EdPy worksheets

Student worksheets and activity sheets

Lesson 2: Worksheet 2.1 – Drive the robot forward

In this activity, you need to write a program to drive your Edison robot forward.

Look at the following 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
```

Step 1: Start writing the above program into the EdPy app 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', and it makes it quicker for you to program.

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

Ed.Drive() is a function in Python that's being imported from the Edison 'Ed' library module by the Setup code.

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

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_ 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

    Note: If you have an Edison V1 robot, you must use time, so make sure that Ed.DistanceUnits = Ed.TIME. The time distance unit is milliseconds, which means that to drive for 2 seconds, you need to set the distance input parameter to 2000 (since 2000 milliseconds = 2 seconds).

Step 4: Check your program.

Let's take a closer look at the drive function and parameters in this program.


```
13   Ed.Drive(Ed.FORWARD,Ed.SPEED_5,8)
```
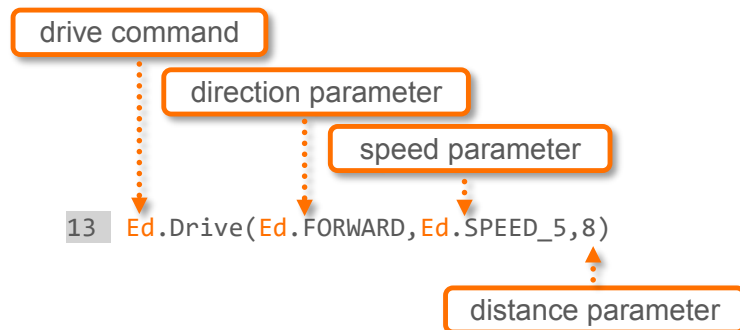
Once you have written the full program, click the 'Check Code' button and look at the 'Compiler Output' window to make sure you haven't made any errors while typing.

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 to your Edison.

1. Turn Edison on, then press Edison's record (round) button once
2. Connect Edison to the computer using the EdComm cable and confirm the volume is turned up to full
3. Press the 'Program Edison' button in the upper right corner of the EdPy app
4. Follow the steps in the pop-up window, then press 'Program Edison'

Once the program downloads, unplug the EdComm cable. Use activity sheet 2.1 or coloured tape to mark 'start' and 'finish' lines on a desk or the floor as a test area for your

program. Press the play (triangle) button once to run the program and watch what happens.

Step 6: Experiment with your program.

Measure the distance between your start line and finish line. Try to modify your program to make your Edison robot finish driving just before the finish line. Experiment to see what works.

Your turn:

1. What constant did you have 'Ed.DistanceUnits' set to in the Setup code?

    _____

2. What number did you need to enter as the distance input parameter to make Edison drive from the start to the finish line?

    _____

3. Experiment with driving the Edison robot at different speeds. What does the robot do when you drive Edison at speed 10? Do you notice any changes in the accuracy of Edison when it is driving at speed 10?

_____

_____

_____

_____

Lesson 2: Worksheet 2.2 – Drive the robot backwards

In this activity, you need to write a program to drive your Edison robot backwards.

Whenever you write a program for Edison in EdPy, you always follow the same basic steps:

- **Step 1:** Check the Setup code is using the constants you want.
- **Step 2:** Write the program, selecting the functions you want to use, and filling in the input parameters with the values you want.
- **Step 3:** Check your program for errors using the 'Check Code' button.
- **Step 4:** Download and test your program using your Edison robot.

Remember, if you have Edison V1 robot, make sure that Ed.DistanceUnits = Ed.TIME. The distance parameter for Ed.TIME is in milliseconds.

Your turn:

Task 1: Drive backwards

Write the following 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.BACKWARD,Ed.SPEED_5,8)
14 |
```

Task 2: Use the constant 'Ed.DISTANCE_UNLIMITED'

There are multiple ways of programming your Edison to drive forward and backwards. Another way is to use 'Ed.DISTANCE_UNLIMITED' for the distance parameter. This constant turns both of Edison's drive motors on.

Unlike when you use a number value for the distance parameter, Ed.DISTANCE_UNLIMITED doesn't specify an exact value after which the motors will stop. Instead, it turns the motors on and then moves on to the next line of code. A stop of the motors will be needed later in the code.

