OZOBLOCKLY SKILLS 6
USE FUNCTIONS

ESSENTIAL QUESTION

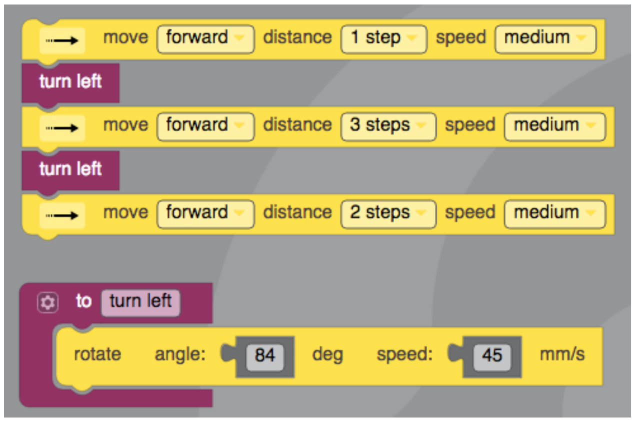
Can students use **functions** with “get wheel speeds” and “rotate by xx degrees” to move Ozobot over a path of straight lines and right angles?

OVERVIEW

So far, functions have been an option in your coding repertoire. In this lesson, functions are formally introduced and prove their usefulness. Students get practice using them, whether they’ve learned about them before or this is their first time.

The purpose of functions is to save a few, or many, lines of code under one name, and then be able to use that set of code whenever you need. While Loops help you with a repeated action, functions help you use the action at any time.

In our coding repertoire, we have a special method for turning. Let’s say we have to turn only once in a while on a map. Instead of writing:
Forward 1 step
Rotate 85 at 30mm/s
Forward 3 steps
Rotate 85 at 30mm/s
… and then have to edit every rotation that is mentioned, we save a lot of time and energy by only editing one block that’s inside a function called *turn left*:



```
move forward distance 1 step speed medium
turn left
move forward distance 3 steps speed medium
turn left
move forward distance 2 steps speed medium

to turn left
    rotate angle: 84 deg speed: 45 mm/s
```

In this lesson, students emulate an old-school game called Snake by making functions for the individualized forward movement and rotations learned in the preceding OzoBlockly Skills lessons. Those functions become moves in the game, and these moves help Ozobot reach all of the apples.

LESSON OUTLINE

1. Students begin by finding the best wheel speeds and right angled turns for their Ozobot.
2. They put their discovered wheel bias into functions.
3. Pairs or groups write and test their programs for completing the Snake game.
4. Students present their successful solutions to the class or teacher.

PREREQUISITES

OzoBlockly Skills 1-5

GROUPING

Pairs or groups of three

MATERIALS

- Tablet or computer with OzoBlockly editor ozoblockly.com/editor
- Ozobot Bit or Evo, one per group
- Printout of Snake Map per group
- Printout of Grid per group

GRADE LEVEL

Grades 2 and up

OZOBLOCKLY PROGRAMMING TOPICS

Modes 3 and 4

DURATION

50 minutes

VOCABULARY

- *Function* - a computer programming concept wherein a code that is needed often is saved under a special name, and can be used anywhere in the current program.
- *Function Call* – the block of code (or the variable, in text-based programming languages) that is used in the main program and runs what is in the function.
- *To Call* - this verb is used to express when a program reaches the function block in the main program and then executes the code inside of the function block at the side.

QUESTIONS ABOUT THIS LESSON?

Please contact us at ozoEdu@ozobot.com

LESSON

PART 1 SETTING WHEEL BIAS

Begin with students going into **Pair Programming** groups or pairs.

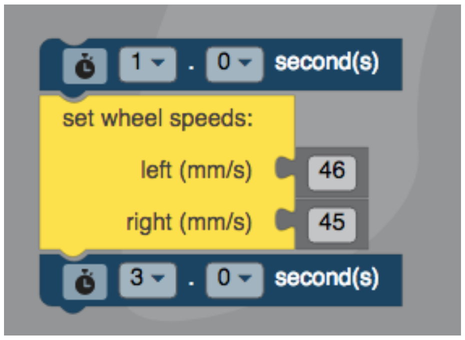
When you see this, have your Pair Programmers switch roles.

Give each group the instructional handout. The first page of tasks can be done independently, but all students should follow the instructor for the second page, where functions come in.

STEP 1

Students' first task is to find the best way to code "go straight". Have students find the 'set wheel speed' values for their Ozobot. Each group or pair has a grid on which to test their Ozobot's straightness. Students write down their best speeds on the paper and keep their programs from each step.

During this testing phase, the Ozobot can be charged when not in use to make sure it will perform its best.

