify the various ways that one algorithm could be better than another.

LEARNING EVALUATION AND GUIDANCE

Compare the advantages and disadvantages of similar algorithms.

In Reflection Questions 2 and 3, check that your child has identified at least three key ways that one algorithm can be better than another. They should identify at least one advantage related to the performance of the algorithm and one related to the ease with which programmers can understand, maintain and modify the code.

Use a Repeat loop to make code more manageable

In Missions 8, 9, and 10, check that your child has used Repeat loops to complete each mission.

Use a Print block to print text to the console

In Reflection Question 1, ensure that your child understands that the Print block prints to the console, rather than affects the robot's behavior. You can also check that they have used a Print statement inside the loop in Missions 9 and 10.

REFLECTION QUESTIONS

Q1: How can Print statements help you to debug?

Q2: What are two advantages to using a Repeat block versus writing the code over and over?

Q3: What are three things that can make one algorithm better than another?

ADDITIONAL SAMPLE QUESTIONS

Q1: Where does the information attached to the Print block appear on the screen when printed?

A1: In the console underneath the Code Editor.

Q2: How many code blocks can go into a Repeat loop?

A2: There is no limit to the number of blocks that can go into a Repeat loop.

Q3: Why is the shape of a Repeat block different from other blocks?

A3: Because the blocks to be repeated need to go inside of it.

Q4: If two algorithms work equally well to instruct the robot, does that mean that they are equally good?

A4: No, one algorithm might be easier for programmers to understand, maintain, and debug.

Q5: What does the Reset Gyro block do?

A5: It resets the Gyro's measurement to zero.

Course 7: Codachrome >>

OVERVIEW

This lesson introduces both the Color sensor and conditionals (the 'If' block).

We begin with a Manual Control mission that asks kids to use color as a directional guide. The Color sensor is introduced, and kids use it in conjunction with a Wait Until block to guide the robot to the target in an unpredictable environment. We then use the Color sensor with If statements, which check for a particular condition and execute code only if that condition is true, responding to increasingly complex environments. Last, we reflect on the different ways that we are able to instruct the robot to make decisions while the program is running.

LEARNING EVALUATION AND GUIDANCE

Use conditionals to make decisions as a program is running

Check that your child has successfully used If statements in Missions 9 or 10.

Use the Color sensor to detect colors in the environment

Check that your child has successfully used the Color sensor in Missions 4, 5, 9 , or 10.

REFLECTION QUESTIONS

Q1: Now that you can instruct the robot to make decisions while the program is running, do you think they need more or less planning to run your programs? Why?

Q2: What is one way the Color sensor is the same as other sensors that you have used (Touch sensor, Gyroscope) and what is one way that it is different?

Q3: Why should conditions always be 'True' or 'False' rather than a number value such as '8' or a color value such as 'red'?

ADDITIONAL SAMPLE QUESTIONS

Q1: What's the difference between using the Wait Until block and using the If block?

A1: The Wait Until block stops executing the program and waits until the condition is true. The If block checks once, then executes the code inside the block only if the condition is true, then moves on regardless.

Q2: Where is the Color sensor located on the robot?

A2: The Color sensor is toward the front of the robot, facing the ground beneath it.

Q3: Can the Color sensor sense every color?

A3: The Color sensor can distinguish among ten different colors (black, white, gray, red, yellow, green, blue, cyan, magenta, and brown). If it senses a color outside of those ten colors, it will return the closest color in its list.

Q4: What can you do if you want to check for two different things?

A4: You can use two different If blocks.

Q5: How many times will an If block check for the condition?

A5: The If block will only check one time, unless it is inside a Repeat loop.

Project 2: Robots, Unite! >>

OVERVIEW

Kids investigate teams of robots that work together to achieve a goal. They then model communications protocols that allow robots to communicate with each other over networks, and explore how these communications can be kept private. Last, they design their own team of robots and explain how their robots work together to solve a problem.

LEARNING EVALUATION AND GUIDANCE

Explain how multiple computing systems can work together to achieve a goal

On page 3 of the worksheet, check that your child has explained how the robots work together. In the final presentation, ensure that they have explained how their own team of robots works together.