Using Ed.DISTANCE_UNLIMITED can be useful when you want to write a program where the motors only stop once some other event occurs, for example, when an obstacle is detected.

Look at the following 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.BACKWARD, Ed.Ed.SPEED_6, Ed.DISTANCE_UNLIMITED)
14  Ed.TimeWait(200, Ed.TIME_MILLISECONDS)
15  Ed.Drive(Ed.STOP, Ed.SPEED_10, 0)
16
```

This program turns on the Edison motors to drive backwards, then waits for 200 milliseconds before turning the motors off.

Write a program using the Ed.DISTANCE_UNLIMITED parameter to drive backwards.

1. Set the speed in your program to Ed.SPEED_6. How many milliseconds do you need to input to the TimeWait() function to make Edison drive backwards from the finish to the start line?

   _____

2. Experiment with different speed and TimeWait() input parameters. What are the fastest and the slowest times you can make Edison drive backwards from the finish to the start line?

   Fastest: _____   Slowest: _____

Lesson 2: Worksheet 2.3 – Forwards, then backwards

In this activity, you need to write a program to drive your Edison robot forward, then backwards.

Remember to follow the four basic programming steps for Edison using EdPy:

- **Step 1:** Check the Setup code is using the constants you want.
- **Step 2:** Write the program, selecting the functions you want to use, and filling in the input parameters to the values you want.
- **Step 3:** Check your program for errors using the 'Check Code' button.
- **Step 4:** Download and test your program using your Edison robot.

Remember, if you have Edison V1 robot, make sure that Ed.DistanceUnits = Ed.TIME. The distance parameter for Ed.TIME is in milliseconds.

Your turn:

Task 1: Write the following 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  Ed.Drive(Ed.BACKWARD,Ed.SPEED_5,8)
15  |
```

1. What are the correct distance input parameter values so that Edison drives forward, then backwards between the start and the finish lines?

   Forward _____   Backwards _____

Task 2: Write a new program that makes Edison drive forward, then backwards between the start and the finish lines. This time, use the Ed.DISTANCE_UNLIMITED parameter and Ed.TimeWait() functions. Use Activity sheet 2.1 or coloured tape to mark 'start' and 'finish' lines on a desk or the floor as a test area for your program.

Lesson 2: Worksheet 2.4 – Expressions in Python

In this activity, you will learn about an important element of code we use when programming in Python: expressions.

What are expressions?

An expression is a question that can be resolved as being either 'true' or 'false.' For example: 'Is A less than B?' or 'Is A the same as B?'

In code, expressions are written using mathematical notations instead of words.

In the Setup code, you have seen the = notation being used. For example, Ed.DistanceUnits = Ed.CM. Using the 'A = B' notation means 'set A to the same value as B.'

Expressions also use notations. These are some of the basic notations in expressions we can use in Python:

Expression	Meaning
A == B	Is A the same as B?
A != B	Is A not equal to B?
A > B	Is A greater than B?
A >= B	Is A greater than or equal to B?
A < B	Is A less than B?
A <= B	Is A less than or equal to B?

Expressions compare the left side to the right side of the notation in the expression.

You can replace the 'A' and 'B' in the list of expressions above with any value or function that returns a value. You can also do maths to those values. For example, '(A + 2) > B' means 'Is A plus 2 greater than B?'

In code, expressions work in a specific order. When your expression includes maths to a value or calls a function, the expression will resolve the maths or function first. It will then compare the left side of the expression to the right side and resolve to either 'true' or 'false.'

What are expressions used for in Python?

Expressions are used along with other elements of code, such as 'while' loops and 'if' statements, to change the flow of code. These elements allow the code to move differently than just in the sequential flow of line 1 → line 2 → line 3.

Your turn:

Practice resolving expressions. First, write out what each expression means, then resolve it to either true or false.

If A = 2 and B = 4, what does each of the following expressions mean and what does each resolve to (true or false)?

1. (A*2) == B

   Meaning: _____

   Resolves to: _____

2. A >= B

   Meaning: _____

   Resolves to: _____

3. (A+A) != B

   Meaning: _____

   Resolves to: _____

4. (A-1) < (B-3)

   Meaning: _____

   Resolves to: _____

Lesson 2: Worksheet 2.5 – Keypad activated driving

In this activity, you need to write a program to drive your Edison robot forward only when either the round button or the triangle button is pressed. To do this, we will use expressions and the 'while' loop.

