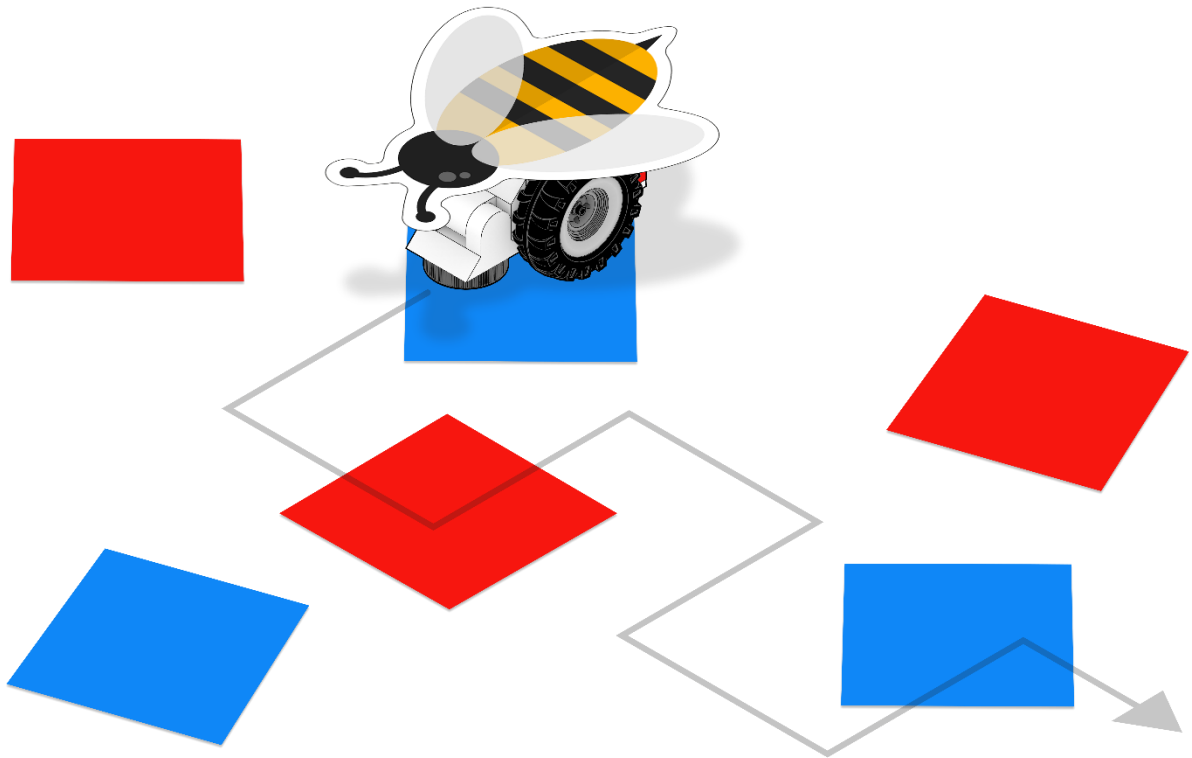




# The Foraging of the Bee

## Teachers Guide

For secondary level I and II



Learning  
Level  
2



This teaching material supports schools and teachers at secondary level I and II in helping children and young people to develop the areas of competence defined in the competence framework of the Conference of Ministers of Education and Cultural Affairs. Their aim is to teach young people safe, creative, and responsible use of media, and to foster individual and self-directed learning, thus enabling them to competently participate in a digital society.

Concept and implementation: Helliwood media & education, Kinematics GmbH

Design and typesetting: Helliwood media & education, Kinematics GmbH

Authors: Helliwood media & education, Kinematics GmbH

Title of the work: Learning with robots II

Rights holder: Kinematics GmbH

Format of the work: Multiple formats

Version: 2.0 | 20.01.2021

## Content

<b>Foreword .....</b>	<b>4</b>
Overview .....	4
<b>Lesson 1 – Preparation for variables .....</b>	<b>6</b>
Introduction topic bees .....	6
Preparation of the simulation .....	7
Program the search pattern of a bee .....	7
Additional task: Complex search pattern .....	9
<b>Lesson 2 – Define variables .....</b>	<b>10</b>
Explanation of terms: Variable .....	10
Make the search pattern of the bee variable .....	10
Additional task: Logical operation .....	12
<b>Lesson 3 – Assign a new value to a variable .....</b>	<b>14</b>
Introduction .....	14
Increase the search radius of the bee gradually .....	14
Additional task: Signalize recognition .....	16
<b>Lesson 4 – Variables in a new program .....</b>	<b>18</b>
Introduction .....	18
Let the bee fly back again .....	18
Additional task: Spiral .....	20
<b>Lesson 5 – One variable and various tasks .....</b>	<b>21</b>
Introduction .....	21
The dance of the bee .....	21
Additional task: Speed up zigzag .....	23
<b>Lesson 6 – Combine multiple variables .....</b>	<b>24</b>
Introduction .....	24
Complete the dance of the bee .....	24
Additional task: Turning circle .....	26
<b>Lesson 7 – Get to know functions .....</b>	<b>28</b>
Introduction .....	28
Structure the tasks of the bee .....	28
Additional task: Swap out even more blocks to functions .....	30
<b>Lesson 8 – Adjust program code with functions .....</b>	<b>31</b>
Preparation .....	31
Return to the beehive .....	31
Additional task: Unsuccessful search .....	33
What is the next step? .....	34

## Foreword

With Tinkerbot's products, students can be taught knowledge in the areas of robotics, programming and computational thinking. The overarching approach of Tinkerbots is to promote the playful, haptic and creative transfer of knowledge that is relevant for the future.

The topics of robotics and programming have become indispensable in a digitalized world. It is therefore necessary to convey these future technologies to children to prepare them for the challenges and opportunities of the 21st century.

