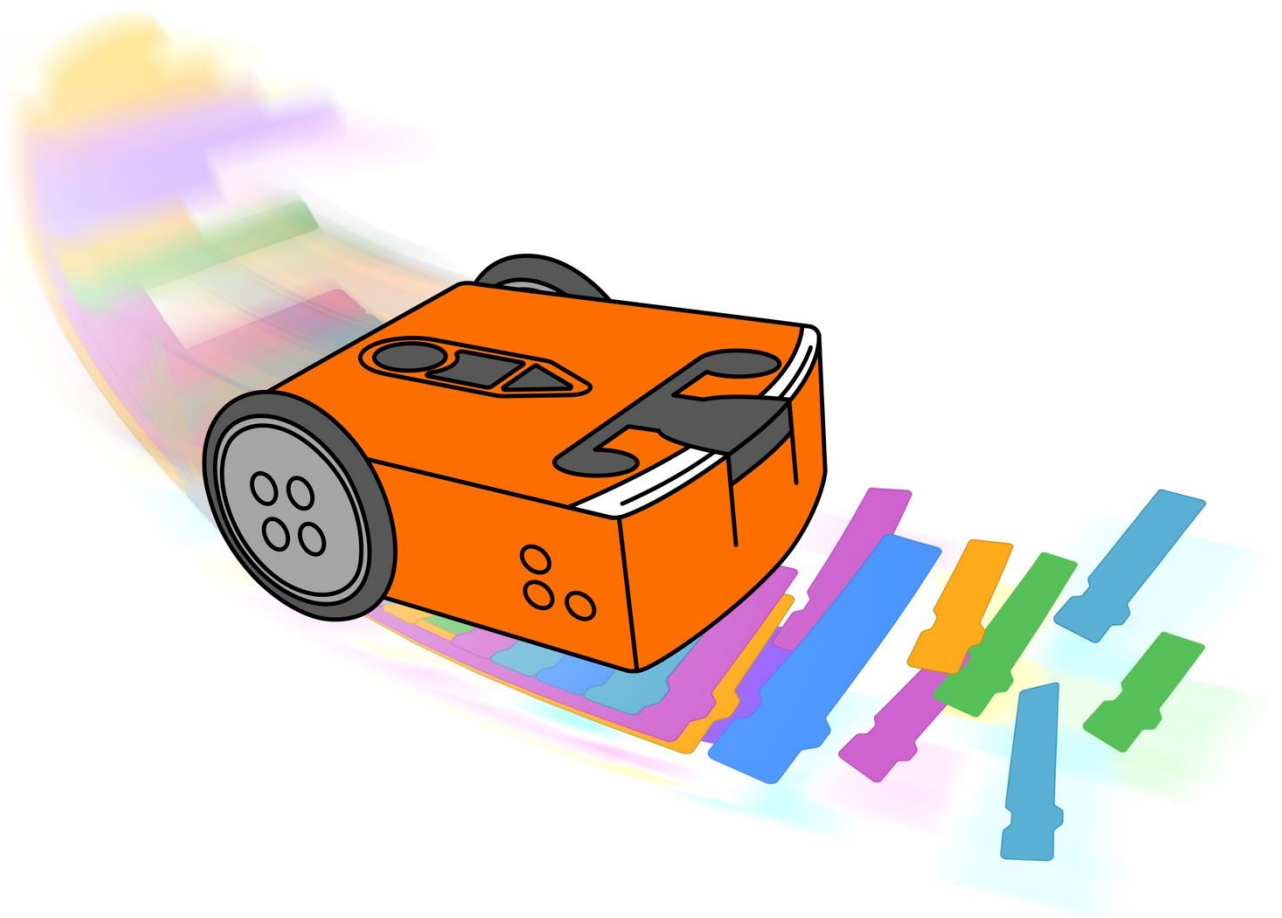




EdScratch lesson activities

Student worksheets and activity sheets

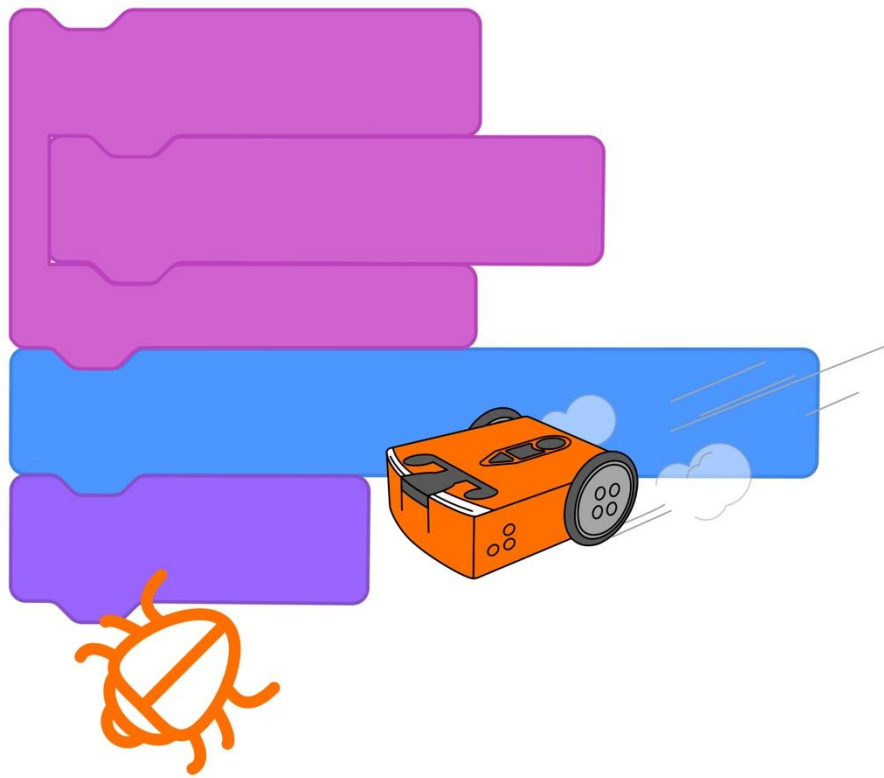


The EdScratch Lesson Plans Set by [Kat Kennewell](#) and [Jin Peng](#) is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](#).

Contents

Unit 2: Move it!	3
U2-1.1 Let's explore how computers 'think'	4
U2-1.1a Change it up: Make a PBJ sandwich	7
U2-1.1b Change it up: Human robots	8
U2-1.2 Let's explore going step-by-step in EdScratch	9
U2-1.3 Let's explore driving Edison	11
U2-1.3a Challenge up: Maze madness	13
U2-1.3b Challenge up: Self-walking pet	14
U2-2.1 Let's explore Edison's outputs	15
U2-2.1a Challenge up: Drive the maze safely	19
U2-2.2 Let's explore input parameters	20
U2-2.2a Change it up: Teach Edison to count to 9	22
U2-2.2b Challenge up: Teach Edison to count to 9 out loud	25
U2-2.3 Let's explore Edison's musical talents	26
U2-2.3a Change it up: Play a song in a round	30
U2-2.3b Challenge up: You are the conductor	31
U2-2.4 Let's explore bugs and debugging	32
U2-2.5 Let's explore Edison's motors	36
U2-2.5a Challenge up: Spinning garden	41
U2-2.5b Challenge up: Spinning solar system	42
U2-2.5c Challenge up: Cartographer and navigator	43
U2-2.5d Challenge up: Writer and director	44
Activity sheet U2-1: Go step-by-step	45
Activity sheet U2-2: Driving track	45
Activity sheet U2-3: Mini maze	45
Activity sheet U2-4: Digital display 2	45
Activity sheet U2-5: Digital display 5	45
Activity sheet U2-6: Digital display 7	45
Activity sheet U2-7: Digital display 8	45

Unit 2: Move it!



U2-1.1 Let's explore how computers 'think'

Imagine a computer. Now imagine a person. Which one do you think is smarter?

This is a bit of a trick question. Believe it or not, most computers cannot do anything without help from people.



Why is that?

Both computers and people can follow instructions, but people can also think for themselves. We can learn and change what we do based on new knowledge.

Most computers, however, cannot do that. An Edison robot, for example, cannot think for itself. It can only follow instructions. Where do those instructions come from? A person like you!

People give computers instructions by giving them computer programs.

To make a good computer program for our Edison robot, or any computer, we need to write that program in a way that the computer can understand. To do this, we need to try to think about things as if we were a computer.

This kind of thinking is called **computational thinking**.



Don't forget

A **computer program** is a collection of instructions that tell a computer to perform a specific task.



Jargon buster

Computational thinking means thinking about a problem or task similar to how a computer thinks. It is a way of logically working through problems, breaking them down into smaller pieces, finding patterns, and then using the information to come up with a step-by-step solution.

In other words, computational thinking is a way of planning, problem-solving and analysing information the same way a computer does.

Whenever you want to write a program for your Edison robot, you need to use computational thinking to help you work out what to do. By learning to think in a way that will make sense to Edison, you will be able to give the robot instructions to get it to do what you want.

One of the most important things about giving instructions to Edison is the order in which you give the instructions.

The importance of going step-by-step

Computers, including Edison robots, are very good at following the instructions that we give them as computer programs. In fact, an Edison robot will follow the instructions in a program *exactly* as they are written. That's why one of the most important parts of computational thinking is using **sequence**.



Jargon buster

Sequence means going in order, step-by-step.

Imagine you want to bake a cake. You might look up a recipe in a cookbook. To make the cake, you would then follow each step one by one. That's sequence!

Whenever you write a program for Edison, you will need to use sequence in the same way. You need to tell Edison exactly what to do, in the exact order you want the robot to do each step.