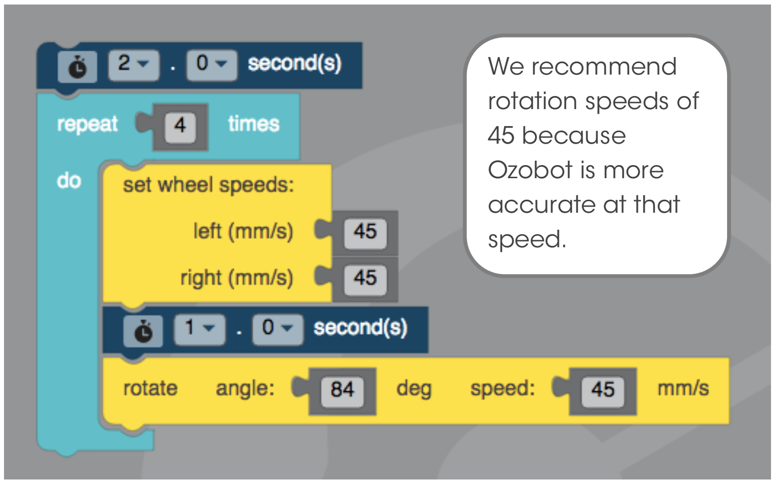


```
1 . 0 second(s)
set wheel speeds:
    left (mm/s) 46
    right (mm/s) 45
3 . 0 second(s)
```

EXAMPLE Wait one second, move forward for three seconds.

STEP 2

Next, students use this forward movement to test going straight and turning left to make a square. Example:

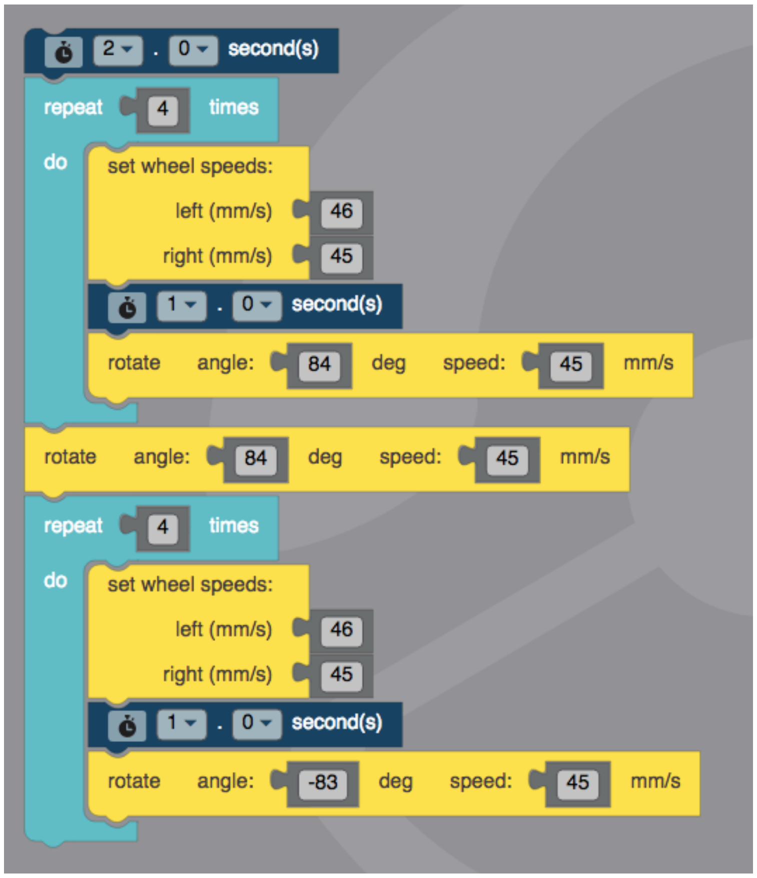


```
[wait] 2 . 0 second(s)
repeat 4 times
do  set wheel speeds:
        left (mm/s) 45
        right (mm/s) 45
    [wait] 1 . 0 second(s)
    rotate  angle: 84 deg  speed: 45 mm/s
```

We recommend rotation speeds of 45 because Ozobot is more accurate at that speed.

ROLES SWITCH

STEP 3 Finally, students add on to the bottom the code to test going right:



```
[wait] 2 . 0 second(s)
repeat 4 times
do  set wheel speeds:
        left (mm/s) 46
        right (mm/s) 45
    [wait] 1 . 0 second(s)
    rotate  angle: 84 deg  speed: 45 mm/s
rotate  angle: 84 deg  speed: 45 mm/s
repeat 4 times
do  set wheel speeds:
        left (mm/s) 46
        right (mm/s) 45
    [wait] 1 . 0 second(s)
    rotate  angle: -83 deg  speed: 45 mm/s
```

EXAMPLE pause 2 seconds, repeat 4 times: go forward one second, turn left; turn left, repeat 4 times: go forward one second, turn right.

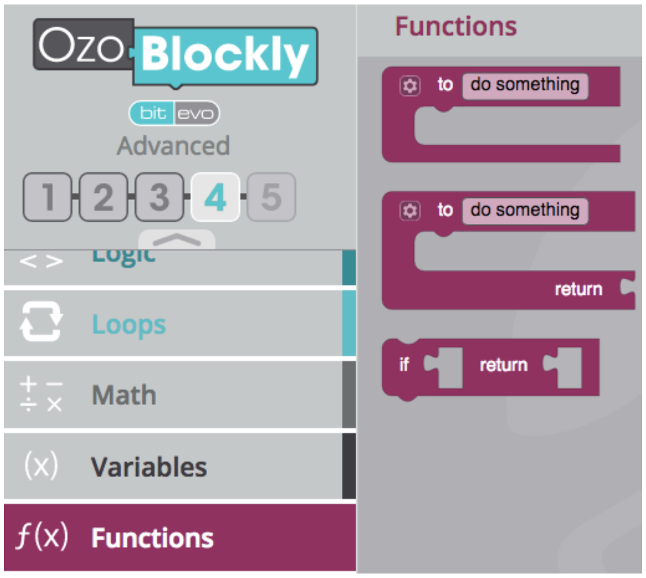
Currently the program demonstrates making a counter-clockwise square, then a clockwise square over the same area. Keep this code for the next section!