STEAM skills are not only important for children who are looking for a career in IT, engineering or research. For every child, the use of media and modern teaching aids is an advantage. However, Tinkerbots products not only promote the teaching of classic STEAM skills, but also strengthen self-confidence, problem-solving abilities and social skills.



Science



Technology



Engineering



Arts



Mathematics

## Overview

The present **material "The Foraging of the Bee"** contains thematically the programming concepts variables and functions. The material is based on curricula of computer science education and expands the students' competencies in the fields of computer science, programming and robotics.

The concepts of variables and functions are transferred in the material using an example from biology and are implemented with the robots. In the tasks it concerns to simulate the food search and communication possibilities of bees with the help of Tinkerbots. The bee is the "red thread" that leads through the lessons. The material can thus be used across disciplines with links to the natural sciences/biology department.

The robots described in the material can be built with the Tinkerbots Education Sets (Basic and Expert). The tasks in the lessons can be solved with learning level 2 of the Tinkerbots Blockly App.

The material contains eight lessons of 45 minutes each. The time specification for the individual units is based on experience. Depending on the learning ability of the students, they can be individually varied and adapted. Each lesson unit focuses on a central task that can be solved by all students at their individual learning pace. This means that the learners always have an additional task at their disposal if required.

In addition to the teacher's material, task cards are available for each lesson for the students. The cards are designed in such a way that they can be worked on by the learners in a self-discovering way and as independently as possible. Further tip cards can be issued as assistance if required. With a solution card the own program code can be checked for correctness. It should always be noted that there is not THE one solution, but that several solutions can lead to the desired result.

<b>Age group</b>	From grade 5
<b>Area</b>	STEAM subjects and robotics work group
<b>Learning Level</b>	This unit uses the categories and blocks of learning level 2 in the Tinkerbots Blockly programming app
<b>Prior knowledge</b>	Tinkerbots Blockly learning level 1 Concepts commands, loops Tinkerbots-Module Powerbrain and Multisensor
<b>Competence framework</b>	Pupils know and understand the functionality and basic principles of the digital world
<b>Social form</b>	Class discussion and partner work
<b>Material</b>	Computer (Windows, Mac) or Tablet (iOS, Android) Tinkerbots Education Set, Bluetooth-Dongle (Windows), Power supply unit Assembled Basic Robot Tinkerbots Blockly App Task cards, Tips & Solutions Cardboard (blue, red, green), scissors, bee template
<b>Lessons</b>	<b>Introduction variables</b> 1. Preparation variables: Programming the search pattern of a bee 2. Define variables: Making the search pattern of the bee variable
	<b>Change variables</b> 3. Assign a new value to a variable: Increase the search radius of the bee gradually 4. Use variables in another context: Let the bee fly back again
	<b>Combine variables</b> 5. One variable and different tasks: Programming the bee dance 6. Combine several variables: Complete the dance of the bee
	<b>Functions</b> 7. Getting to know functions: Structuring the tasks of the bee 8. Adapt the program code with functions: Return to the beehive

## Lesson 1 – Preparation for variables

In programming, the principle of the variable is used to define contexts for flexible values with a kind of placeholder and to not commit to a number.

To enable the students to understand the usefulness of a variable, a preparatory program is written in this lesson, which then introduces the variable in the next step. The lesson is a form of repetition of the knowledge already acquired and enables the students to apply already known building blocks, such as the loops, in a new context.

Furthermore, lesson 1 serves to introduce the topic bees and to prepare the simulation.



### Introduction topic bees

Right at the beginning of the series of lessons, establish the link between real bees and the programming of the Tinkerbots. Also, explain what a simulation is.

#### Lesson talk

Discuss with the students what you already know about bees, for example what different roles there are in a colony, how they forage for food or how bees communicate with each other. To support the discussion, you can also show a film about bees, let the students do research or post pictures of bees in the room.



You can decide yourself how detailed you want to treat the topic bees. Important is to deal with the search for food and the special kind of communication of the bees among themselves, the tail dance. A cooperation with the department biology is suitable.

#### Transition to programming

In a bee colony there are different roles. The role of the queen is the best known. However, the so-called track bee also has a particularly important role, because it finds new sources of

food. The bee flies off the area, looks for new sources of food and then passes the information where the sources are located on to its bee colleagues. In the following tasks we will use the Tinkerbots as bee robots to simulate how the bee does this.



For the lessons, the students need an assembled Basic Robot. They can optionally design the robot as a bee and have the bee template glued on, which is included in the material.

### Preparation of the simulation

In order to let the Tinkerbots robots search for food as bees, a flower meadow with different flowers as food sources is required.

Let the students distribute color cards in blue and red throughout the room. Use the color cards included in the Education Set. These are optimized with the RGB values for the sensor of the Tinkerbot and can be recognized without any problems. The color cards now symbolize the flowers on a meadow, which the bees can fly to forage for food.



If the detection of a color by the sensor does not work under certain lighting conditions, it may help to increase the distance from the sensor to the color. In most cases it is sufficient to let the sensor point diagonally forward, in rare cases it is necessary to remove the Lego adapter plate completely.

### Program the search pattern of a bee

Once the simulation is prepared, the students can solve the main task.

#### Task 1

In this task, your Tinkerbot simulates a bee looking for food. The bee "flies" a pattern in order to be particularly successful in its search for flowers and at the same time save energy.

Think of a certain pattern that your Tinkerbot "flies off" as a bee to find the flowers of your flower meadow. When your robot bee finds a red flower, it should give a signal by playing a tone or melody.