Task 1: Follow step-by-step

If your teacher tells you to go to the door, what actions do you have to take to get there? You probably don't think about how many steps you need to take. You just do it! If there is a desk in your way, you simply turn and walk around it.

That's not how a robot works. To get your robot to the door, you would need to give it very careful instructions with each step explained one by one. In other words, you would need to tell the robot each action you want it to take in sequence.

Thinking about doing something sequentially like this takes some practice. People are so good at 'just doing' things, we don't usually think about what it is that we are doing broken out into each and every step.

Try following some exact step-by-step directions to see how it feels. Use activity sheet U2-1 to answer the following questions.

1. Start on the ice cream cone, pointing towards the heart. Turn right. Move forwards 2 squares. Where are you?

2. Start on the panda bear, pointing towards the bicycle. Move backwards 1 square. Turn left. Move forward 2 squares. Turn right. Move forwards 1 square. Where are you?

3. Start on the star, pointing towards the cat. Turn left. Turn left again. Move backwards 2 squares. Turn right. Move forward 1 square. Turn right. Move forwards 1 square. Turn left. Move backwards 2 squares. Where are you?

Task 2: Give step-by-step instructions

Let's practice giving exact instructions, describing each item step-by-step. Use activity sheet U2-1 to answer the following questions.



Hint!

Use these commands to write your answers:

move forwards

move backwards

turn left

turn right

4. Write directions for this: start on the rainbow, pointing towards the dog. End on the bird.

5. Write directions for this: start on the rainbow, pointing towards the dog. Do NOT touch the dog. Do NOT touch the cat. End on the bird.

6. Write directions for this: start on the diamond, pointing towards the beachball. Do NOT use any 'move forwards' commands. End on the beachball.

U2-1.1a Change it up: Make a PBJ sandwich

Do you know how to make a peanut butter and jam sandwich? Could you teach a robot?

There are lots of things that you know how to do so well that the task seems very easy. These 'easy tasks' can actually be made up of lots of individual steps, however. Making a peanut butter and jam sandwich is a good example. To make the sandwich, you have to do many different things, and you have to do each one in the right **sequence** to end up with a sandwich.



Jargon buster

Sequence means going in order, step-by-step.

Task 1: Write down how to make a sandwich step-by-step

1. To make a sandwich, what do you need to do? Write down each step sequentially.

Task 2: Follow the directions

Your teacher will help you set up for this task.

Make a sandwich following the directions that have been written down. Be sure you follow the directions EXACTLY as they are written. Don't add any extra steps!

2. How did your sandwich turn out?

U2-1.1b Change it up: Human robots

There are lots of things you know how to do that you think are easy but, actually, have lots of individual steps. Robots cannot lump individual steps together very well. Instead, robots need to have clear instructions that lay out every single move step-by-step. Explaining what needs to happen with clear instructions which are in **sequence** is important to get a robot to do what you want.



Jargon buster

Sequence means going in order, step-by-step.

Task 1: Write down the step-by-step instructions

Your teacher will help you set up for this task.

1. To get the human robot to the goal, what do you need to do? Write down each step sequentially.

Task 2: Follow the directions

Your teacher will help you set up for this task. Be sure you follow the directions EXACTLY as they are written. Don't add any extra steps!

2. Did the human robot reach the goal? Did you need to change anything in your directions to get the instructions to work?

U2-1.2 Let's explore going step-by-step in EdScratch

When you write a program for your Edison robot in EdScratch, you are writing the instructions that will tell the robot what to do and in what order to do each thing. Each EdScratch block is one action you are telling the robot to take.

The order in which you connect the blocks together in your program tells the robot in what sequence to do each action. Edison will do the actions in order, one at a time, starting with the top block.

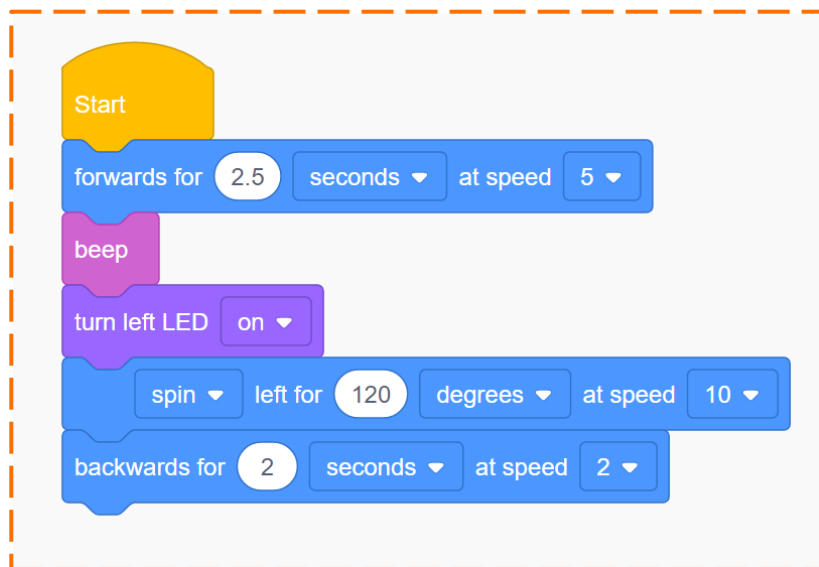


Don't forget

Sequence means going in order, step-by-step.

Task 1: What will Edison do?

Look at this EdScratch program:



All EdScratch programs need to have the yellow **Start** block at the very top. This lets the robot know that the program starts with the first block below the **Start** block.

Read each line block by block, starting with the top block below the **Start** block.

1. If you download this program to Edison, what will the robot do? Be sure to write your answer in the correct sequence.

Task 2: Write the program

Go to the EdScratch app on your computer and write the program from the picture. Find each block in the block pallet and drag it into the programming area.



Hint!

You can change the numbers in a block by clicking on the number and changing it using your keyboard.

You can change a drop-down item in a block by clicking on the down arrow and selecting the option you want.

Make sure that you put all the blocks in the correct sequence in your program.

Once you have written your program, download it to your Edison robot. Run the program and watch what Edison does.

Look at what you wrote down about what you thought the robot would do in your answer in task one. Compare what Edison does when you run the program to your answer. Does the robot do what you predicted? If not, what is different? Check your program and your answer to see if you can spot, and fix, the difference.

Task 3: Change the sequence

Try changing the sequence of your program. Change the order of at least three of the blocks in your program. Don't add any more blocks or change any of the options inside a block – only change the sequence.

Download your new program to Edison and run it in your robot.

2. Which blocks did you move to change the program's sequence? Write down your new program.

U2-1.3 Let's explore driving Edison

One of the groups of blocks in the EdScratch block pallet is the **Drive** category. All the blocks in this category relate to using Edison's motors. One of the things you can use the motors to do is drive the robot like a car.

No driver needed! Just a programmer

How do you 'drive' an Edison robot? By **programming** the robot with **code**!



Jargon buster

Programs are the sets of instructions you create for a computer, or an Edison robot, to follow. The stuff inside a program is sometimes called **code**.

People often use the words **code** and **program** to mean the same thing. For example, you could say 'write a program for Edison' or 'write some code for Edison'. Either way it means 'write commands to instruct the Edison robot what to do using a programming language it understands.'

When you use EdScratch to create a set of instructions for Edison, you can say you are **programming** or that you are **coding** or both. Which makes you a **programmer** and a **coder**!

Let's try programming Edison using drive blocks.

Task 1: Drive a straight track

For this task, you need to get Edison to drive a straight track. Use activity sheet U2-2. You need to write a program so that Edison can drive the track. Start Edison on the outline and have the robot stop after it crosses the 'finish' line.

Go to the EdScratch app on your computer. Look at the blocks in the **Drive** category. Which blocks will you need to write your program? Test your program by downloading it to your Edison robot and running it using the activity sheet. Did it work?



Don't forget

You can change the numbers in a block by clicking on the number and changing it using your keyboard.

You can change a drop-down item in a block by clicking on the down arrow and selecting the option you want.

1. You can code a solution for this task using just one block! Which block will work? Fill out the block below to match what you used in your successful one-block program.

Task 2: Drive a maze

For this task, you need to get Edison to drive through a maze. Look at the maze on activity sheet U2-3. Think about the different actions Edison will need to take to drive the maze. Be sure to consider the sequence of the actions too!

2. What actions do you think Edison will need to take to complete the maze? Write down a plan to get Edison through the maze.

Use this plan to help you write a program in EdScratch for Edison to drive through the maze. You will need to use several different blocks in your program to be able to complete the maze. You will also need to figure out what **input parameters** to use in each block.



Jargon buster

The things you can change inside a block, like the numbers and choices in the drop-down lists, are called **input parameters**.

Start Edison on the outline in the maze and have the robot stop after it crosses the 'finish' line. Be sure to have Edison stay inside the lines all through the maze - no cheating!



Hint!

Your program might not work the first time – and that's okay! Part of coding is experimenting and problem solving. If your program isn't working, think about the actions you need Edison to do one by one to complete the maze.

Are you missing any actions in your program? Are there any actions out of order?

You can also try changing some your input parameters. Sometimes changing an input parameter even just a little bit makes a big difference!

U2-1.3a Challenge up: Maze madness

Did you get Edison to drive the maze on activity sheet U2-3 successfully? Try these other maze challenges! Remember to have Edison stay inside the lines all through the maze - no cheating!

Mirror track

Complete the maze by starting from the 'finish' line and driving to the start.