PART 2 MAKE FUNCTIONS

ROLES SWITCH

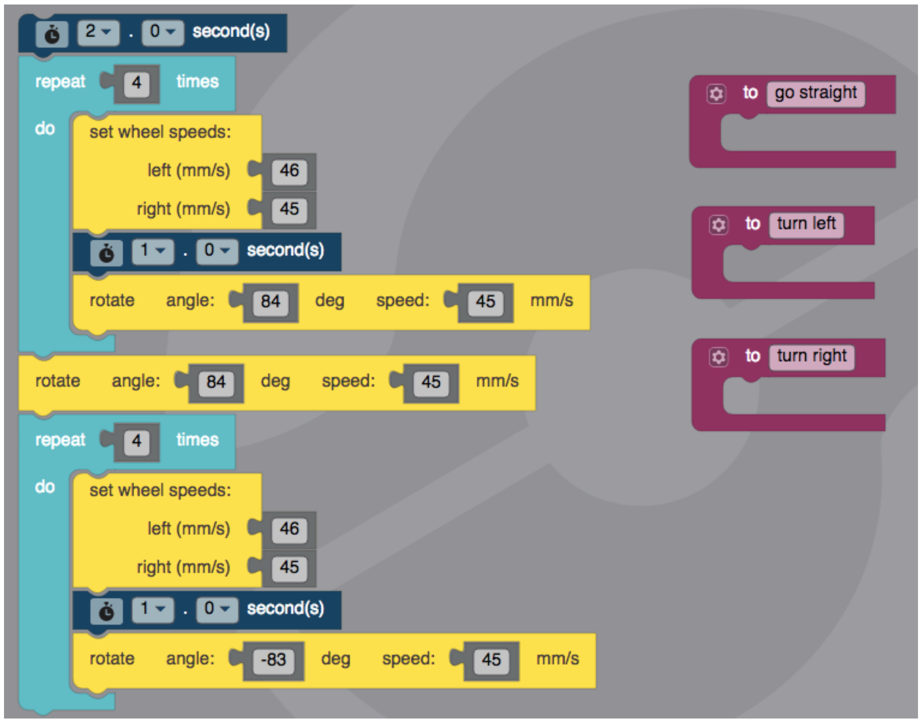
Lead students through these steps to set up functions:

1. Go to Mode 4, Functions, and choose 'to do something', on the top,

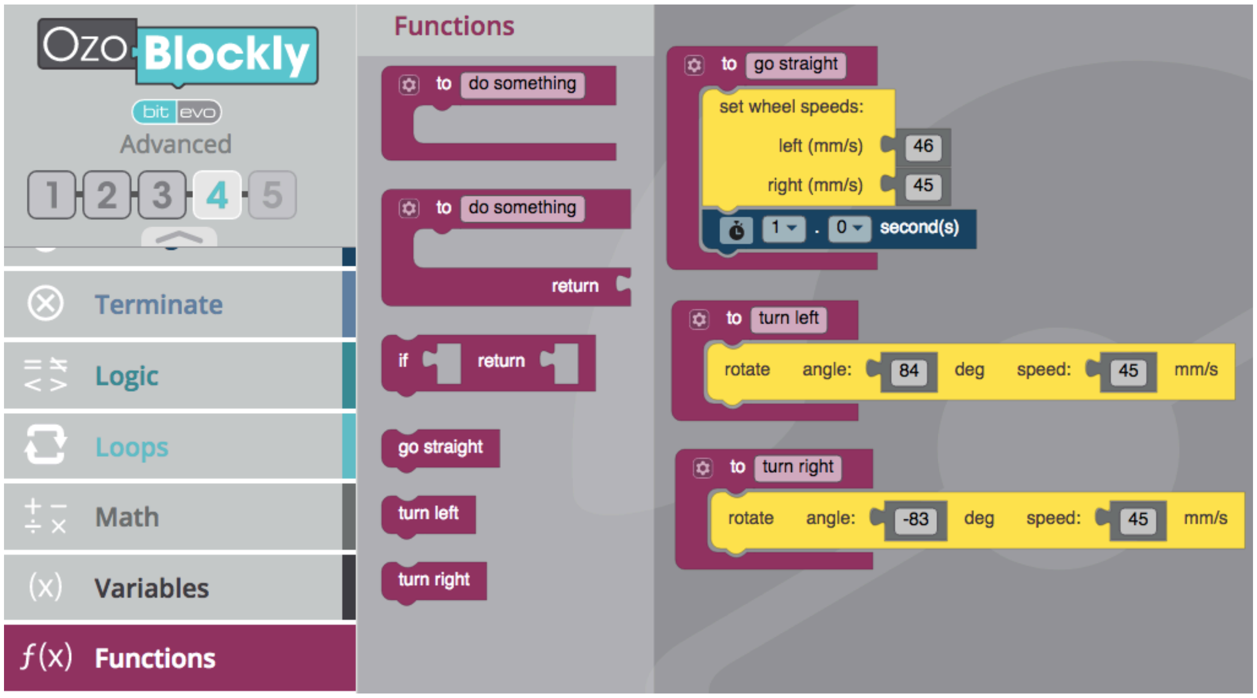


CamelCase is a way that programmers save names for files, functions, etc. In programming languages, spaces aren't allowed in function names, but sometimes the names need to be descriptive. That's why we use camelCase - the first word can be lower-case, but each following word is capitalized (yes, even those tiny prepositions and articles): goStraight, makeAUTurn, twoMinuteDance.

2. Select that block, and rename it to 'go straight'. Optionally, use CamelCase.
3. Select two more, and call one turn left and another turn right:

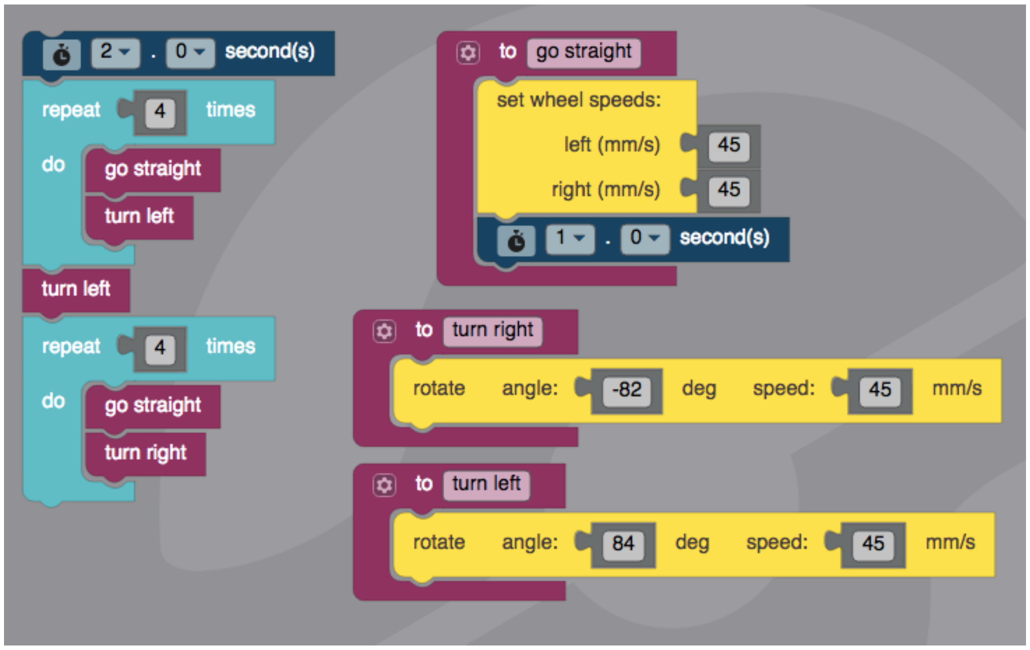


4. Students drag code from the main program for going straight into that **function** on the right, and do the same with the turns.



```
to go straight
  set wheel speeds:
    left (mm/s) 46
    right (mm/s) 45
  1 . 0 second(s)

to turn left
  rotate angle: 84 deg speed: 45 mm/s

to turn right
  rotate angle: -83 deg speed: 45 mm/s
```

5. To "call" the function in the main program, use the new blocks that appeared in the Functions menu in Mode 4, called **function calls** (see above). Place those blocks in the main program (see below).
6. Test the program, and tweak the function to get the program to achieve the goal! See below:

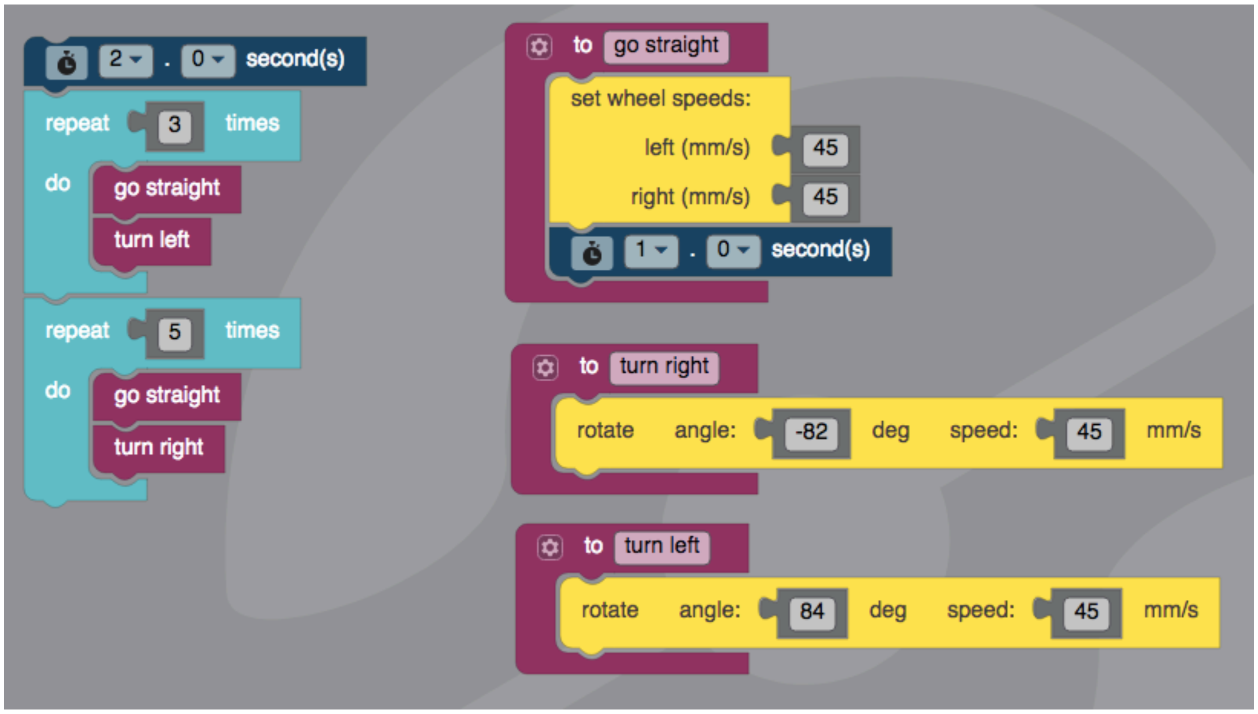


```
2 . 0 second(s)
repeat 4 times
do
  go straight
  turn left
turn left
repeat 4 times
do
  go straight
  turn right

to go straight
  set wheel speeds:
    left (mm/s) 45
    right (mm/s) 45
  1 . 0 second(s)

to turn right
  rotate angle: -82 deg speed: 45 mm/s

to turn left
  rotate angle: 84 deg speed: 45 mm/s
```

You'll notice the values in the functions on the right are different from the previous examples.
That's because Ozobot needed some more tweaking once the program was set up.
This shows how using functions makes editing the program quick and easy!

Students can now use those functions anywhere in this program. **Warning**: if the function block is deleted, so is the function call block used in the main program. The function blocks must stay in the screen!

You can choose to allow students to experiment with functions. Maybe a Figure 8?



```
2 . 0 second(s)
repeat 3 times
do  go straight
    turn left
repeat 5 times
do  go straight
    turn right

to go straight
  set wheel speeds:
    left (mm/s) 45
    right (mm/s) 45
  1 . 0 second(s)

to turn right
  rotate angle: -82 deg speed: 45 mm/s

to turn left
  rotate angle: 84 deg speed: 45 mm/s
```

Class Responds: Why would we make a function, instead of just putting all the code blocks together? Possible answers are that it:

- shortens the actual running code,
- makes it easier to read for human eyes,
- is easier to edit,
- reduces bugs,
- keeps the programmer more organized,
- makes the program run more efficiently on the computer, etc.

ROLES SWITCH

PART 3 WRITE SNAKE PROGRAM

Students now have the tools they need to program the Snake Game.

Hand out the map to each group. Point out that their goal is to cover all of the apples by using their new moves. Allow 10-15 minutes for programming and testing. Students will have to debug their program (fix problems they find during testing) while writing the program. Pair Programmers can switch roles midway through.

RULES:

1. Start with the line-follow block to get Ozobot straight.
2. Ozobot must fully cover the apples to "eat" them.
3. All function calls must be used, and loops are allowed.
4. Any route can be taken to reach all the apples, but the most efficient and quickest route is best.

Remember: Students should charge their Ozobot while working.

PART 4 PRESENT TIGHTROPE PROGRAM

Student present their final program! Everyone watches in anticipation to see if Snake Ozobot manages to eat all of the apples!

EXTENSIONS

1. Style Points. This activity can be repeated, this time with style points! Students must throw in some cool moves while playing the game. Can Ozobot go backwards and stay straight? Maybe a quick spin? Time to show off!
2. Make a new game! Students design a game that requires forward, left and right movements, and maybe backwards or something new. Can their peers win the new game?

ozobot Testing Grid

Create Functions

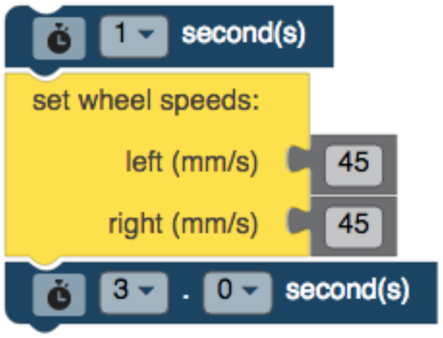
Names___________________

____

PART 1 SETTING WHEEL BIAS

STEP 1 GO STRAIGHT

Write a program like this:



```
⏱ 1 second(s)
set wheel speeds:
    left (mm/s) 45
    right (mm/s) 45
⏱ 3 . 0 second(s)
```

Change the wheel speeds to make your Ozobot go straight. Use the grid to test straightness.