Model how packets are used to send information over a network

Check that your child has broken up the content of their messages and placed them in the 'body' portion of the packet, and that the information needed to put the messages back together is in the 'header' portion of each packet.

Describe ways information can be protected in computing systems and on a network

In the exit ticket for 'Robots, Defend!', check that your child has listed at least two reasonable ways to keep information safe. These can include digital security measures such as encryption or physical security measures such as keeping devices stored where others cannot access them.

Document and present a product

In 'Robots, Present!' section of the project guide, check that their poster includes the required information in the poster checklist.

REFLECTION QUESTIONS

Q1: What is one way a robot team's communication is the same as a human team, and one way it's different?

Q2: What problems might occur if someone else could read the communication of one of the robot teams?

Q3: What are three important things to remember if you want robots to work as a team?

FAQs

1. For what age groups is this curriculum suitable?

Kids as young as 9 years of age can complete the courses found in Codabunga World, depending on their existing level of STEM proficiency. But the self-paced structure makes it suitable for a fairly broad age range.

2. Can kids complete the courses independently without parental guidance?

CoderZ courses are flexibly designed to adapt to both independent and parent-led learning. The parent-led experience offers more opportunity for enrichment, but is not required to successfully complete the course.

3. Where can I manage my subscription?

All your account details can be updated under My Account >> Manage Subscription.

4. Where can I find support?

Parents have full access to mission solutions, support articles, and can contact us at Parents@gocoderz.com for any type of problem. Open Office hours for personal online support from one of our friendly pedagogical guides can be scheduled through the membership subscription area, after logging in.

5. Do you have any additional curriculum?

Yes! We have 2 additional full worlds for you to explore: Codabunga, and RoboRover, with +20 more fun courses and ~200 missions. And we are always updating our content, so have a look at the website!

6. What should I do before the first lesson?

Make sure the computers that will be used meet the minimum requirements to run CoderZ.

Review this Teacher's Guide and complete the relevant missions for the first session.

7. Do we need a special computer or software?

CoderZ is only compatible with Chrome browser. It can be downloaded here.

Users should make sure webGL is enabled. If it is not enabled, here is how to enable it.

8. Does this curriculum align to any state or national standards?

Yes. You can see the related standards by logging into the relevant course under Session Plans and and clicking on the Alignment to Standards tab.

Robo Rover World

>>Intro to Using this Guide

The courses in Robo Rover World are flexibly designed so parents can choose the level of involvement in accompanying their child's learning - from hands-on to completely hands-off. Parents who choose to take a more active role can refer to this guide regularly or as needed to help answer their child's questions and monitor their progress. Each course includes: an overview of the learning goals for each course; resources such as video tutorials and related articles; and Q&As for review.

>> Overview: Robo Rover World

This is an intermediate-level curriculum designed for ages 11 and up. It introduces kids to next-level coding and robotics and builds foundational knowledge in geometry, engineering, and physics. In addition to the hard skills covered in the course materials, the curriculum also blends critical life skills such as: combining multiple strategies to solve complex problems, the importance of planning, and the power of persistence!

1

Course 1: STEM Is a Snap

Overview of STEM and first steps in CoderZ learning environment.

2

Course 2: Ahead of the Curve

Drive the robot! Learn about drive systems and how to navigate your robot using computer code.

3

Course 3: Circtacular

Learn EVEN MORE about drive systems and how to navigate your robot using computer code!

4

Course 4: Sense It

Learn how to use the Robot's touch sensor for autonomous navigation using basic coding blocks.

5

Course 5: Repeat After Me

Learn how to code more efficiently with the Repeat loop.

6

Course 6: Heads Up!