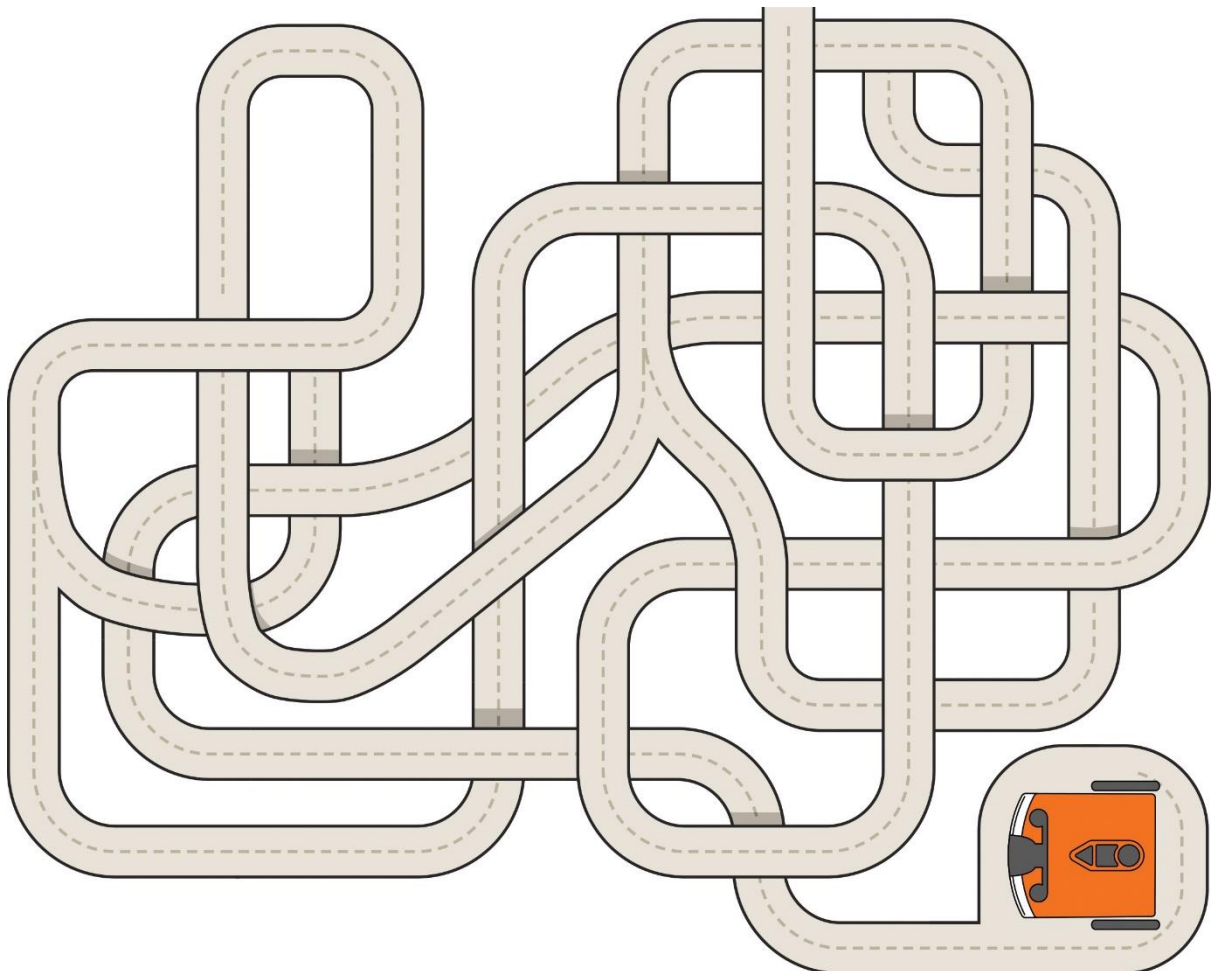
Backwards bot

Complete the maze by driving backwards from the beginning of the maze all the way to the end.

Make your own maze

Create your own maze for Edison to drive through, then write an EdScratch program for Edison to be able to complete your maze.

Once you solve your maze, swap with a partner. While they work out a solution to get Edison through your maze, you need to figure out how to get Edison to drive through their maze!



U2-1.3b Challenge up: Self-walking pet

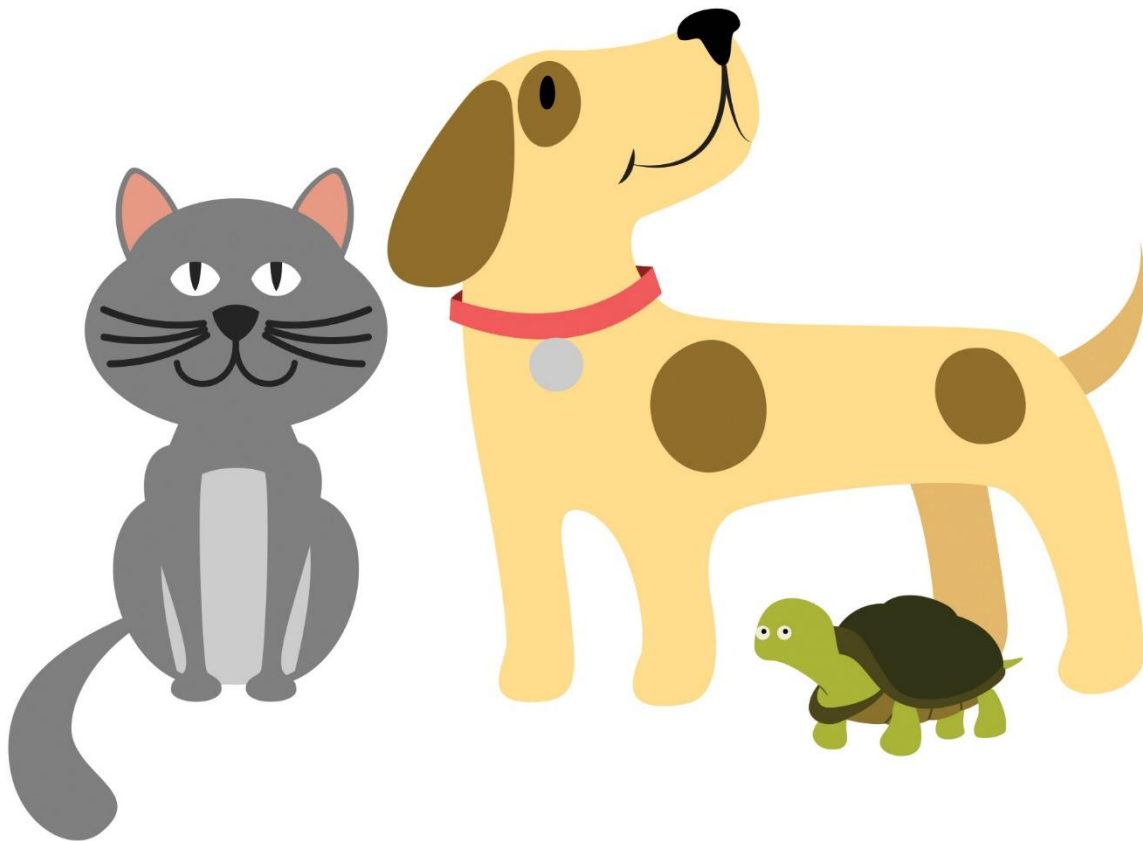
Having a pet means taking care of that pet. You have to feed it, clean up after it, and make sure it gets exercise. If only there were an easier way...

In this activity, try turning Edison into an animal that can take itself for a walk.

How can you make Edison into a self-walking pet? Here are a few ideas:

- You could decorate Edison to turn the robot into a pet.
- You could build a pet that attaches to Edison.
- You could make something that goes on the outside of Edison which Edison can move around.

Create your pet. Then write a program so that your pet will go for a walk and come back to you!



U2-2.1 Let's explore Edison's outputs

What do computers do? Computers **process** information. This means that computers take information from somewhere and do something with that information. For example, you can give a computer two numbers and tell it to add them together. The computer can then add those numbers and show you the result.

This cycle of information coming in, the computer doing something with the information and then creating some result is called the **input-process-output cycle**.



Jargon buster

Inputs are the information and instructions that you give a computer.

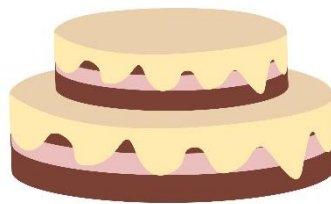
A **process** is what the computer does with a computer program full of information and instructions. This is sometimes called 'running' the program.

Outputs are the results you get from a computer. What the computer displays, or how the robot behaves, are the outputs you get based on the information and instructions you gave the computer.

We call this process of **inputs** going into a computer, that computer **processing** the information, and then generating some type of **output** the **input-process-output cycle**.

The input-process-output cycle isn't only used in computers. You can see this cycle in action in your daily life too.

Baking a cake is a good example. You **input** ingredients into a pan and put that pan into the oven. The **process** of baking then happens in the oven. After a while, the **output** of a cake is ready!



Inputs, outputs and Edison

When you write a program for your Edison robot, you are telling the robot what you want it to do by giving it inputs. Edison's microchip then processes the information to tell the robot what to output.

Your Edison robot has three main types of outputs: outputs using the motors, outputs using the LEDs and outputs using sounds. In EdScratch, the blocks related to Edison's main outputs are organised into three different categories: **Drive**, **LEDs** and **Sound**.

1. Look at the blocks in the **Drive**, **LEDs** and **Sound** categories in EdScratch. Which category contains the code blocks you would need to input into a program to get Edison to generate each output? Match each output to the category where you can find the blocks you would need.

Output	Category
 Turn Edison's lights off ☐	 Drive
 Play a musical note ☐	 LEDs
 Spin Edison right ☐	 Sound

Task 1: Flash and beep

Edison has two red LED lights. When you turn Edison on, you can see these two LEDs begin to flash.