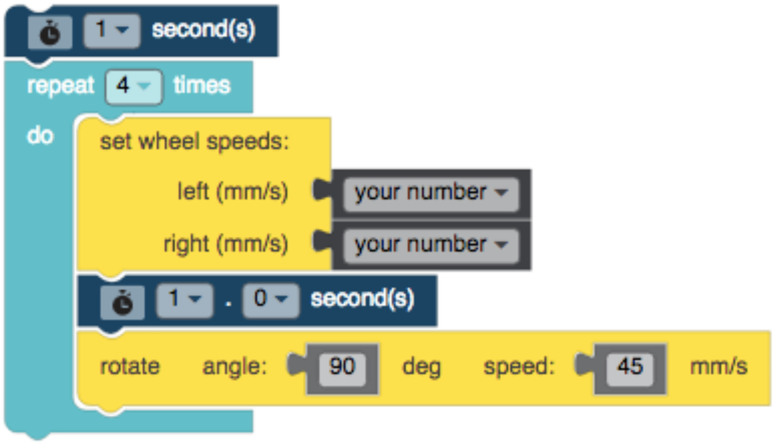
Our Ozobot's best wheel speeds:

left ______ mm/s

right _______ mm/s

STEP 2 TURN LEFT

Add to the last program to make this program:



```
⏱ 1 second(s)
repeat 4 times
do
    set wheel speeds:
        left (mm/s) your number
        right (mm/s) your number
    ⏱ 1 . 0 second(s)
    rotate angle: 90 deg speed: 45 mm/s
```

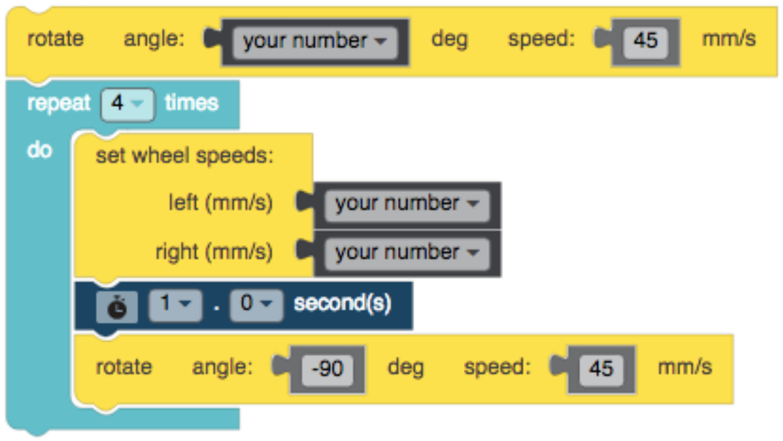
Replace "your number" with the numbers you found in Step 1. Change the angle to make a perfect left turn.

Our Ozobot's left angle:

_____ deg

STEP 3 TURN RIGHT

Attach this to the bottom of the last program:



```
rotate angle: your number deg speed: 45 mm/s
repeat 4 times
do
    set wheel speeds:
        left (mm/s) your number
        right (mm/s) your number
    ⏱ 1 . 0 second(s)
    rotate angle: -90 deg speed: 45 mm/s
```

To make a right turn, the angle must be a negative number.

Our Ozobot's right angle:

_____ deg

Your program will now make a left-turning square, then a right-turning square.

Keep your whole program for the next step!

CREATE FUNCTIONS

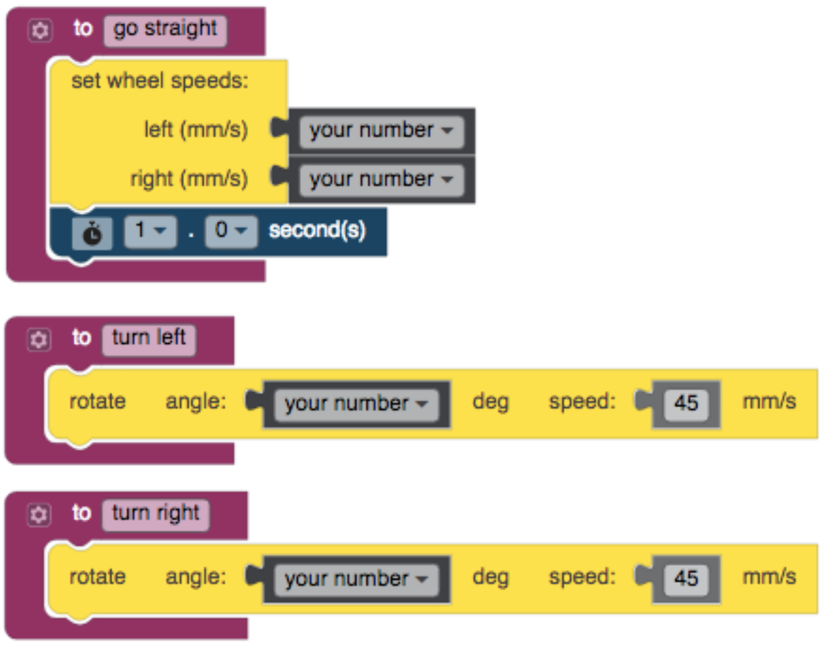
NAMES____________________

____

PART 2 CREATING FUNCTIONS

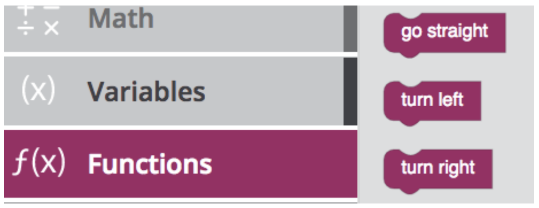
Use your program from Part 1. Do the following:

1. In OzoBlockly, choose Mode 4;
2. Choose f(x) Functions category;
3. Select "to do something", a purple block that can hold other blocks;
4. Drag "to do something" to the right of your program;
5. Change the name to "go straight";
6. Select two more function blocks, name them "turn left" and "turn right".
7. Drag the blocks for each function to inside the function, like below:



```
to go straight
    set wheel speeds:
        left (mm/s)  your number
        right (mm/s) your number
    1 . 0 second(s)

to turn left
    rotate angle: your number deg speed: 45 mm/s

to turn right
    rotate angle: your number deg speed: 45 mm/s
```

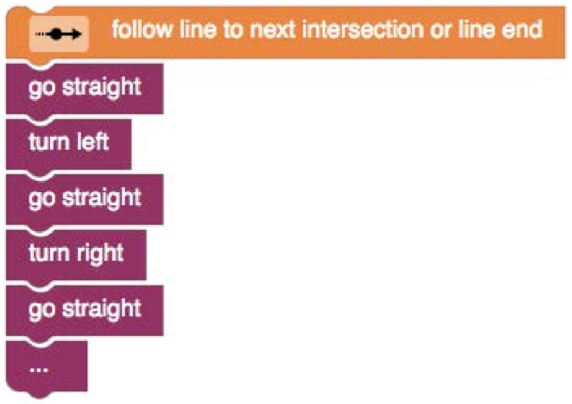
8. Find your Function Calls in Mode 4 f(x) Functions:



9. Replace the movements with the new function call blocks. These function calls are all you need to write your game program.

PART 3 WRITE SNAKE PROGRAM

You can now write your Snake program! Use only the Function Calls to complete the map.

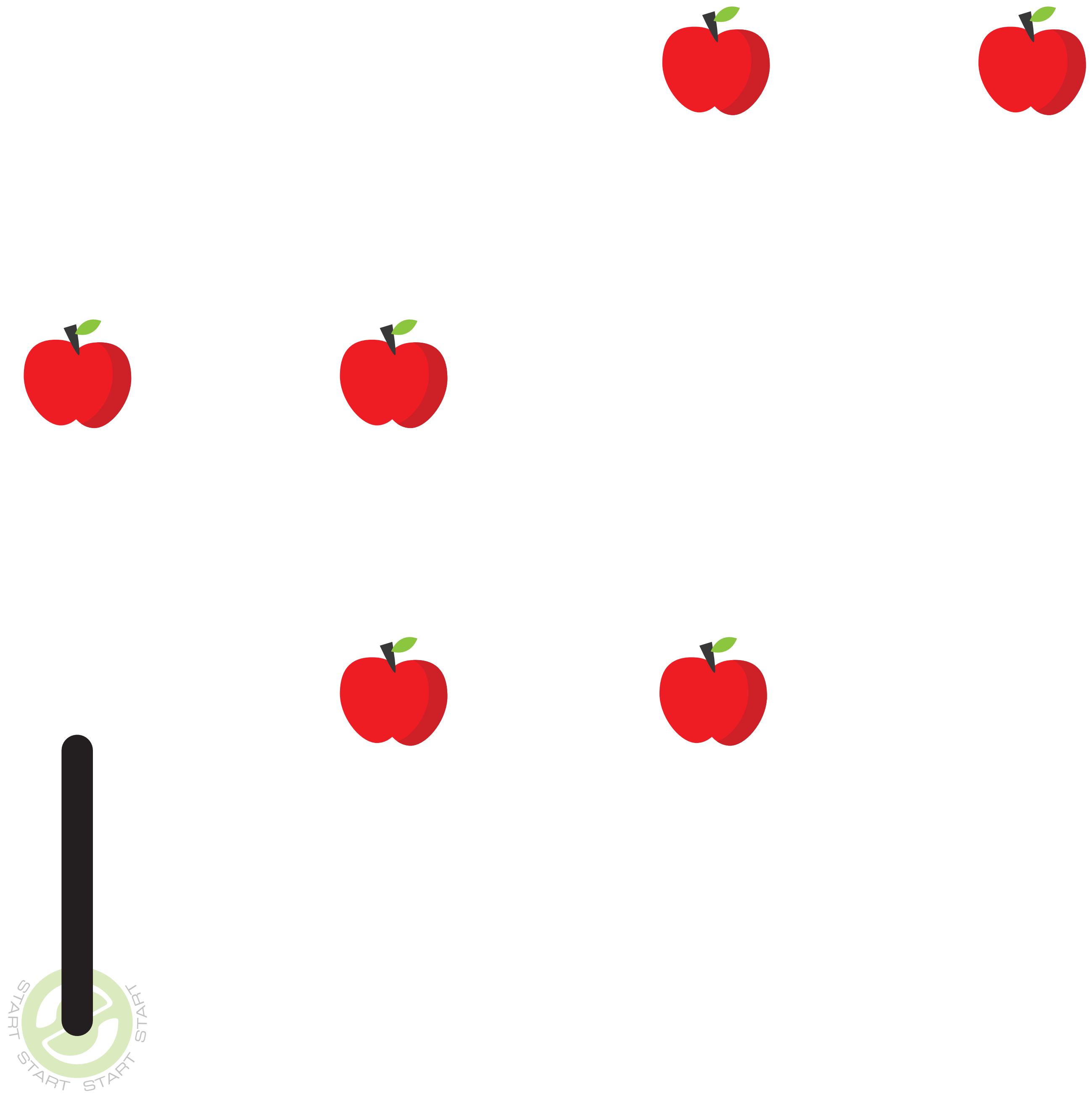


```
follow line to next intersection or line end
go straight
turn left
go straight
turn right
go straight
...
```

