Edison and EdPy: Get Edison moving

How do you ‘drive’ an Edison robot? By programming the robot with code! Let’s write code in EdPy to get Edison moving.

To do this, there are a few things we need to learn:

- Part 1: How do computers ‘think’?
- Part 2: How do you navigate a maze?

Is this your first time using Edison robots or EdPy? Start with the activity *Edison and EdPy: Get started* first! Ask your teacher for a copy.

Part 1: How do computers ‘think’?

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. 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 writing code is using **sequence**.

Did you know….

One of the most important things about giving instructions to Edison is the order in which you give the instructions, or the sequence. Sequence means going in order, step-by-step. Sequence is really important to computers. Why?

Imagine you want to bake a cake. You need to mix the ingredients, bake it and then decorate it. What if you mixed up the order of those instructions? Can you frost a cake before you bake it? Nope!

The order you follow instructions in makes a big difference. That’s why sequence is important!

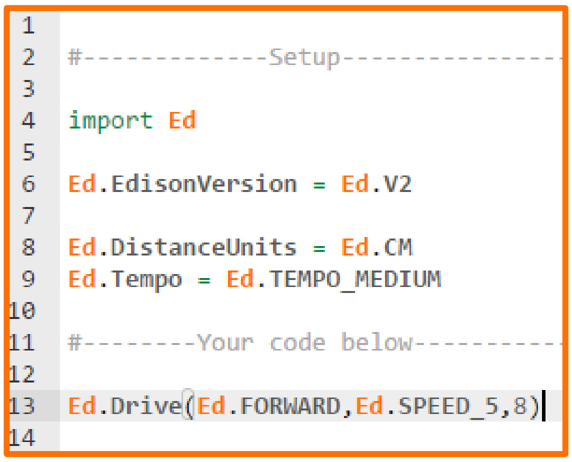
To make a good computer program for our Edison robot, or any computer, we need to write that program in a way that the computer can understand. That means you need to tell Edison exactly what to do, in the exact order you want the robot to do each step.

When you write a program for your Edison robot in EdPy, you are writing the instructions that will tell the robot what to do and in what order to do each thing. Each line of code in your program is one command you are telling the robot to take.

Edison looks at each line of code in a program, starting with line 1. The robot runs whatever command is in that line. Then it moves on to the next line.

Let’s try writing a program for Edison in EdPy to see how sequence works.

Look at this program:



```
1
2   #-------------Setup---------------
3
4   import Ed
5
6   Ed.EdisonVersion = Ed.V2
7
8   Ed.DistanceUnits = Ed.CM
9   Ed.Tempo = Ed.TEMPO_MEDIUM
10
11  #--------Your code below----------
12
13  Ed.Drive(Ed.FORWARD,Ed.SPEED_5,8)
14
```

When you run this program in Edison, what will the robot do? Try writing the program in EdPy below the setup code.

Did you know….

All EdPy programs use the Setup code. This is a special set of commands that helps get the robot ‘set-up’ to run the program.

You don’t need to change the Setup code – just write your code below it!

Step 1: Start writing your drive command by typing ‘Ed’ into line 13.

As you begin typing ‘Ed’ on line 13, you will see a prompt box pop up. The prompt box shows a list of possible commands for you to select. This is a feature of the EdPy app called command line completion that makes it quicker for you to program.

Step 2: Type ‘Ed.Drive(’ into line 13, and select the ‘Ed.Drive()’ function.

Did you know….

A function is a piece of code that performs a particular role or job, depending on which parameters are input.

Ed.Drive() is a function that’s being imported from the Edison ‘Ed’ library module by the Setup code. All the functions that are imported from the ‘Ed’ library must start with ‘Ed.’ This tells the program which library to go to in order to find that function.

Step 3: Fill in the input parameters.

Did you know….

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

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.

When a function has input parameters, you need to enter a value for each one.