Edison also has a special bit of tech which you can see just to the left side of the round button on the top of the robot. This is a buzzer and a sound sensor all in one. It can detect noise, but it can also make noise!

For this task, you need to write a program which will have Edison use both the LED and buzzer outputs. Write a program in EdScratch **using eight blocks** which tells Edison to do the following things in sequence:

- ☐ turn on the left LED light
- ☐ beep
- ☐ turn off the left LED light
- ☐ beep
- ☐ Turn on the right LED light
- ☐ beep
- ☐ Turn off the right LED light
- ☐ beep

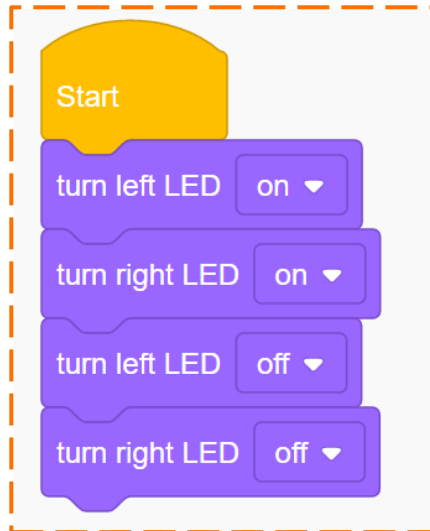
Download and test your program with your Edison robot.

2. Did the program work the way you expected? Could you observe the robot perform each step of the program?

Task 2: Make Edison blink

Some of Edison's outputs, like turning a LED on or off, happen very quickly. In fact, this can happen so fast, it can be really hard to see.

Try writing the following program in EdScratch:



Download it and run it with your robot. Can you see Edison blink?

Because the robot flashes its LEDs when it is in standby mode, it will be really hard to see this program in action.

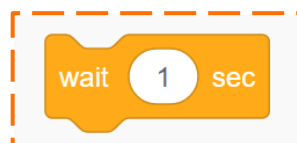


Why is that?

Edison moves through each EdScratch block one at a time, but the robot is able to process each block very quickly. Computers can process information very fast – that's one of the things that makes them so useful!

If you want Edison to pause after it completes one block before going onto the next block, you have to tell it that.

One of the EdScratch block categories is the **Control** category. The blocks in the **Control** category allow you to control the flow of your program. One of the blocks in this category is the **wait** block:



Name _____

This block tells Edison to wait the amount of time you specify before moving on to the next block. Let's try using this block in a program.

Modify the 'blink' program from before but use this new control block to make the program work better. You want a program where it's very easy to see Edison 'blink'. Experiment using at least one **wait** block. Test using **wait** blocks in different places in your program to see what works best.

3. What does your program look like? Which blocks does it use, in which order? Write your program below. Be sure to include the input parameters you used.



Mini challenge!

When a person blinks, what happens? Their eyes start open and then...

Look at your blink program. Can you adjust your program to make it more like a blink?

U2-2.1a Challenge up: Drive the maze safely

You can write programs for Edison which tell your robot to use multiple types of outputs.



Don't forget

Your Edison robot has three main types of outputs: outputs using the motors, outputs using the LEDs and outputs using sounds. In EdScratch, the blocks related to Edison's main outputs are organised into three different categories: **Drive**, **LEDs** and **Sound**.

For this activity, you will need to write a program telling Edison to drive the maze on activity sheet U2-3. This time, however, Edison needs to be a very safe driver. On the road, drivers use their indicator lights and horn to alert other drivers. Edison can do these things too!

Drive the maze starting at the outline and driving forwards to the finish line. Your program should end after Edison crosses the finish line.

Your program needs to tell Edison to pause and use the LED lights to 'indicate' before making each turn in the maze. Make sure other drivers would be able to see the LEDs indicate!



Hint!

If you are going to turn left, which LED should you use to indicate? What about when you turn right?

Your program should also use the 'beep' block at least one time somewhere in the program.



Hint!

Pretend the 'beep' block is like a horn in a car. Where in your program might it make sense for Edison to beep?

U2-2.2 Let's explore input parameters

Every block in EdScratch contains inputs which tell Edison what you want the robot to do. You have probably noticed that lots of the blocks also have **input parameters**.



Jargon buster

Input parameters are the things you can change inside a block, like the numbers and choices in the drop-down lists. You can think of input parameters as specific pieces of information needed in an input.

For example, if you want Edison to drive forward, you need to give the robot specific information about that command, such as how far to drive and at what speed.

There are three styles of input parameters in EdScratch:

- **numbers** you type into the block using your keypad,
- **drop-down menus** where you choose an option, and
- **round or diamond-shaped holes** you fill with special blocks.

Each input parameter in a block gives a different piece of information that Edison will need to be able to run that command. You can think of input parameters as the answers to questions the robot has about what you are asking it to do. For example, if you want Edison to drive backwards, there are three questions you need to answer:

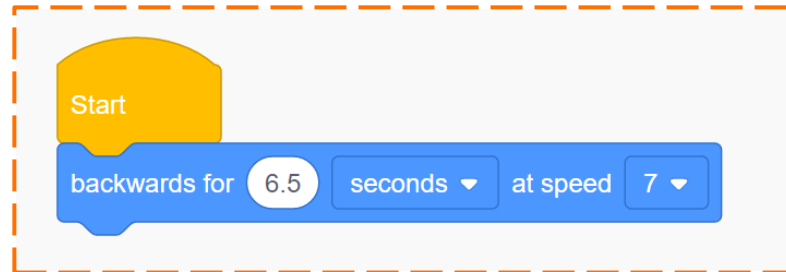
1. **Question:** How far do you want the robot to go?
→ **Answer:** **distance input parameter**
2. **Question:** What units are you using to measure distance?
→ **Answer:** **distance units input parameter**
3. **Question:** How fast do you want the robot to go?
→ **Answer:** **speed input parameter**

You always need to fill in all the input parameters in a block to give the robot all the information it needs.

Try it out!

The input parameters in a program give you lots of information about what that program is asking the Edison robot to do. Try reading the following programs and answer the questions.

Look at this program:

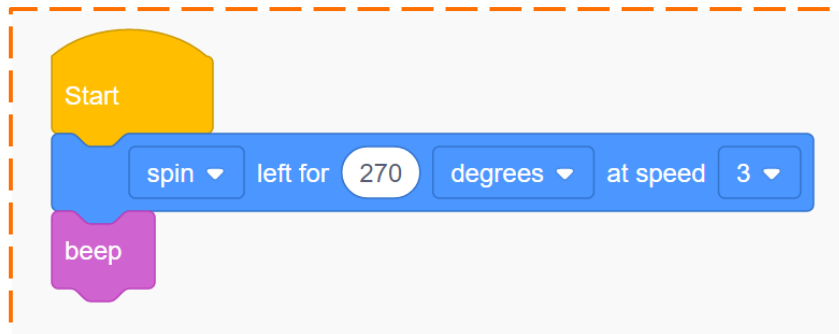


1. In this program, what is the input parameter for **distance units**?

2. In this program, what is the input parameter for **speed**?

3. What is the full program telling Edison to do? Highlight all of the input parameters in your answer in another colour or by underlining or circling each one.

Look at this program:



4. How many input parameters are there in this program?

5. Remember that input parameters are answers to questions the robot needs to know. What question is the **spin** input parameter answering? *Hint:* you might want to look at the other options for this input parameter in EdScratch to help you think about your answer.

U2-2.2a Change it up: Teach Edison to count to 9

Have you ever looked at how old digital signs and clocks display numbers? The numbers are displayed using lines which make up a rectangle-shaped grid. Each number, from 0 up to 9, can be displayed by using some combination of the lines in that grid.

Digital display numbers don't have curves or diagonal lines. They only have straight lines and right angles. That makes them perfect patterns for Edison to drive!

Task 1: Teach Edison a number

Look at activity sheets U2-4, U2-5, U2-6 and U2-7. Choose one of the activity sheets to use.

Write a program for Edison so that the robot can trace over the digital display number you chose. Start the robot off of the digital display number and drive so that the robot traces over every segment of the number.

1. Which digital display number did you use?

2. What does your program look like? Which blocks does it use, in which order? Write your program below. Be sure to include the input parameters you used.

Name _____

3. Look at the distance input parameter in the blocks in your program. What do you notice about the inputs you used? How could this help you plan out a program for a different digital display number?