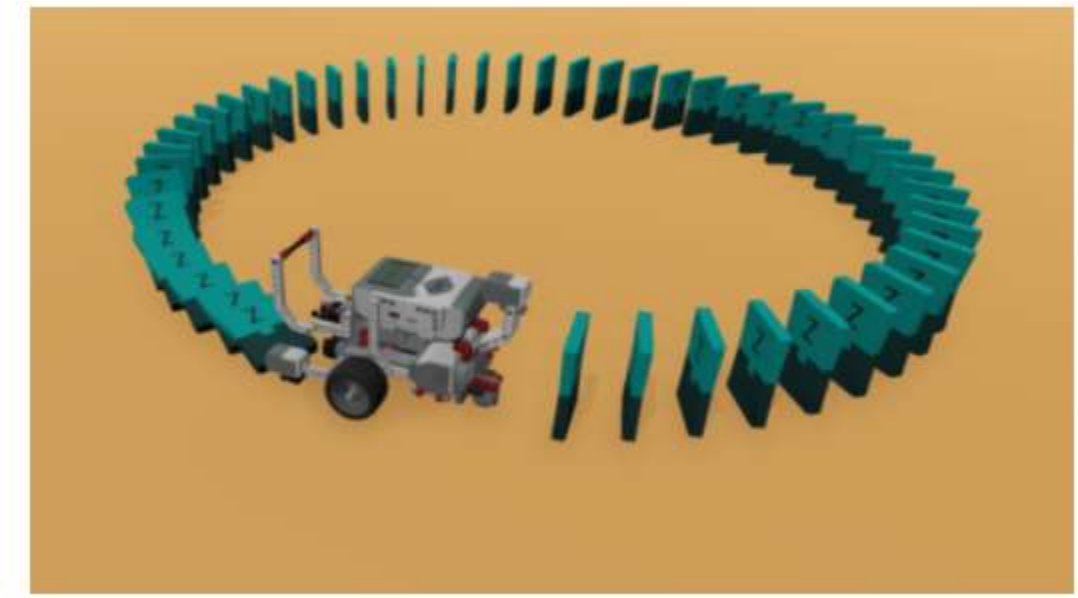
Learn how to make accurate turns using data from the Gyroscopic sensor.

7

Course 7: Gyro Returns

Learn how to make accurate turns using data from the Gyroscopic sensor and use of reset gyro.

8



Course 8: Domino Creations

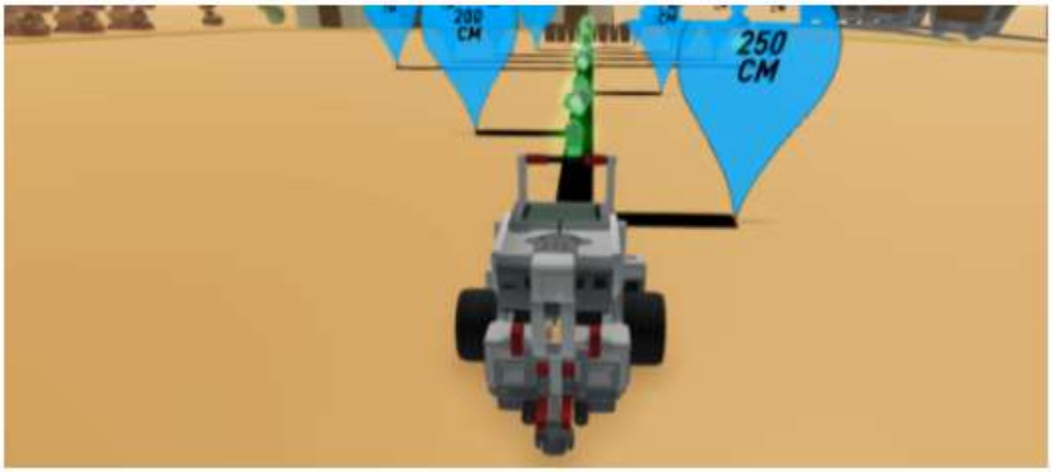
Time to use all your creativity and imagination together with all you've learned so far and take on fun challenges that puts your skills to the test.

9

Course 9: Mission Unleashed

Apply all you've learned so far and take on advanced challenges that put your skills to the test.