## Example solution 1



### Tip 1

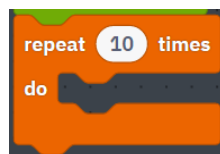
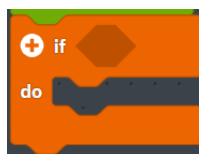
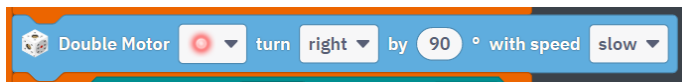
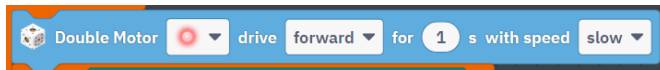
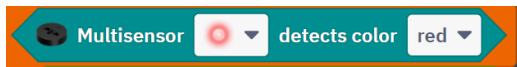
It is up to the students to decide which form of search pattern they use. A meander, for example, is a good choice.





## Tip 2: Required blocks

Do the students need a little reminder to support them? The Multisensor module is used for color recognition. To play sounds you need the Powerbrain.



## Additional task: Complex search pattern

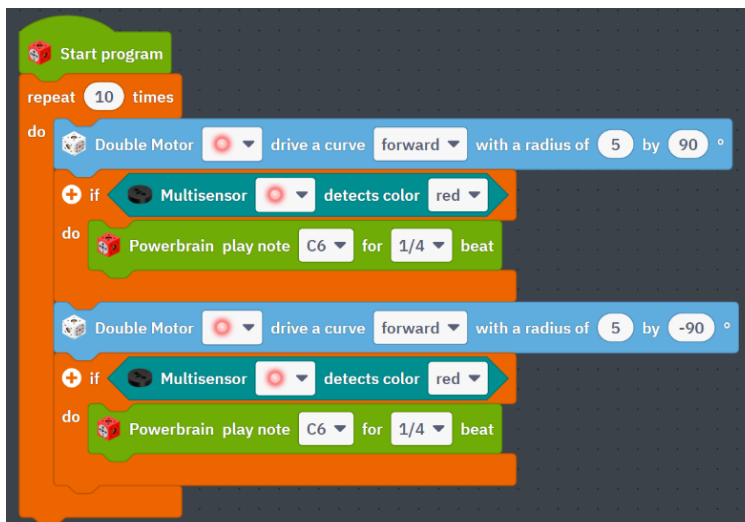
Once the students have solved the main task, they can try an additional task.

### Additional task 1

Let your bee search with a wavy movement pattern.

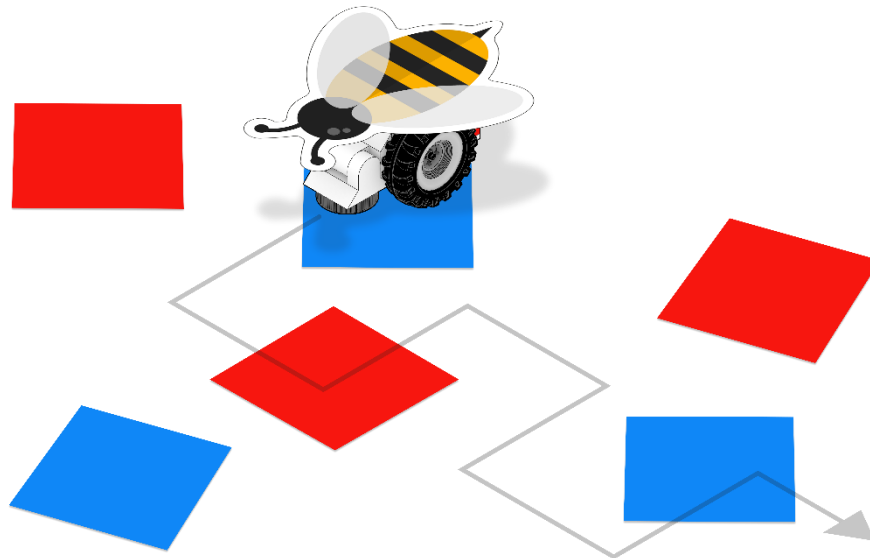


### Example solution additional task 1



## Lesson 2 – Define variables

Lesson 2 introduces the concept of the variable. The students learn how to assign new values to their bee using the placeholder, thus making their program more flexible. By only one single change in the program code, you can tell the bee to fly a different radius.



### Explanation of terms: Variable

#### Lesson talk

Reflect together with the students at the beginning on how they proceeded in programming the search pattern. Talk about the fact that the bee in the simulation searches the same route every time it takes off. If it did this every day, all food sources would soon be exhausted.

It is important for the bee that the route it takes can be changed. It means its variable. In the simulation and in the program code, for example, the time the bee flies in one direction can be changed. Now the variable comes into play. Once defined and used, the value of the variable can be reassigned again and again. The variable can be named uniquely and is thus made "speaking".

### Make the search pattern of the bee variable

Show students where to find the variable in Tinkerbots Blockly and give the main task of the lesson.

#### Task 1

Make the distance the bee flies variable (or in other words: changeable), so that the bee can search in a larger radius. For this the bee must fly longer in one direction. Think about where in your code you want to make the numerical value variable.

## Example solution 1



### Tip 1

Variables can not only be called x and y when programming but can be given any name. To keep a good overview, the variables can be made "speaking". Instead of an x as the variable for the flight time, the students can name the variable *Flight time*.

**Tip 2: Required blocks**

The block *Variable* can be found in the *Variables* section. In order to be able to add the suitable numbers, one selects a block of numbers from the column *Math*.