Task 2: Teach Edison a different number

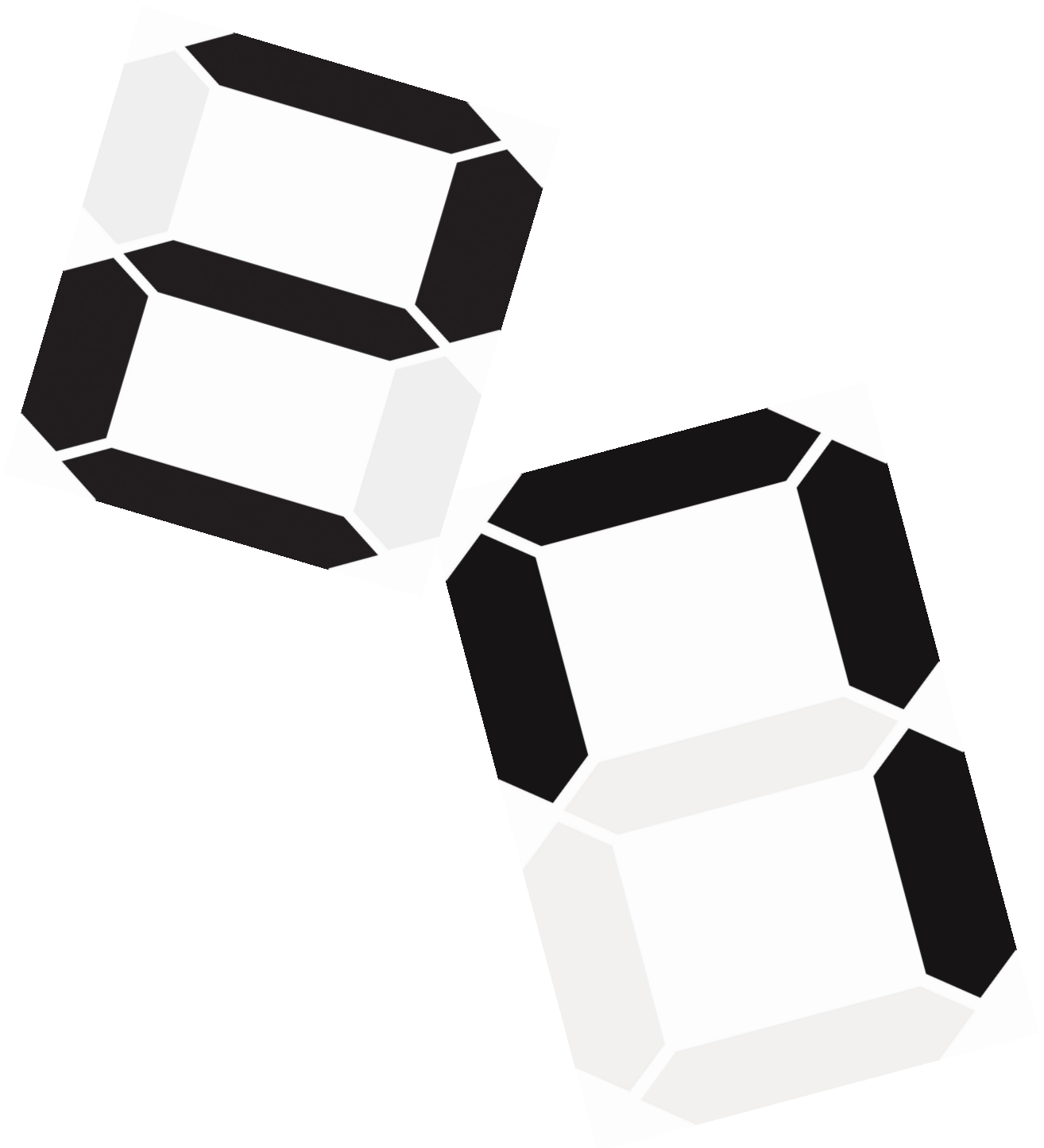
Choose a different digital display number activity sheet. Use what you learned about the distance input parameter from your last number and write a program for Edison so that the robot can trace over your new digital display number.

4. Which digital display number did you use?

5. What does your program look like? Which blocks does it use, in which order? Write your program below. Be sure to include the input parameters you used.

Name _____

6. Compare your two programs. Are there any patterns you notice that are similar in both?
What are they?



U2-2.2b Challenge up: Teach Edison to count to 9 out loud

Can you get Edison to trace a digital display number by driving over it and count the same value 'out loud' in just one program?

What to do

Look at activity sheets U2-4, U2-5, U2-6 and U2-7. Choose one of the activity sheets to use.

Write a program for Edison so that the robot traces over the digital display number you chose. You also need Edison to count 'out loud' somehow.

Your program needs to have Edison count the same amount as the digital display number it is driving. For example, if you choose the number 5, your program needs to have Edison give some sort of signal as it 'counts' to 5.

Think about the sequence of things you want Edison to do. Will the robot drive the whole path and then count? Count before driving? Drive a little, count to one, then drive a bit more before counting the next number?

How you do it is up to you!



Hint!

Which of Edison's outputs could you use to signal the robot is counting?

Mini challenge!

What about the rest of the numbers? Make your own digital display number with a different number than the activity sheets.



Don't forget

Digital display numbers don't have curves or diagonal lines. They only have straight lines and right angles. The numbers are displayed using segments which make up a rectangular grid. Each number, from 0 up to 9, can be displayed by using some combination of the lines in the grid.

Once you have made your digital number, test it out! Write a program for Edison to trace and 'count' your number.

U2-2.3 Let's explore Edison's musical talents

Outputs using sounds are one of your Edison robot's three main types of outputs. In EdScratch, the blocks related to sound outputs are in the **Sound** category.



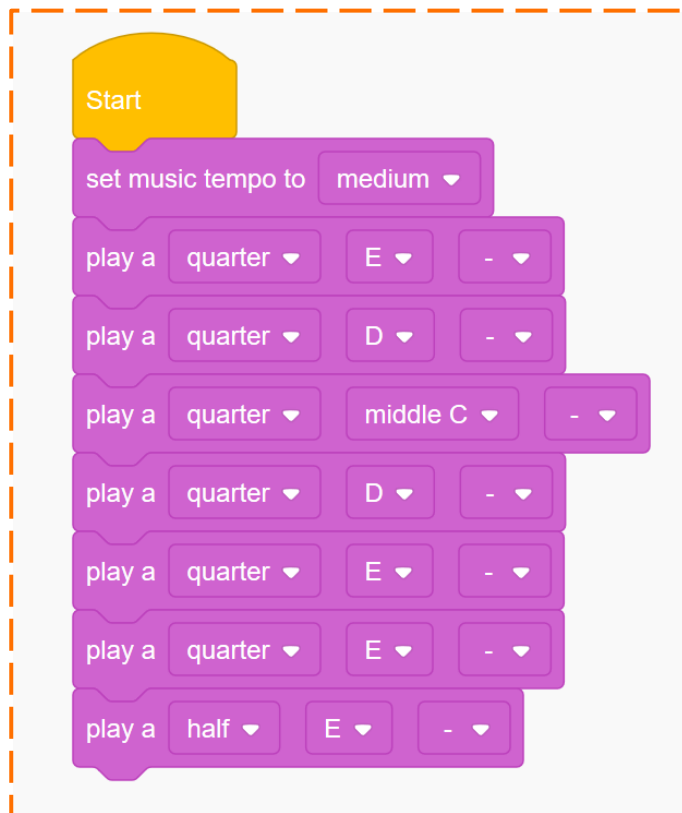
Don't forget

Edison has a special bit of tech which you can see just to the left side of the round button. This is a buzzer and a sound sensor all in one. It can detect noise but it can also make noise, like beeps or musical notes.

Have a look at the **Sound** category in EdScratch. There are not many blocks in this category, so you might think there is not a lot you can do with Edison and sounds. By using the sound blocks in different sequential orders and with different input parameters, however, you can play whole songs!

Task 1: Play a tune

In EdScratch, write the following program:



Download the program to your Edison robot and run it. This program is the first part of a song you might know. Do you recognise the tune?

The first block in the program is the **set music tempo** block. Try clicking on the drop-down menu in the **set music tempo** block to see the input parameter options for the block. Choose a different input parameter for the block, then download the adjusted program to Edison. Run your new program in your robot.

1. Which new input parameter did you select for the **set music tempo** block?

2. What happened when you played the new program? What changed compared to the original program?

3. What does the **set music tempo** block do?

4. If you put the **set music tempo** block at the end of this program instead of the beginning, what would happen? *Hint:* Remember that Edison will read each EdScratch block one at a time from the top of the program in sequence.



Hint!

Not sure what would happen if you make a change to a program? One of the best ways to find out is to make the change, download the adjusted program and test it out using Edison!

You can always experiment in coding!



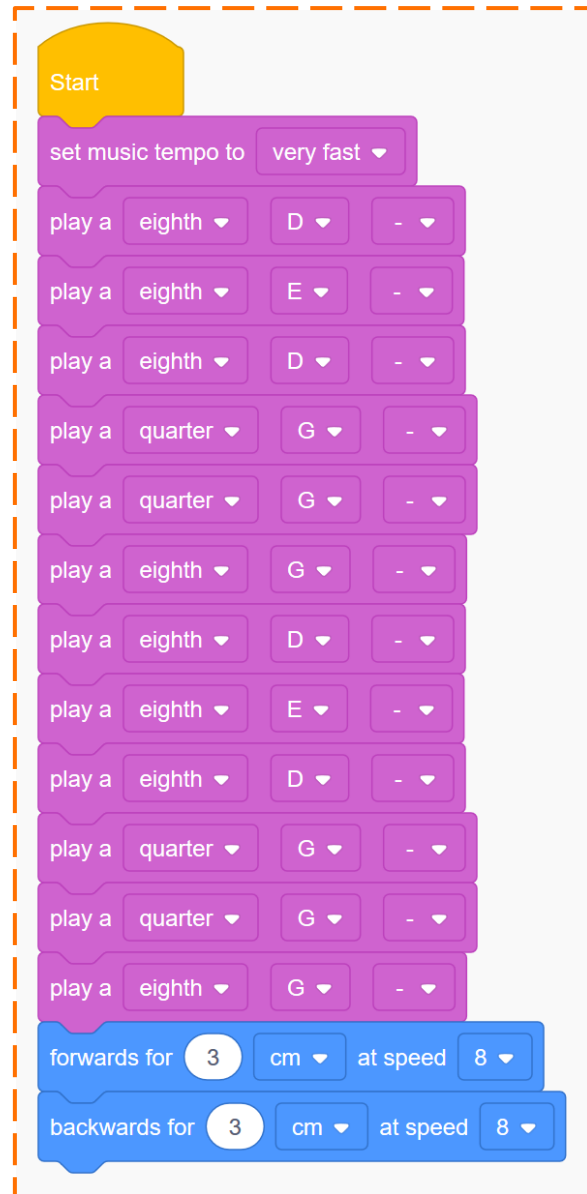
Use this link

Want to hear Edison play more of this song?