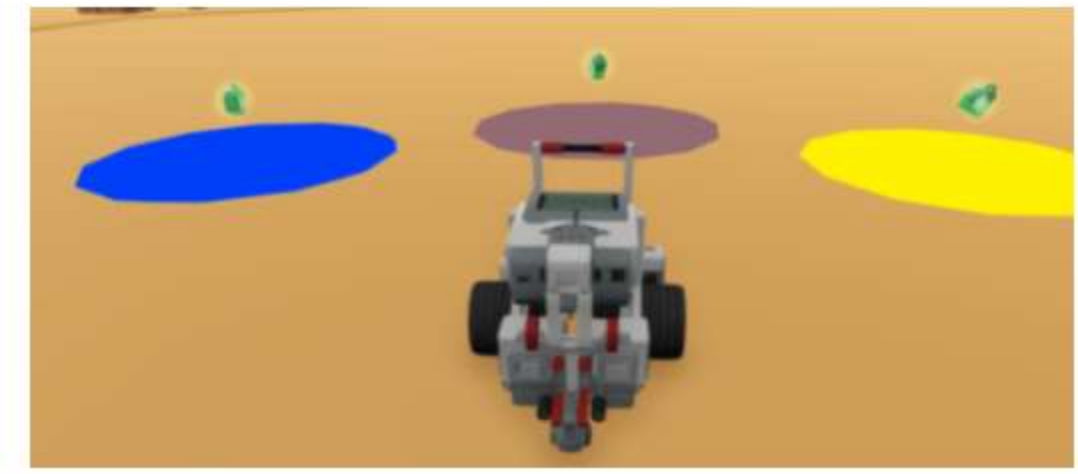
10



Course 10: Supersonic

Learn how to avoid obstacles by sensing them from afar. The Ultrasonic sensor is the way to go.

11



Course 11: Color Me Smart

The robot can detect colors on the floor and use it to make better decisions. It’s all about waiting for the right moment.

12

Course 12: Mission Rising

More advanced challenges to put your skills to the test.

13

Course 13: A Call to Arms

Control the robot's arm to interact with objects in the scene and solve complex challenges.

14

Course 14: What If?!

Use the sensors so your robot can make informed decisions in real time.

15

Course 15: Mission Master

Implement all you've learned in this series of complex challenges.

>>Course 1: STEM Is a Snap

LEARNING GOALS

By the end of this lesson kids will:

- Understand the meaning of STEM, and how the subjects are integrated.
- Have a basic comprehension of what robots are.
- Analyze technological systems based on their inputs and outputs.
- Login to CoderZ and complete basic navigation missions.

Q&A FOR REVIEW

>>Course 1: STEM Is a Snap

Q1. What is STEM?

A1. STEM is an educational approach that combines science, technology, engineering, and math.

Q2. What is a robot?

A2. A robot is an automated problem solver.

Q3. What are inputs and outputs?

A3. An input is information that is received by a sensor. An output is a controlled action that is carried out (this can be both according to information received by the sensor, or not).

>>Course 2: Ahead of the Curve

LEARNING GOALS

By the end of this lesson kids will:

- Identify the key components of a drive system.
- Explain how the virtual robot can drive.
- Use computer code to drive the virtual robot around.
- Explain how to perform various turns by changing parameters in their code.

RESOURCES

- **Drive block support article**
- **Wait For block support article**
- **YouTube steering tutorial**

TIPS

1. This is a double session – it will require 2 hours to complete with lesson 3: Circtacular
2. Begin with the slideshow, taking breaks to complete the missions whenever prompted by the slides.
3. The missions might take longer, no need to rush!
4. Check out the help widget.
5. Go through the additional resources such as the aforementioned YouTube steering tutorial and support article.

Course 3: Circtacular >>

LEARNING GOALS

By the end of this lesson kids will:

- Identify the key components of a drive system.
- Explain how the virtual robot can drive.
- Use computer code to drive the virtual robot around.
- Explain how to do various turns by changing parameters in their code.

RESOURCES

- **Drive block support article**
- **Wait For block support article**
- **YouTube steering tutorial**

TIPS

6. This is a double session – it will require 2 hours to complete with lesson 2: Ahead of the Curve
7. The missions might take longer, no need to rush!

8. Check out the help widget.
9. Go through the additional resources such as the aforementioned YouTube steering tutorial and support article.

Q&A FOR REVIEW

Q1. How many motors does the robot have?

A1. Two motors.

Q2. How does the robot use the motors to drive forward and turn?

A2. Forward: Both motors drive at the same speed.

Turning: One motor drives slower than the other.

Q3. Which way does the robot turn with negative steering?

A3. Left.

Course 4: Sense It >>

LEARNING GOALS

By the end of this lesson kids will:

- Have a basic understanding of what sensors are used for.