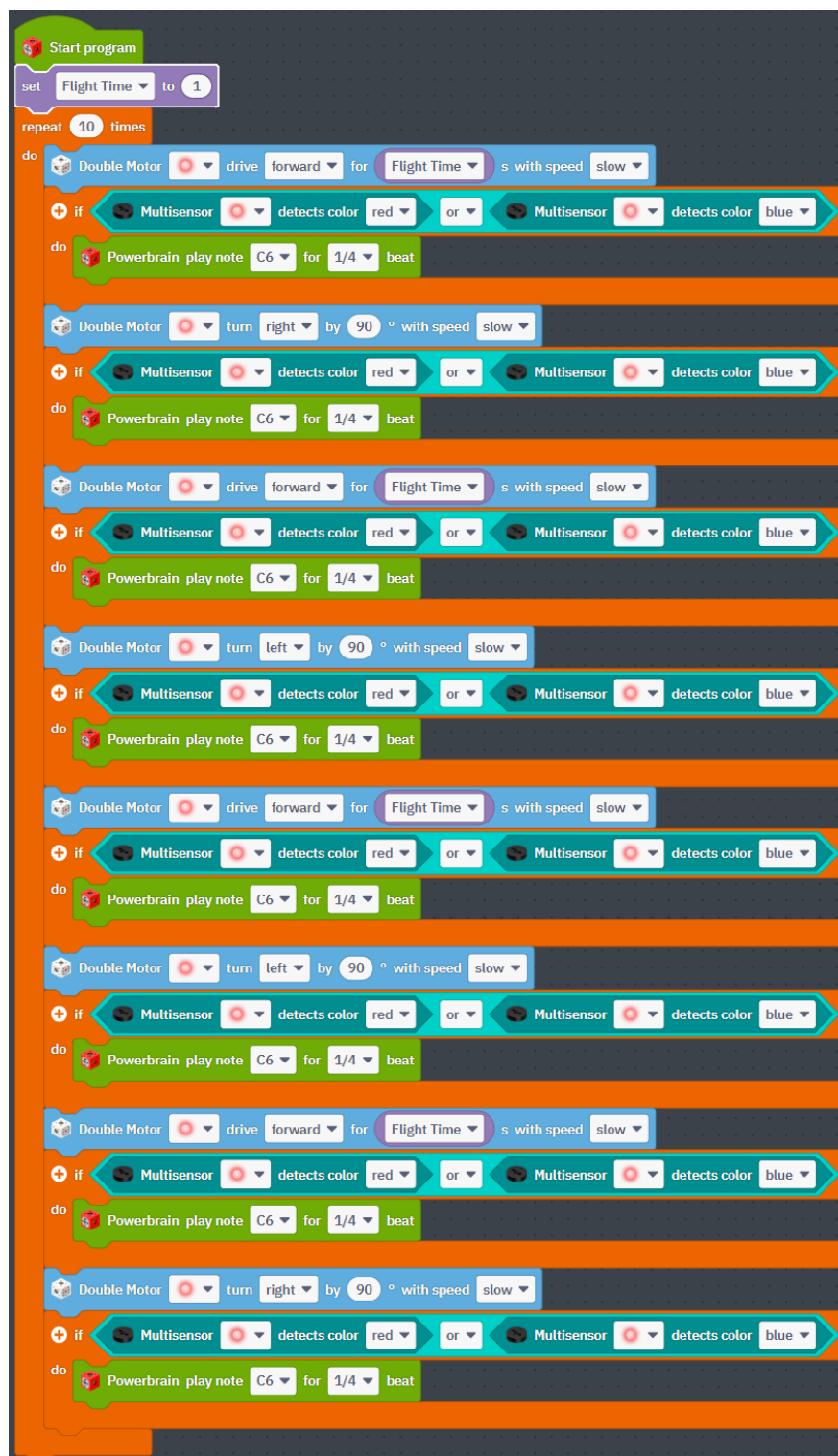
**Additional task: Logical operation**

Once the students have understood the concept of the variable, the transfer into the code will proceed quite quickly. This leaves time for an additional task.

**Additional task 1**

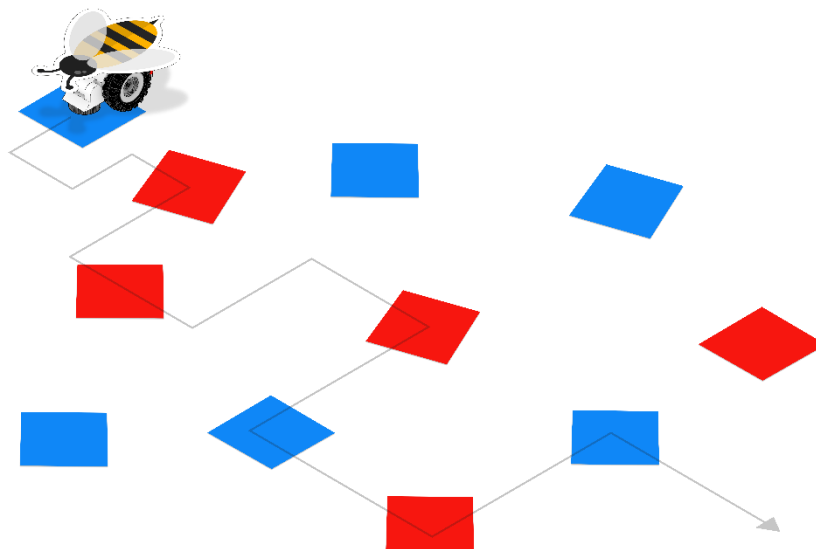
The Tinkerbot should play a tone with both colors, red OR blue. Use the blocks from the *Logic* section.

## Example solution additional task 1



The grammatically correct name of the task should be red AND blue. But programmatically this is a logical OR-conjunction. The integration of the word OR into the task is a little help.

In lesson 3, students learn how to change a variable and assign a new value to it. They apply mathematical structures and let the program do the calculations for them. In this way they achieve that the bee increases its flight radius step by step.



## Lesson talk

Reflect with the students what they have achieved with the last task. Instead of changing each value for the time, the flight time can be easily adjusted with one value, by introducing a variable.

Talk to the students about how to optimize the search for food if the bee first searches for flowers nearby and then gradually flies further and further away.

To do this, you can change the program so that the variable increases after each repetition loop.

## Task 1

Extend your code so that the bee first searches very close to you and then slowly moves further and further to find more flowers.