Open the program using this link: <https://www.edscratchapp.com?share=Eb12x3Dm>

Task 2: Move to the music

Look at this EdScratch program:



This program has the music for the first part of the song *The Hokey-Pokey*. This is a great song to dance along to – the song tells you just what moves to do!

In EdScratch, write the Hokey-Pokey dance program. Download the program to your Edison robot and play it.

5. Did your Edison robot move along with the music? Why do you think this is the case?

Edison can play a musical note while doing something else, like driving, but you have to tell the robot that is what you want it to do. There is a special block in the **Sound** category in EdScratch you need to use to do this. Here is what it looks like:



The **play music in background** block is a grouping block. Other blocks can sit inside this grouping block.



Why is that?

The shape of a block in EdScratch can give you a hint about what that block is used for in programs. Look at the **play music in background** block. See how it has a shape a bit like a mouth? Other blocks can sit inside the opening of this block's 'mouth'.

Any block that sits inside the **play music in background** block will be affected by this grouping block. Remember, Edison will follow each EdScratch block one at a time. The robot will see the grouping block first and know that any blocks inside that block get the 'play music in background' affect.

Add a **play music in background** block to your Hokey-Pokey dance program. Think about where in the program this block needs to go.

Download the adjusted program to your Edison robot and play it. Edison should now start playing the song and move at the same time!

Edison's dance moves need a bit of work, however.



Hint!

Remember that if something isn't quite right in a program, a warning message will show up in the bug box. These messages can help you work out what should be inside the grouping block and what should not!

Task 3: Dance along

Add more dance moves to your Hokey-Pokey dance program. You could get Edison to follow the steps in the Hokey-Pokey song lyrics or make up your own dance!

Can you get Edison to dance in time to the music?

U2-2.3a Change it up: Play a song in a round

For this activity, you need to use multiple Edison robots together to play a song in a round.



Why is that?

A round is a musical piece where two or more people (or robots!) sing or play the exact same melody, but each begins at a different time. While every participant is singing or playing a different part of the song, the melody still harmonises them together!

Work in a group to turn your Edison robots into harmonising musical stars!

What to do

The first thing you need to do is choose a song for the robots to play. Many nursery rhyme songs work well in a round. One example is the song *Row, Row, Row Your Boat*.

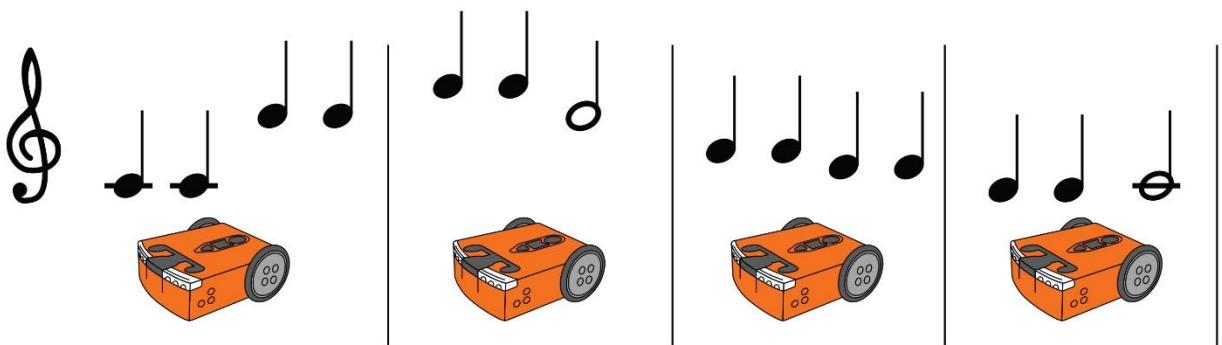
Once you choose your song, you will need to program each robot to play it. Each robot's program will be a little bit different as you need each robot to wait until the right time to join in the round.

Test out your programs with your robots. Then put on a musical show!



Hint!

Which **Control** block tells Edison to **wait**?



U2-2.3b Challenge up: You are the conductor

What is your favourite song? Is there a theme song to a TV show or a movie you love to hear?

For this activity, the song choice is up to you!

What to do

You are the conductor, and you need to get Edison to play the song of your choice. Write a program for Edison so that the robot plays your song. Test out different tempos to see which works best.

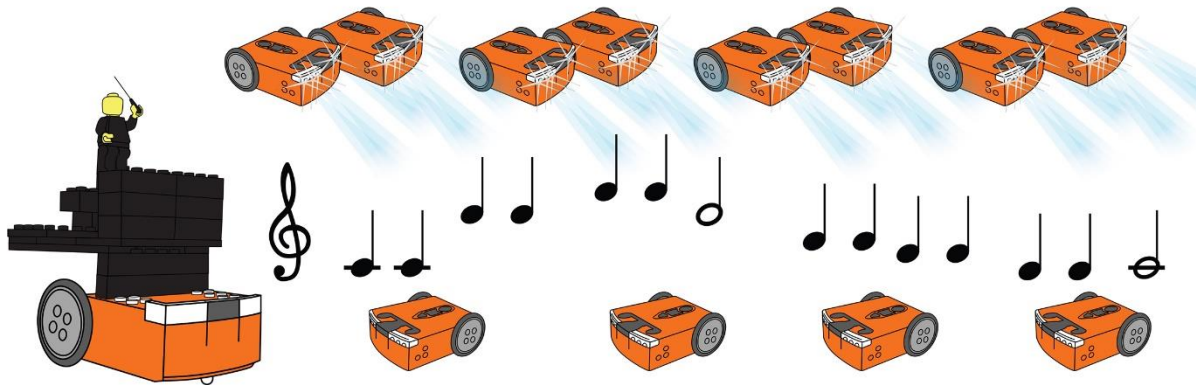
If you want, you can also program the robot to move or dance to your song.

Once you have your program ready, run it in Edison so that the robot performs your musical masterpiece!



Hint!

Which block tells Edison to
play music in the background?



U2-2.4 Let's explore bugs and debugging

Do you want to know a secret about working with computers and writing code? Here it is:

SOMETHING ALWAYS GOES WRONG!

Okay, so that isn't actually a secret, but it is really important. A major part of computational thinking and working with computers is problem-solving.



Don't forget

Computational thinking means thinking about a problem or task in a way that is similar to how a computer thinks. It is a way of planning, problem-solving and analysing information the same way a computer does.

When things don't work the way you want, remember that is okay! It just means it is time to problem-solve. One of the main types of problem-solving you do in computer science is finding and fixing **bugs**.

Debugging is a major part of coding and using robots. People who work as computer scientists, computer programmers or roboticists professionally spend a lot of time debugging. Probably even more time than they do writing their code!



Jargon buster

When something isn't working in a computer program, it is called a **bug**.

Finding and fixing bugs in a computer program is called **debugging**.



Why is that?

Calling something that's going wrong a 'bug' might seem a bit strange, but that's really what these errors are called.

Why are computer problems called **bugs**? A woman named Grace Murray Hopper, who is one of the inventors of modern computer programming, came up with this term. She once discovered that the issue causing her computer to malfunction was an actual moth that had gotten into the hardware! She fixed the problem by literally **debugging** her computer.

The name stuck and now problems with software or hardware in computers are called **bugs** and fixing them is called **debugging**!

Debugging in EdScratch

The bug box in the EdScratch programming environment is a special feature to help you find and fix any bugs in your code.



Don't forget

The **bug box** is the area below the block pallet and programming area in EdScratch. If there is a bug or if it seems like something isn't quite right in a program, a warning message will show up in the bug box.

The warning messages that appear in the bug box are there to give you information about any problems or potential problems in your code. These messages are EdScratch's way of saying that there is an error in your code, or, that you might find an error when you run your program in Edison. In coding, there are two main types of errors: **syntax errors** and **logical errors**.



Jargon buster

Syntax is the rules of how a programming language works. All languages have rules. For languages people speak, like English or Chinese, there are rules about spelling and grammar and how to write the letters or characters in that language. Syntax is the same thing but for a computer language.

Syntax errors are caused by problems in how you wrote your code which break the rules of the language. These errors are sort of like typos or spelling mistakes in writing.

Logic is an organised way of thinking that makes sense to a computer. Logic determines the flow of a program, how you order things inside a program and what inputs you use to generate the outputs you want.

Logical errors are problems with the logic of a program. Logical errors are basically problems with the way of thinking in a program. Programs that do not make sense to the computer and programs that ask a computer to do something that it cannot do are examples of logical errors. If a program works differently than you expected it to work, there is a good chance that there's a logical error somewhere.

Understanding if a bug is the result of a syntax error or a logical error can help you fix the problem. If there is a syntax error in your EdScratch program, you will get a red warning message in the bug box, and you won't be able to download the program to Edison. If you can download your program and run it in Edison, but it doesn't do what you want, that probably means there is a logical error.

Try it out!

Look at the following program:



This program is full of bugs. Your job is to fix it!

**Hint!**

One of the best ways to see what's going wrong – and right – with a program is to test it out using EdScratch and Edison!

Name _____

1. The programmer who wrote this program made two errors with the **set music tempo** block. One error is a **syntax error**, and one error is a **logical error**. Write an explanation of each of these two errors to explain them to the programmer. *Hint:* the programmer wants the music to play at a very fast tempo.

Syntax error: _____

Logical error: _____

This is what the programmer who wrote this code said about what the program is meant to do:

"I want the Edison robot to play a tune very fast. While the music is playing, I want the robot to spin left for one full circle (360 degrees), then spin back right the same distance at the same speed."