- Know what sensors are used with our virtual robot.
- Differentiate between robotic input and output processes.
- Understand how to code a combination of sensor-related blocks and navigational blocks.

RESOURCES

- **Wait Until support article**
- **Touch Sensor support article**
- **The Wait Blocks video tutorial**
- **Smart Blocks video tutorial**

Q&A FOR REVIEW

Q1. What is the input of the touch sensor?

A1. The input is contact with an object.

Q2. The touch sensor returns Boolean values, what does this mean?

A2. A Boolean value is true/false.

Q3. What block uses the touch sensor?

A3. Wait Until Touch block.

Course 5: Repeat After Me >>

LEARNING GOALS

By the end of this lesson kids will:

- Define what a loop is.
- Comprehend what a repeat loop is, what it does, and how it works.
- Create even more complex programs with just the drive block and loops.

RESOURCES

- **Repeat block support article**

Q&A FOR REVIEW

Q1. What does a repeat loop do to the code inside it?

A1. It repeats the code however many times it is set to run.

Q2. What is an iteration, regarding the repeat loop block?

A2. An iteration is the number of times the code will run.

Q3. What happens if you take code that is in a repeat loop of 4 iterations, and put its loop in a loop of 2 iterations?

A3. The code inside the first loop will run 8 times.

Course 6: Heads Up!>>

LEARNING GOALS

By the end of this lesson kids will:

- Identify the purpose and function of a gyro sensor.
- Understand what rotational speed is.
- Apply the gyro sensor and its accompanying commands to a CoderZ program.
- Create a program in which the robot completes accurate turns using the robot's gyro sensor.
- Understand the effect of speed on accuracy of turns.

RESOURCES

- Wait Until support article
- The Wait Blocks video tutorial
- Gyro sensor support article

Q&A FOR REVIEW

Q1. Where is the gyroscopic sensor located on the robot?

A1. At the rear of the robot, at the top.

Q2. What does the gyro sensor detect?

A2. Degrees turned and turn rate (degrees per second).

Q3. If the robot turns right, will the gyro sensor count up or down?

A3. Down.

Course 7: Gyro Returns

LEARNING GOALS

By the end of this lesson kids will:

- Identify the purpose of resetting the gyro sensor.
- Understand the effect of rotation speed on the cumulative loss of accuracy.
- Apply the gyro reset block to a CoderZ program.
- Create a program in which the robot completes accurate turns using the robot's gyro sensor, and gyro reset block.

RESOURCES

- **Wait Until support article**
- **The Wait Blocks video tutorial**
- **Gyro sensor support article**

Q&A FOR REVIEW

Q1. Which category is the 'Gyro Reset' block found in?

A1. In the 'Sensor' category.

Q2. If the robot turns 70 degrees to the left, and then 86 degrees to the right, and then runs a gyro reset block, what will its gyro read at the end of the code?

A2. Zero degrees.

Q3. In a repeat loop that uses a 'Wait Until Gyro' block, should the 'Gyro Reset' block come *before*, or *after* the 'Wait Until Gyro' block?

A3. It doesn't matter. It can be either before or after.

Course 8: Domino Creations >>

This course offers a fun creative break. Enjoy!

Course 9: Mission Unleashed >>

LEARNING GOALS

This lesson will be based around a series of challenge missions that test kids' understanding and ability to code virtual robots at a basic level.

Course 10: Supersonic >>

LEARNING GOALS

By the end of this lesson kids will:

- Identify the purpose and function of the ultrasonic sensor.
- Apply the ultrasonic sensor and its accompanying commands to a CoderZ program.
- Create a program in which the robot measures distances and avoids obstacles through use of the ultrasonic sensor.
- Be able to combine other sensors and commands of the robot with the use of the ultrasonic sensor.

RESOURCES

- **Wait Until support article**
- **The Wait Blocks video tutorial**

- **Ultrasonic sensor support article**

TIPS

Use the ‘U’ key to show the Ultrasonic beam in the simulation.

Q&A FOR REVIEW

Q1. What does the ultrasonic sensor detect?

A1. Distance.

Q2. What is the maximum distance the ultrasonic sensor can detect?

A2. 250cm.

Q3. Which direction does the ultrasonic sensor face?

A3. Forward.