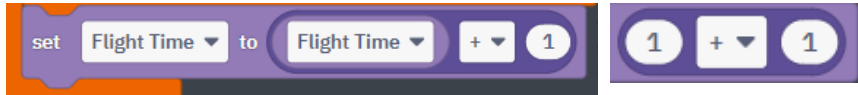
## Example solution 1



### Tip 1: Required blocks

If necessary, give the students a tip to look at the blocks in the section *Math*. There they will find the arithmetic procedure *Addition*.

If you combine this block with the variable *Flight time* instead of the first 1 and redefine the variable, you get the following statement: *set flight time to flight time + 1*.



### Tip 2: Explanation

The construct that a new value is assigned to a variable may need some explanation. Take a close look with the students at how this was done:

At the very beginning the variable *Flight time* was assigned the fixed number 1. If you look at the new block, you will see that at the end of the block there is *Flight time + 1*. So, the number 1 is added to the variable at this point. Since the variable *Flight time* still corresponds to the number 1, the following calculation is performed:  $1 + 1 = 2$ .

With the block *set Flight time to* the variable *Flight time* now corresponds to the value 2.

As soon as the inserted block is called again, the number 1 is added again. The calculation process  $2+1=3$  is executed, and the variable has the value 3 from this point on.

Thus, the program continues to run and the variable increases step by step by the value 1.

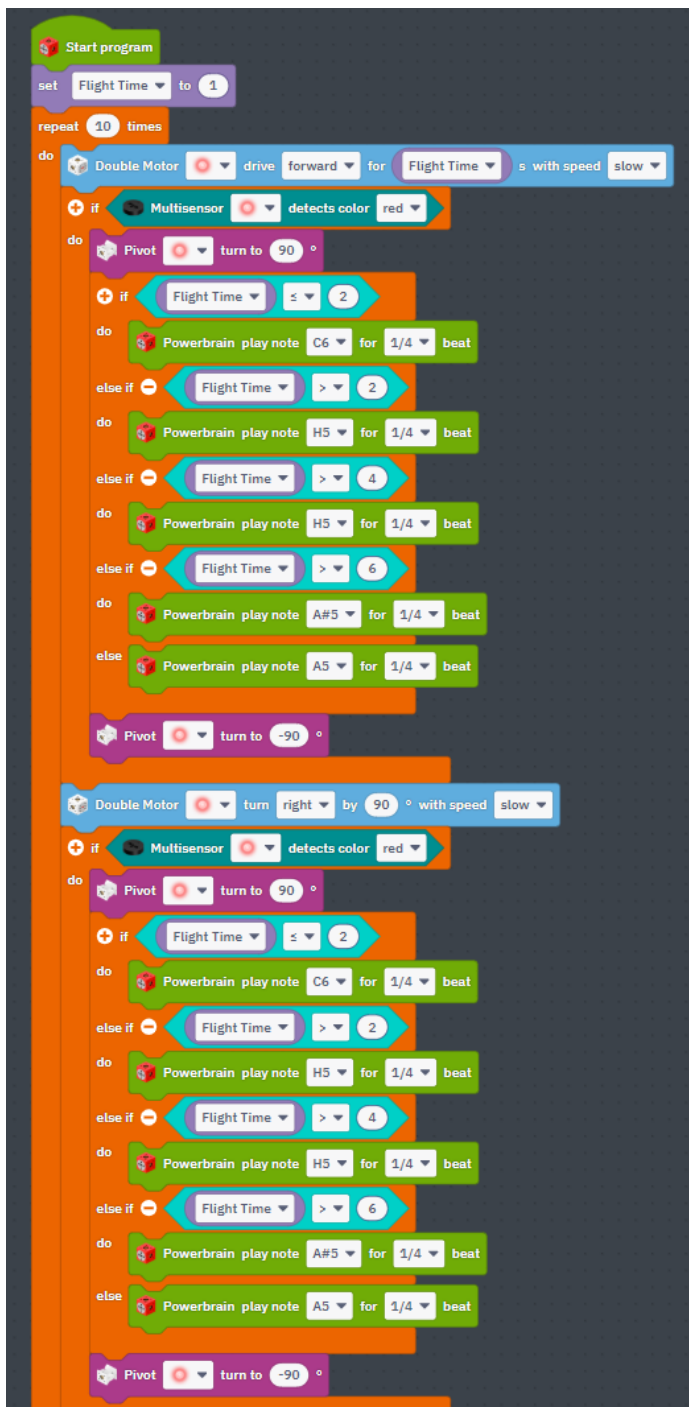
### Additional task: Signalize recognition

#### Additional task 1

If the bee recognized a flower, it is to raise and lower the head again and make a sound in each case.



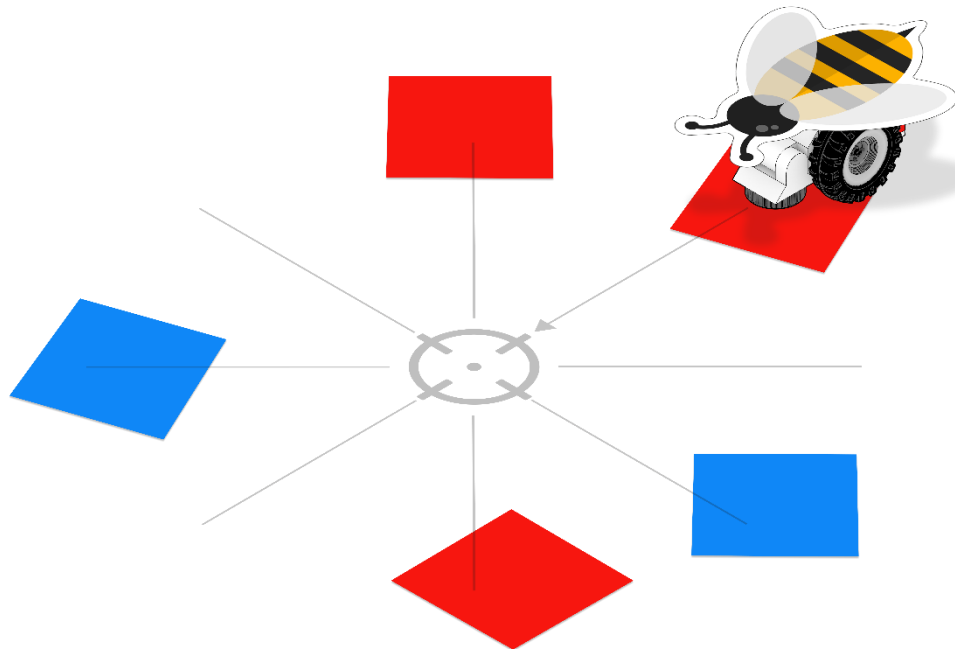
## Example solution additional task 1



This is only a part of the solution. You can find the complete code in the solution sheet.