2. The program does not work the way the programmer wants. You need to debug the program and get it working for the programmer. Test your program using Edison and keep debugging until the program works just the way the programmer explained. Describe the bugs you found and what you did to fix them.

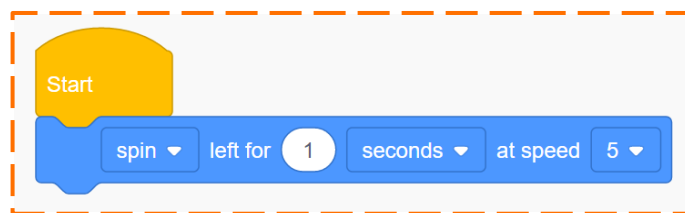
U2-2.5 Let's explore Edison's motors

Edison robots have two motors: one on the left side and one on the right side. Outputs using these motors are one of your Edison robot's three main types of outputs. In EdScratch, the blocks related to motor outputs are in the **Drive** category.

When you write a program for Edison using blocks from the **Drive** category in EdScratch, you are telling the motors what to do. Most of the blocks control both of Edison's motors. Does that mean that both the motors do the same thing?

Task 1: Spin that robot

If you wanted to write code to tell your robot to 'spin left' you can make a simple one-block program like this:



The input parameters in that block tell Edison the direction, the distance, the distance units and the speed you want the robot to use in the program.

The direction input parameter that has been selected is **spin**, which means the whole direction input is **spin left**. What is that input telling Edison's motors to output?

In EdScratch, write the program using the same input parameters as the one in the picture. Download the program and run it with Edison on the desk or floor.

Now run the program again, but this time hold Edison in your hands. Feel how the wheels are moving. What do you notice?

1. Which direction is the left wheel moving?

2. Which direction in the right wheel moving?

Edison's motors don't have to both do the same thing at the same time. Does that mean you could write a program moving just one of the motors? Can you write a program that tells each motor what to do separately?

Open the EdScratch app and look at the **Drive** category in the block pallet. Look at the different blocks and see if you can discover blocks you could use if you only wanted an output from one of Edison's motors.

3. Which blocks do you think only use one of Edison's motors? Why do you think that?

4. You can use Edison to build and invent lots of different things. Imagine you need to create something using Edison which only uses one of Edison's motors. What could you build? How would your creation use the one motor?



Don't forget

The wheels of your Edison robot can be removed from the powered sockets they sit in. These sockets are what Edison's motors actually move.

Task 2: Direction = forward

To get Edison to work the way you want, you have to make sure you give the robot all the information it needs. If the robot doesn't have all the inputs and instructions it needs, the program probably won't work the way you want. This kind of logical error can be frustrating, especially if you think you have given the robot all the information necessary.

One of the main ways you give information to the robot using EdScratch is through input parameters.



Don't forget

Each input parameter in a block gives a different piece of information to Edison that the robot will need to be able to run that command. Input parameters are sort of like the answers to questions the robot has about what you are asking it to do.

In EdScratch, some blocks get all the information they need from their own input parameters. Other blocks get information from inside their block, but also need information from somewhere else in the program as well.

Let's make a program for Edison to get the robot to move its motors. For this program to work, there are four questions you need to make sure your program answers:

Question 1: What direction do you want the robot to go?

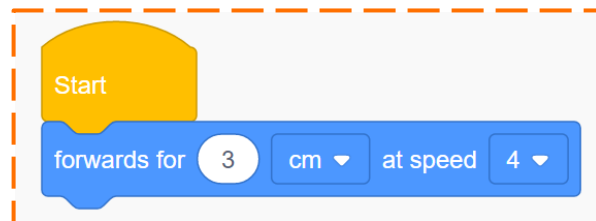
Question 2: How far do you want the robot to go?

Question 3: What units are you using to measure distance?

Question 4: How fast do you want the robot to go?

Your program needs to give the robot an answer to all of those questions.

Look at this program:



If you ran this program in an Edison robot, would the robot have all the information it needed to know what to do? In other words, does this program tell the robot the direction, distance, distance units and speed to move the motors?

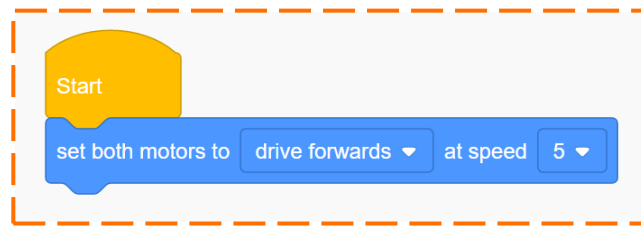
5. Fill in this chart. If the information is in the program, write the value of that input in the 'value' column. For example, the value of 'distance' is the answer to the question, 'how far is this program telling the robot to go?'

Information	In the program?	Value
Direction		
Distance		
Distance units		
Speed		

Write the program in EdScratch, download it and run it in your robot.

6. What did the robot do when you ran the program?

Now look at this next program:



Does this program give the robot all the information it needs?

7. Fill in this chart. If the information is in the program, write the value of that input.

Information	In the program?	Value
Direction		
Distance		
Distance units		
Speed		

Write the program in EdScratch, download it and run it in your robot.

8. What did the robot do when you ran the program? Why did it behave that way?

How can you give the robot the rest of the information it needs so that the robot moves its motors? Experiment in EdScratch to see if you can write a program that uses the **set both motors** block but no other blocks from the **Drive** category and gets the robot to move forward.



Hint!

Don't give up! Experimenting, testing and problem solving is how you learn new things in coding.

Feeling stuck? Think about what it is you are trying to do a different way. Look at different blocks in EdScratch and ask yourself, 'if I use this block, will it fix the problem I am trying to solve?'

If you aren't sure, try it and see what happens! Be sure to check the bug box for hints too!

Name _____

9. What did you try? Did it work? Write what you did, what worked, what didn't work and how experimenting made you feel.

This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

U2-2.5a Challenge up: Spinning garden

Edison's two motors each control a powered socket, one on the right side of the robot and one on the left. When you want the robot to drive, you attach wheels to the powered sockets using the wheels' axles. The motors turn the axles, which turns the wheels and allows Edison to drive.

How else can the powered sockets be used?

If you turn an Edison robot on its side, you won't be able to drive it like a car. Instead, you can use the robot to be the powered base for an invention!

What could you attach into the powered socket instead of a wheel? What will happen when the motor is turned on?



Don't forget

The wheels of your Edison robot can be removed from the powered sockets they sit in. These sockets are what Edison's motors actually move.

What to do

In this project, you need to use your Edison robots to help create a spinning garden. Work in a group to design a garden that uses Edison robots as the bases for plants, flowers, bees, birds or whatever else you would like to have in your spinning garden.



You can take the wheels off of the robots and use a different axle inside the powered socket or build using a wheel as a base. Each robot needs to have something created and attached to it which can spin in the garden.

Each robot will need to be programmed using EdScratch. Write and test your programs for each robot. You may need to make adjustments to your program depending on the type of object you are using and how you attach that object to the powered sockets of each robot.



Hint!

The **set right motor** and **set left motor** blocks are very helpful if you only want one motor to move. Don't forget you need another block, like a **wait** block, in the program to set the duration for the **set motor** blocks.

U2-2.5b Challenge up: Spinning solar system

Edison's two motors each control a powered socket, one on the right side of the robot and one on the left. When you want the robot to drive, you attach wheels to the powered sockets using the wheels' axles. The motors turn the axles, which turns the wheels and allows Edison to drive.

How else can the powered sockets be used?

If you turn an Edison robot on its side, you won't be able to drive it like a car. Instead, you can use the robot to be the powered base for an invention!

What could you attach into the powered socket instead of a wheel? What will happen when the motor is turned on?



Don't forget

The wheels of your Edison robot can be removed from the powered sockets they sit in. These sockets are what Edison's motors actually move.

What to do

In this project, you need to use your Edison robots to help create a model of the solar system where the planets can spin. Work in a group to build your model. Decide how you will build the planets, if you will include moons, how big each solar object will be and how fast each one will spin. How accurate to the real solar system can you make your model?

You can take the wheels off of the robots and use a different axle inside the powered socket or build using a wheel as a base.

Each robot will need to be programmed using EdScratch. Write and test your programs for each robot. You may need to make adjustments to your program depending on the size of each object and how you attach that object to the powered sockets of each robot.



Hint!

The **set right motor** and **set left motor** blocks are very helpful if you only want one motor to move. Don't forget you need another block, like a **wait** block, in the program to set the duration for the **set motor** blocks.



U2-2.5c Challenge up: Cartographer and navigator

A cartographer is a person that makes maps. A navigator is a person who figures out how to get from place to place. In this project, you need to be both!

What to do

The first thing to do in this project is to make a big map. You will use this map with your Edison robot, so it needs to be big enough to allow Edison to drive around on the map.

Decide what place your map will be about. Your map could be of your school, your town, a fictional city or a real place in the world where you want to travel. Whatever you choose, you will need to plan out your map and then make a version big enough for Edison robots to drive on.