RULES:

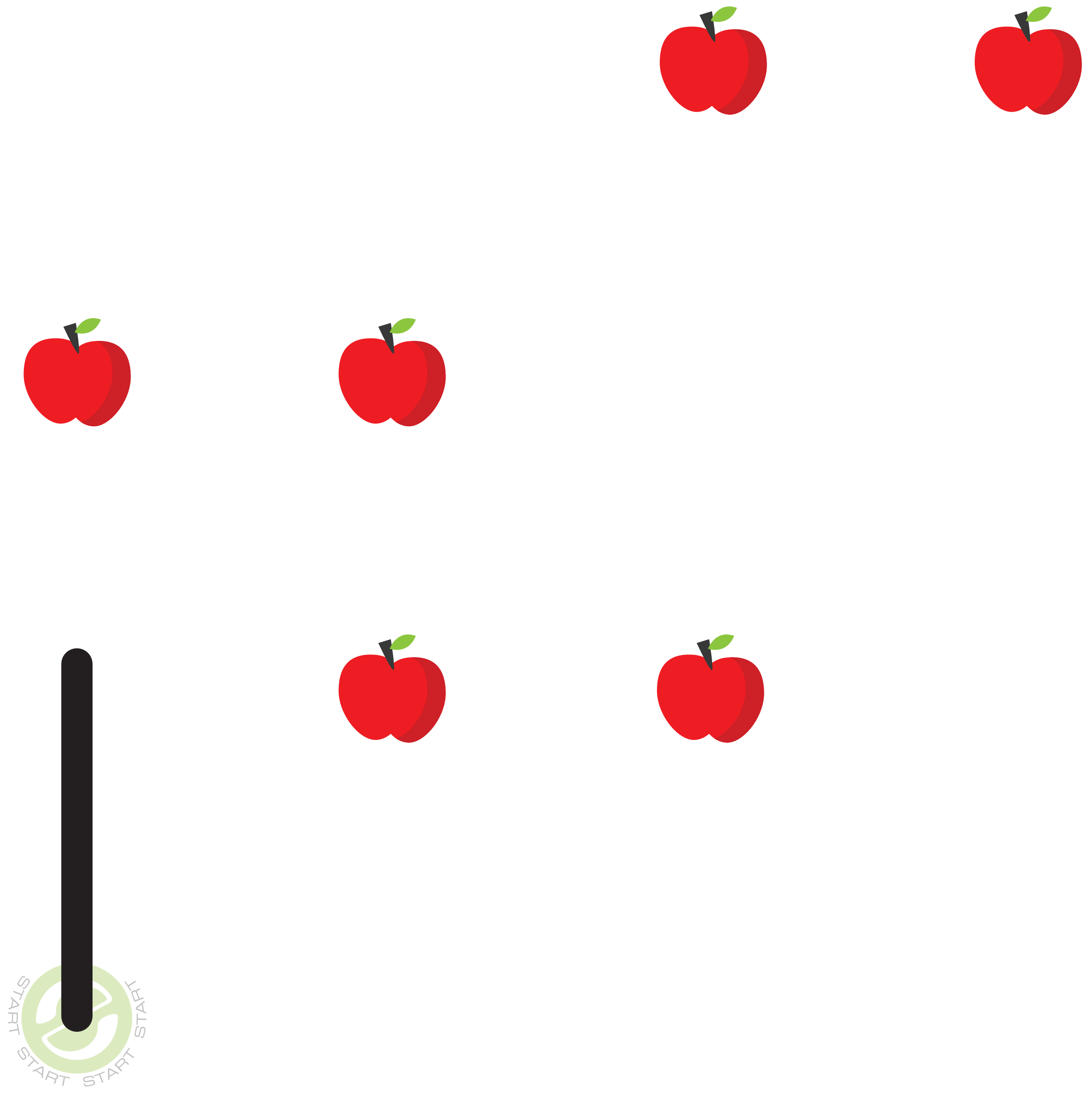
1. Start with the line-follow block to get Ozobot straight.
2. Ozobot must fully cover the apples to "eat" them.
3. All function calls must be used, and loops are allowed.
4. Any route can be taken to reach all the apples, but quickest route is best.

evo



ozobot SNAKE MAP

bit



ozobot SNAKE MAP