## Lesson 4 – Variables in a new program

The aim of lesson 4 is to apply the concept of the variable in a new program and thus to deepen and consolidate the knowledge. This time the bee is programmed in such a way that it not only flies away in search of food, but also returns to the starting point.



### Introduction

#### Lesson talk

Compare once again with the pupils the foraging of a bee with the simulation. In the previous search patterns, one central moment was missing: that the bee would fly back to its hive.

Consider with the students what another search pattern might look like that takes this new aspect into account.

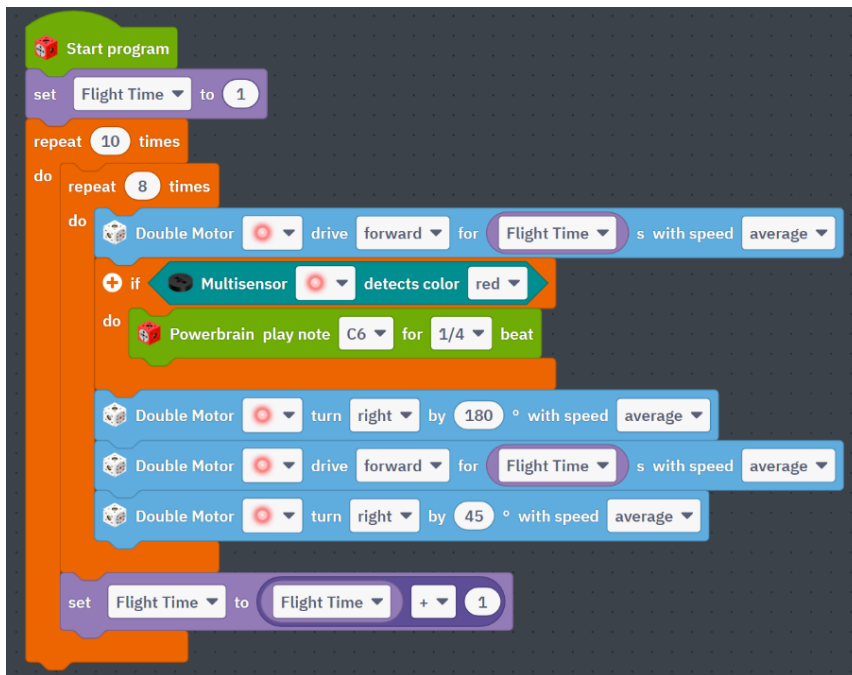
### Let the bee fly back again

#### Task 1

Think of a search pattern where the bee always returns to the start. Begin with the fact that the bee looks for food only in the immediate proximity around it.

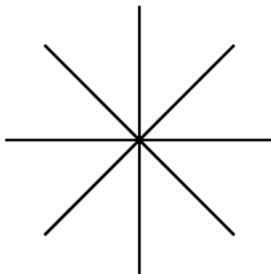
If the bee has searched nearby, let it in the next rounds always fly a piece further. Use variables again for this.

### Example solution 1



#### Tip 1

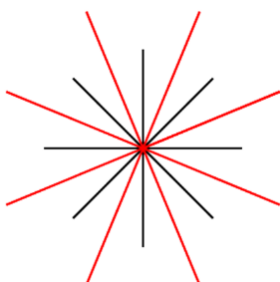
A good pattern for fulfilling the task is a star shape where the bee always flies the same way there and back.



#### Tip 2

It is best to start with a rather small radius and then gradually increase it. The red star shows the search pattern after the variable has been increased by +1.

But the pattern also works the other way around. That means, instead of adding, you can also subtract and always deduct 1. Here you must set the variable larger at the beginning.



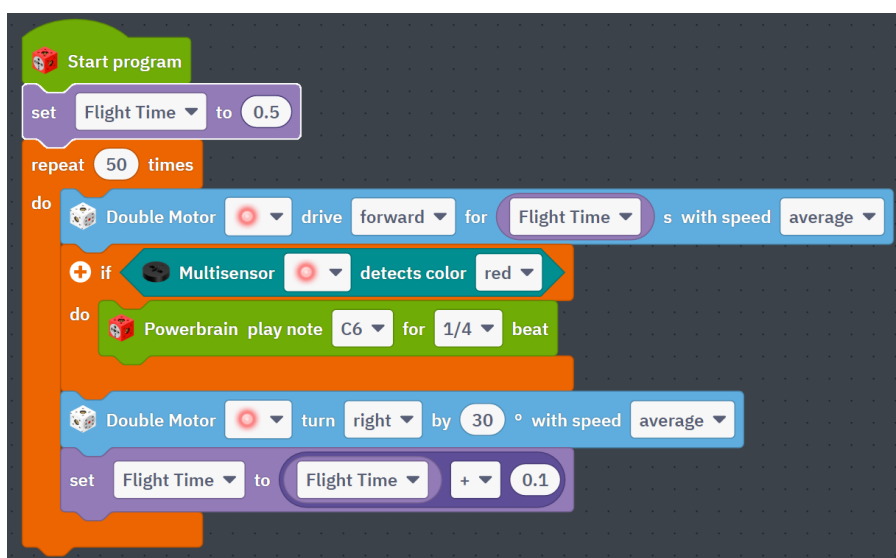
### Additional task: Spiral

With the additional task of lesson 4 you can lure students who like to puzzle out of their reserve. Now they must think carefully about where to use the variable.