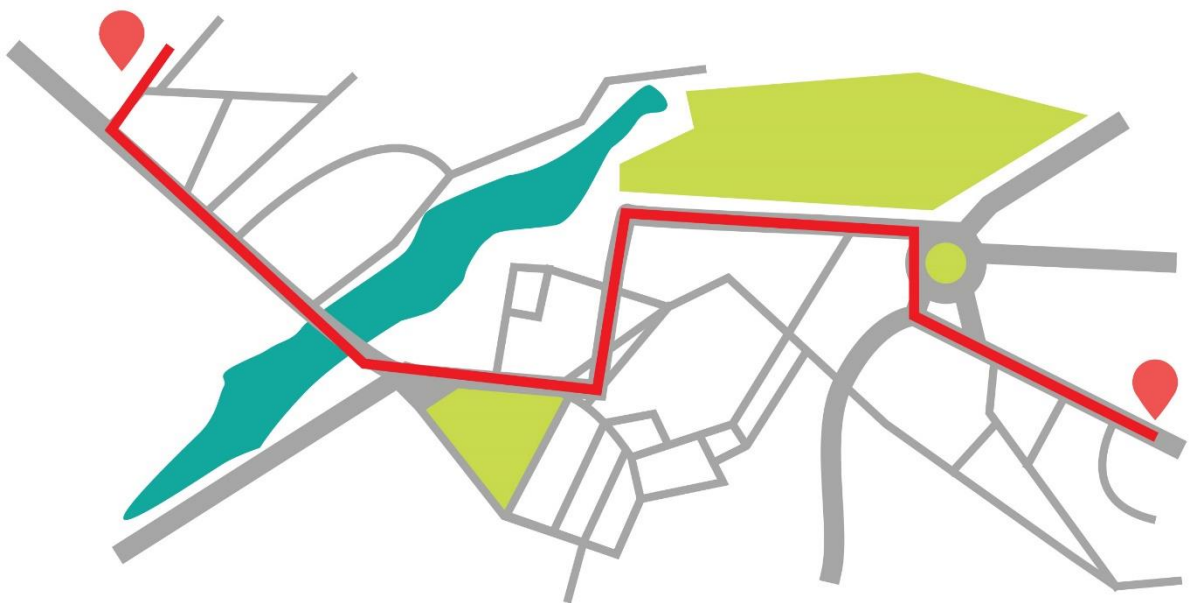
You also need to create programming challenges to be solved using your map. These challenges should tell the programmer where to start the Edison robot, where the programmer needs to have the robot finish, and any rules for the program. For example, you could have a challenge that says: *Start at the school. End at the ice cream shop. Don't go past the park.*



Hint!

Your program rules don't have to be just about where to go and where to avoid. You can also make rules about how Edison travels, such as going backwards or the speed Edison moves. Program rules that require the programmer to use blocks from the **LEDs** and **Sound** categories are good too!

Test the programming challenges to make sure a solution is possible for each one. Then trade challenges with a partner or another team. How many challenges can you solve?



U2-2.5d Challenge up: Writer and director

Edison cannot speak, but that doesn't mean you cannot use the robot to help tell a story! In this project, you need to write a story, then 'direct' Edison to help tell the tale as if the robot were an actor in a play.

What to do

Write a story using a story map. Have Edison help you present this story. You will 'direct' Edison by programming the robot.

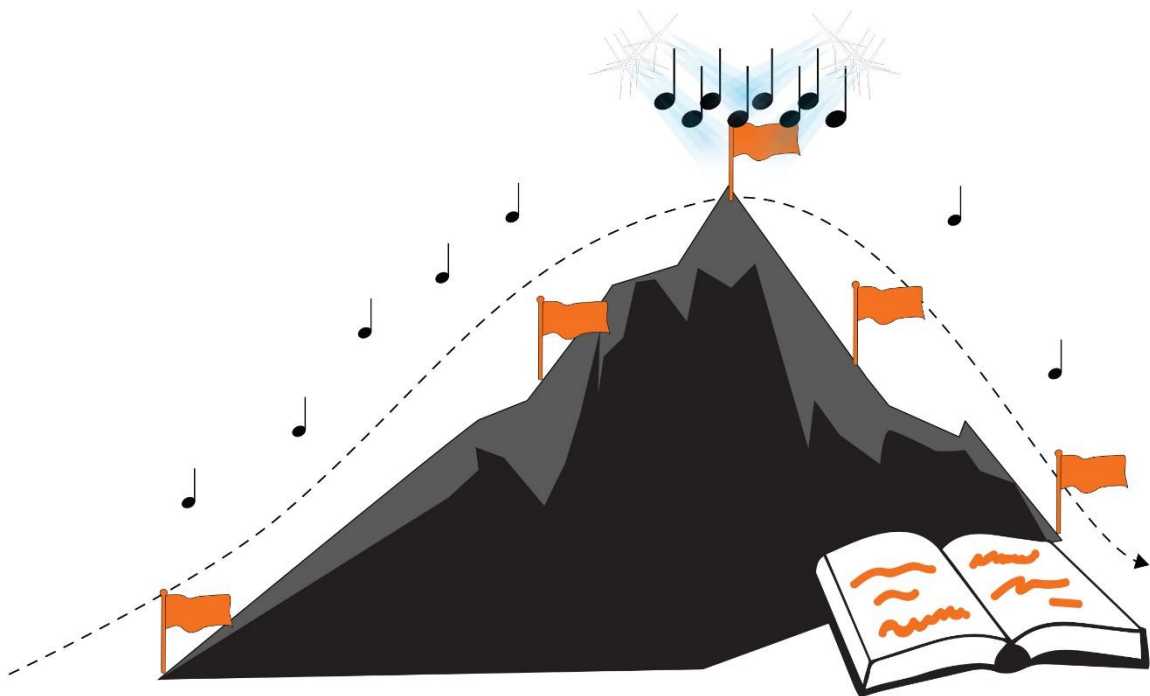


Hint!

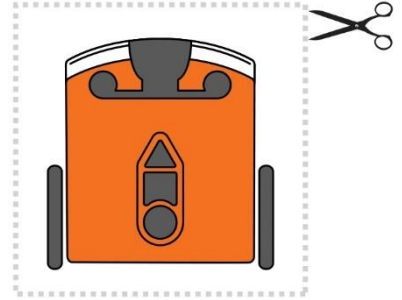
There are lots of ways you can do this project. You could make a story map that is big enough for Edison to drive on, performing actions at each stop on the story map. Or you could have Edison perform actions in time with you as you read the story out loud. Or maybe you can even turn this into a movie by filming Edison performing!

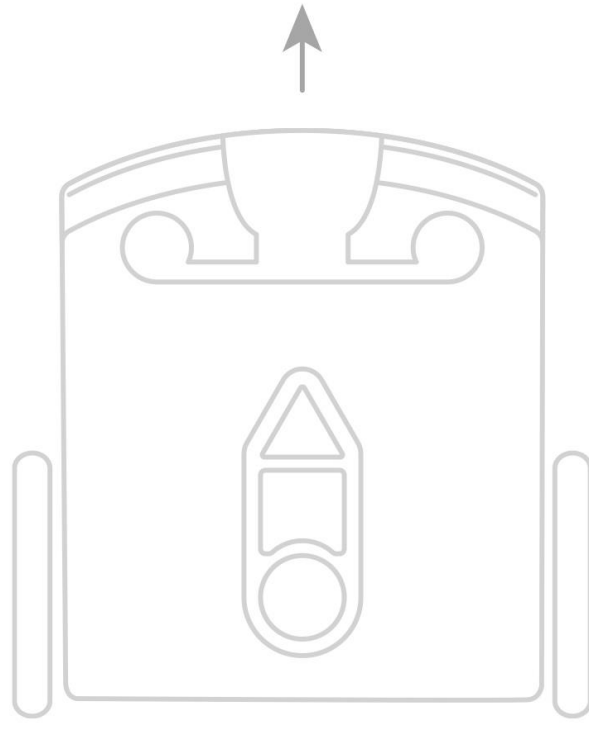
You can use any of Edison's outputs to help tell the story. Choose blocks from the **Drive**, **LEDs** and **Sound** categories. Don't forget about **wait** blocks! You can use **wait** blocks to help control the timing of Edison's actions!

Create a program for the robot to follow along and help make the story exciting by doing something at each major point in the story.

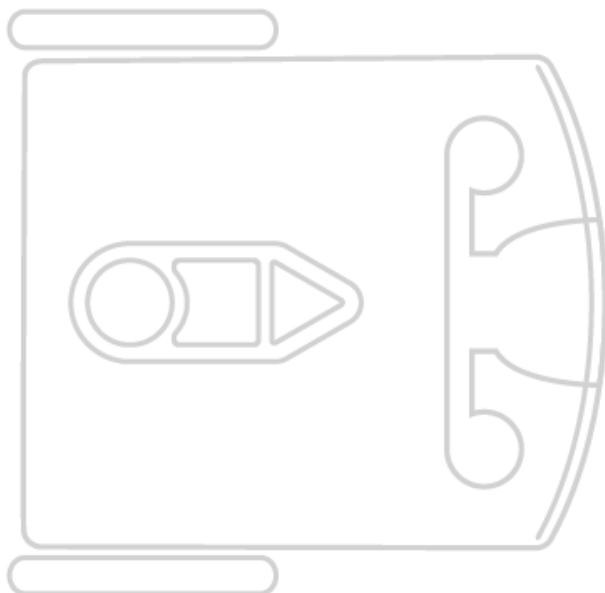
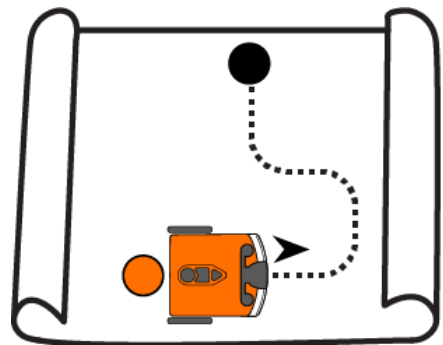


Activity sheet U2-1: Go step-by-step





Activity sheet U2-3: Mini maze



Activity sheet U2-4: Digital display 2



Activity sheet U2-5: Digital display 5