Course 11: Color Me Smart >>

LEARNING GOALS

By the end of this lesson kids will:

- Identify the purpose and function of the color sensor, focusing on Color ID readings.

- Apply the color sensor and Color ID commands to a CoderZ program.
- Navigate missions according to various colors throughout the scene.

RESOURCES

- **Color sensor support article**
- **Wait Until Color block support article**

TIPS

Use the ‘C’ key to show the Color sensor's beam in the simulation.

Q&A FOR REVIEW

Q1. What does the color sensor detect?

A1. Color IDs. (And red values as a bonus)

Q2. Where is the color sensor located on the robot?

A2. Front right.

Q3. How many Color IDs can the color sensor detect?

A3. Seven.

Course 12: Mission Rising >>

LEARNING GOALS

This lesson will be based around a series of challenge missions that test kids' understanding and ability to code virtual robots at a basic level.

Course 13: A Call to Arms >>

LEARNING GOALS

By the end of this lesson kids will:

- Learn to control the arm using rotateTo block.
- Set the correct angle to lift or lower the arm.
- Use the arm to grab and move objects.
- Use the arm to flip switches on and off.

TIPS

You can control the arm using the manual control by pressing the < and > keys, or use the virtual controls.

Q&A FOR REVIEW

Q1. What can we use the arm for?

A1. We can use the arm to interact with objects such as boxes and switches.

Q2. How do we control the arm?

A2. To control the arm we set a degree for the motor to rotate using the Rotate To block.

Course 14: What If?! >>

LEARNING GOALS

By the end of this lesson kids will:

- Understand the structure of an If statement.
- Be able to set conditions based on data from sensors.
- Add Else option to the If block.
- Follow a line using a basic If statement.
- Know why, when and how to use a Repeat Forever loop in conjunction to an If block.

RESOURCES

- **If block support article**

- **Color sensor support article**
- **Repeat Forever block support article**

TIPS

Make sure kids understand that If is not a loop, it runs only once - unless in a loop.

Remind kids of the ColorID values:

Color	Black	Blue	Green	Yellow	Red	White	Brown
ColorID	1	2	3	4	5	6	7

Q&A FOR REVIEW

Q1. What does the If block require?

A1. A condition and an action to do for when the condition is met.

Q2. What is the Else for?

A2. Else can be used to determine the action (or actions) to perform when the condition is NOT met.

Q3. How can we check the condition repeatedly?

A3. We can place the If block inside a loop, like the Repeat Forever loop.

Course 15: Mission Master >>

LEARNING GOALS

This lesson will be based around three parts of one big challenge mission. that tests kids' understanding and ability to code virtual robots at an intermediate level.

FAQs

1. For what age groups is this curriculum suitable?

Robo Rover World is an intermediate-level curriculum intended for ages 11 and up, depending on the existing level of STEM proficiency. Kids will need to have familiarity with more advanced concepts in math, but do not need prior knowledge in coding or robotics to participate.

2. Can kids complete the courses independently without parental guidance?

CoderZ at Home courses are flexibly designed to adapt to both independent and parent-led learning. The parent-led experience offers more opportunity for enrichment, but is not required to successfully complete the course.

3. Where can I manage my subscription?

All your account details can be updated under My Account >> Manage Subscription.

4. Where can I find support?

Parents have full access to mission solutions, support articles, and can contact us at Parents@gocoderz.com for any type of problem. Open Office hours for personal online support from one of our friendly pedagogical guides can be scheduled through the membership subscription area, after logging in.

5. Do you have any additional curriculum?

Yes! We have 2 additional full worlds for you to explore: Codyssey and Codabunga, with 20+ more fun courses and ~200 missions. And we are always updating our content, so have a look at the website!

6. What should I do before the first lesson?

Make sure the computers that will be used meet the minimum requirements to run CoderZ.

Review this Teacher's Guide and complete the relevant missions for the first session.

7. Do we need a special computer or software?

CoderZ is only compatible with Chrome browser. It can be downloaded here.

Users should make sure webGL is enabled. If it is not enabled, here is how to enable it.

8. Does this curriculum align to any state or national standards?

Yes. You can see the related standards by logging into the relevant course under Session Plans and and clicking on the Alignment to Standards tab.