#### Additional task 1

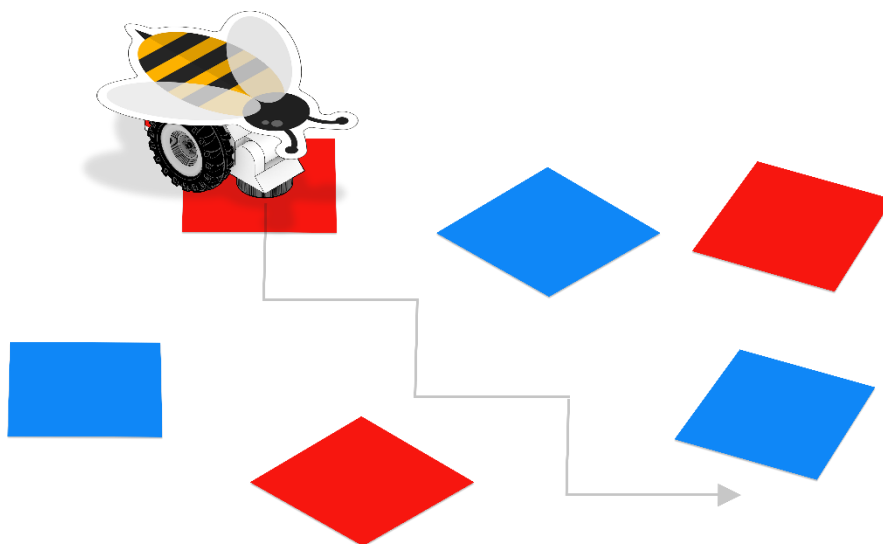
Program your bee to move away from the center in a spiral shape to search for new flowers. Can you make the bee find all the flowers?

#### Example solution additional task 1



## Lesson 5 – One variable and various tasks

In lesson 5 a new behavior of the bee is added: communication. Here the students learn that one and the same variable can even be used for various tasks.



### Introduction

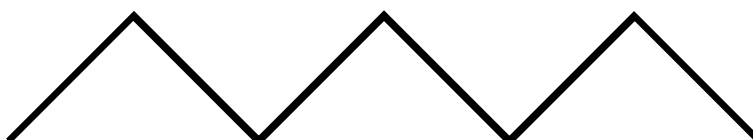
#### Lesson talk

Talk to the students about the special communication of bees and how they tell each other where the best sources of food are: For this purpose, the scout bee performs a dance. In the dance the scout bee tells the forager where they can find nectar, how much there is to find and how far the forager bees must fly.

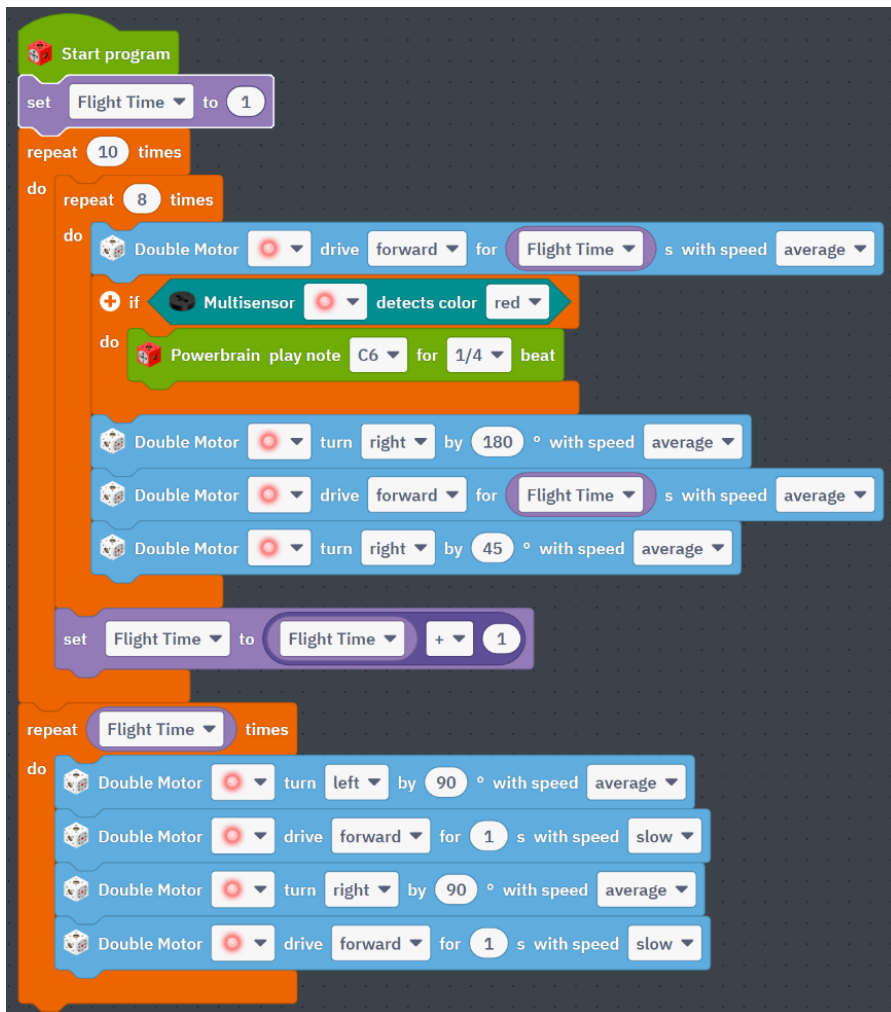
### The dance of the bee

#### Task 1

Program your robot bee to perform the following zigzag dance after the successful and completed search for food. The zigzag movement tells you how far it has flown in its search for food. The farther, the more zigzags the dance has.