The Drive() function has three input parameters:

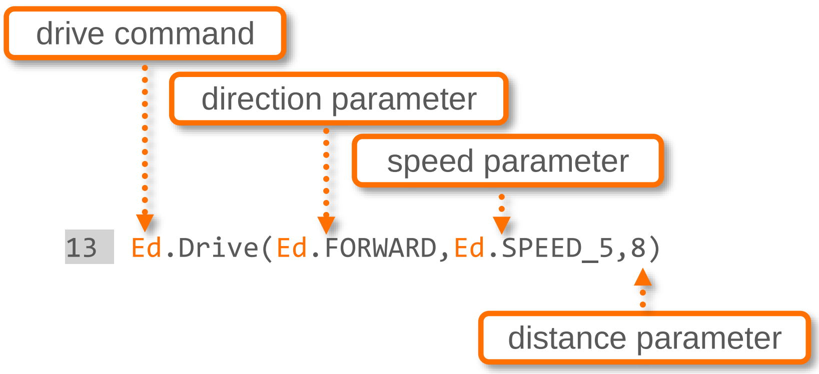
- **direction** – the direction that Edison will drive
- **speed** – the speed at which Edison will drive
- **distance** – the number of distance units Edison will travel

Different input parameters take different values. For example, 'speed' takes Ed.SPEED_ and a number from 1 to 10 (10 is the maximum).

The type of distance units is controlled by the constant that 'Ed.DistanceUnits' is set to in the Setup code. There are three distance units that you can use:

- Centimetres, written as Ed.CM
- Inches, written as Ed.INCH
- Time, written as Ed.TIME

Take a closer look at the drive function and input parameters in this program:



```
Ed.Drive(Ed.FORWARD,Ed.SPEED_5,8)
```

Step 4: Check your program.

Once you have written the program, click the 'Check Code' button and look at the 'Compiler Output' window to make sure you haven't made any errors while typing.

Did you know….

Typo-style errors are called syntax errors. If you type a word that isn't a part of the EdPy program's language, also called syntax, then the EdPy compiler can't understand what it is meant to do. This creates a syntax error.

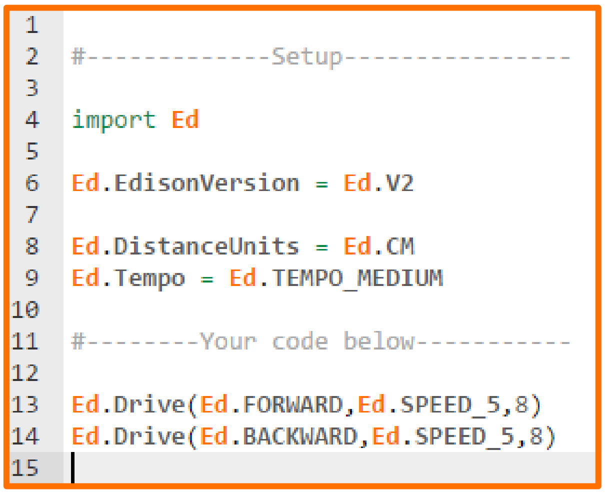
If there are any errors in your code, fix them, so your code matches the example.

Step 5: Download and test your program.

Once you check that your code has no errors, download your program and run it in your Edison. Watch the robot to see the outputs you get from your program.

Step 6: Add a new step.

Next, try adding another command as a new line of code. Modify your program by adding a new line of code under the first command:



```

#------------Setup---------------

import Ed

Ed.EdisonVersion = Ed.V2

Ed.DistanceUnits = Ed.CM
Ed.Tempo = Ed.TEMPO_MEDIUM

#--------Your code below----------

Ed.Drive(Ed.FORWARD,Ed.SPEED_5,8)
Ed.Drive(Ed.BACKWARD,Ed.SPEED_5,8)

```

What will the new line of code tell Edison to do? In what order will Edison do each action in the program?

Check your code, then download the program to Edison. Run the program to see Edison perform the commands in sequence.

Part 2: How do you navigate a maze?

Can you get your Edison robot to successfully navigate through a maze?

Write a new program so that your Edison robot will drive through the mini maze on the activity sheet on the next page when you hit the play (triangle) button.

To successfully complete the maze, you must:

- have Edison start from behind the 'start' line,
- have Edison stop after crossing the 'finish' line, and
- keep Edison inside the border lines of the maze.

You will need to write a program which uses multiple commands using the 'Ed.Drive()' function in sequence to allow Edison to make it through the maze's turns.

Hint!

What actions will Edison need to take, in what order? Plan out the sequence of moves that Edison needs to do in order to get the robot through the maze.

Don't forget! You can look at the 'Line Help' window to see what a line of code you write means to help you make sure your code is communicating what you want correctly. Plus, you can look up functions and input parameters in the 'Documentation' window in EdPy to learn more about what each one does.

You might find these input parameters helpful:

Ed.SPIN_RIGHT	Ed.FORWARD	Ed.SPIN_LEFT

Did you know….