Look at the following 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  Ed.ReadKeypad()
13  while Ed.ReadKeypad() == Ed.KEYPAD_NONE:
14      pass
15  Ed.Drive(Ed.FORWARD, Ed.SPEED_6,8)
16
```

This program uses a 'while' loop with an expression.

In Python, a 'while' loop repeats a statement or group of statements while a given condition is TRUE. It tests the condition, which is written as an expression, before executing the loop body.

While the expression evaluates to TRUE, the program repeats the commands in the loop. When the expression evaluates to FALSE, the program moves on to the next line of code outside the loop.

Using indentation

Python uses indentation to group statements or commands together.

In Python, all the statements indented by the same number of character spaces are considered to be a single block of code.

Look at line 14 of the program. Because 'pass' is indented, it is inside of the loop. Line 15 in the program is not indented, however, so line 15 is not inside of the loop.

Functions and constants in this program

Ed.ReadKeyPad() – this function reads Edison's keypad state. In other words, it determines whether one of Edison's buttons has been pressed or not. Ed.ReadKeyPad() returns a value indicating which button has been pressed: Ed.KEYPAD_NONE, Ed.KEYPAD_TRIANGLE or Ed.KEYPAD_ROUND.

This function does not work for the square button. That's because the square button is designed only to be used to stop a program. The square button will always stop any program running when it is pressed.

Special note: using 'read' functions inside a loop

Some types of data get temporarily stored in Edison's memory. That's how the Ed.ReadKeyPad() function may read a button press from before the read function is called in your code.
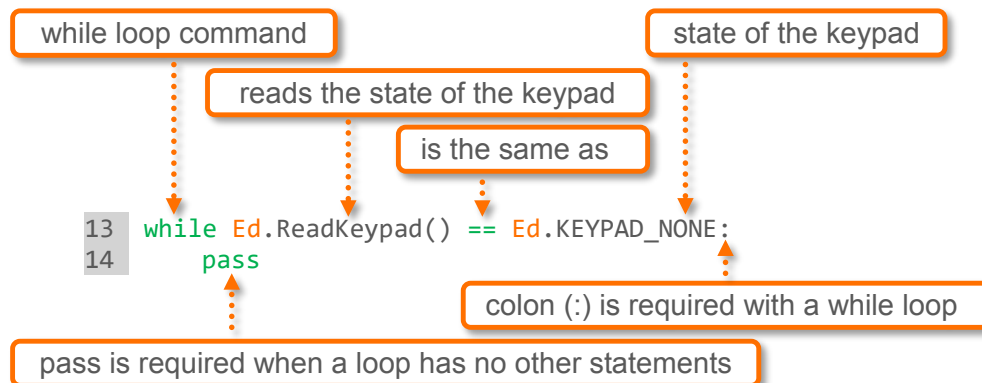
In this program, we want to make sure the Ed.ReadKeyPad() in the while loop will wait until a button has been pressed and not consider any button presses that happen before the while loop. That's why we put Ed.ReadKeyPad() into the line above the while loop (line 12). This will clear any previous key presses before the loop.

You should always follow this process when using a 'read' function inside a loop.

Your turn:

Write the program.

Be sure that when you write the while loop, you include the colon and that you indent correctly:

while loop command		state of the keypad

	reads the state of the keypad	

	is the same as	

```
13   while Ed.ReadKeypad() == Ed.KEYPAD_NONE:
14       pass
```

colon (:) is required with a while loop

pass is required when a loop has no other statements

Download the program and press the triangle button to start the program. Wait a bit, then try pressing either the triangle or round button.

1. What did the robot do when you pressed the triangle or round button a few seconds after starting the program?

_____

_____

2. Run the program again and try pressing the square button instead of the round or triangle button. What happened? Why did this happen? Explain.

_____

_____

_____

_____

3. Now try adding some more code at the end of the program. The new code you write should make Edison drive backward after you press either the round or the triangle button again. In other words, your program should tell Edison to drive forward the first time a button is pressed, then backwards when a button is pressed again. Remember to include the colon and to indent your code inside the while loop. What does your new program look like? Write your code down below.

_____

_____

_____

_____

_____

_____

_____

_____

Lesson 2: Activity sheet 2.1

FINISH LINE

START LINE