Experimenting is a major part of coding!

Try writing a program, download it into Edison and see what happens. If it doesn't work quite the way you expected, that's okay! See if you can figure out what's causing the issue, adjust that spot in your code, and try again.

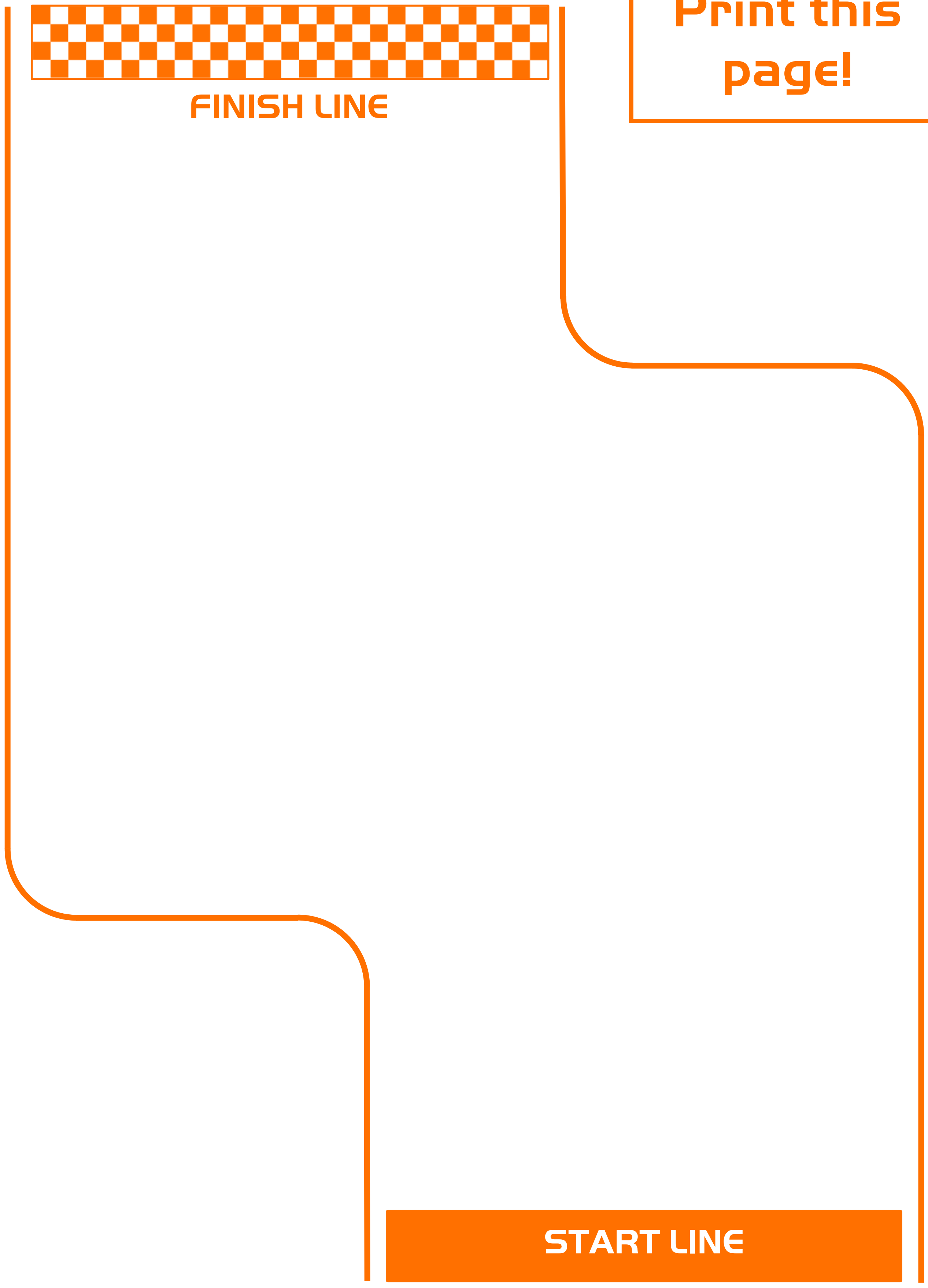
Bonus challenge!

Did you get Edison to drive the maze 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 EdPy program for Edison to be able to complete your maze.

The maze activity sheet

Print this page!


FINISH LINE
START LINE