### Example solution 1

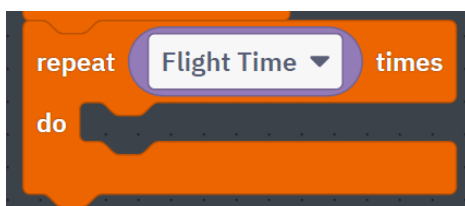


#### Tip 1

The existing program code of the foraging (e.g., the star pattern) is supplemented by the new blocks of the dance. In the dance, this time the length of the strokes is the same, but the number of repetitions is variable.

#### Tip 2

For the distance of the foraging, we had used the variable *Flight time*. Since the forager bee needs to know how far it must fly, the variable *Flight time* comes into play for the dance as well. Since the distance of the food in the dance is determined by the number of jags, we add the variable *Flight time* for the number of repetitions. You need this block:

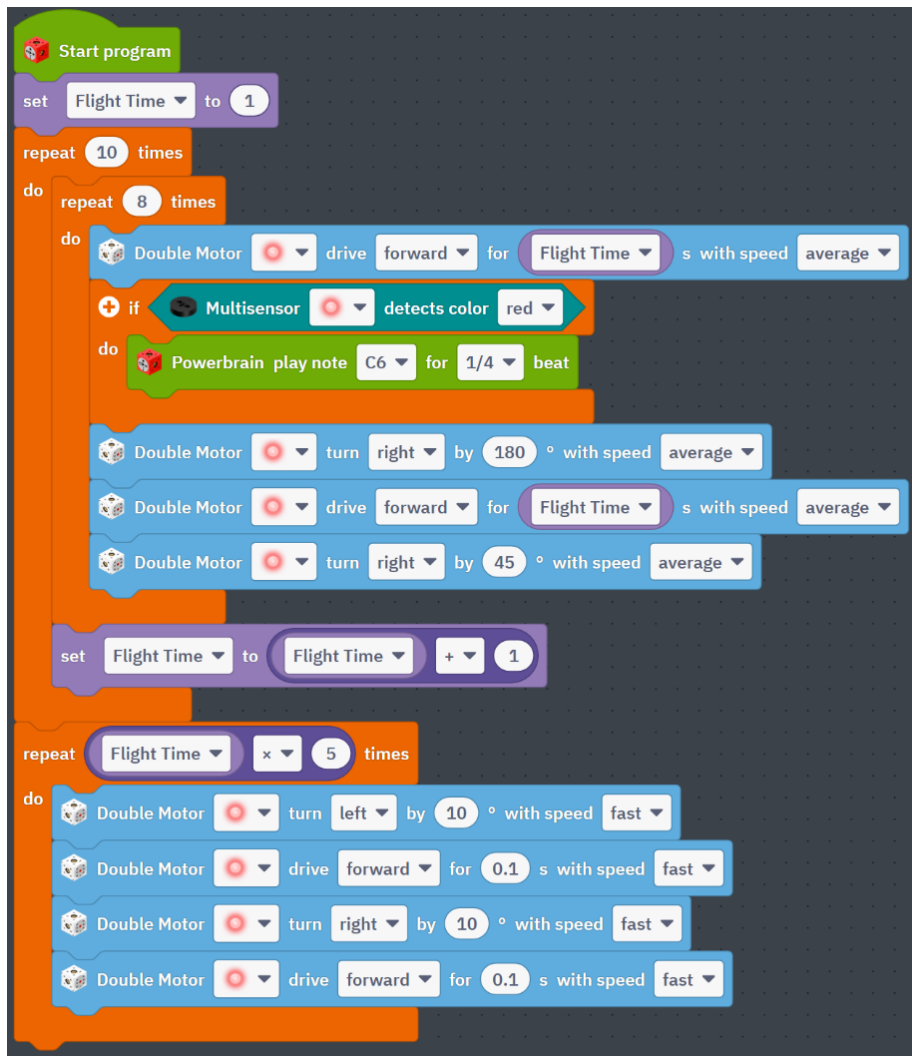


## Additional task: Speed up zigzag

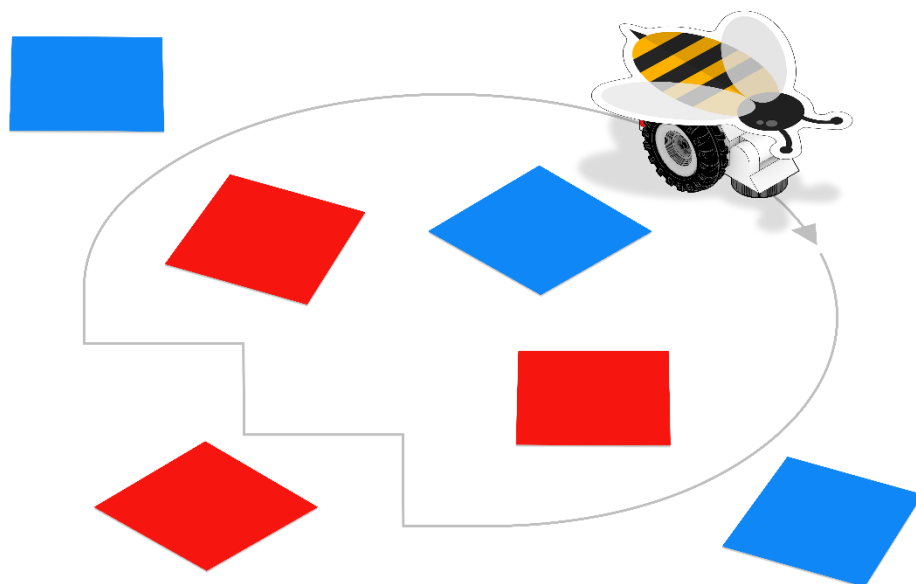
### Additional task 1

The scout bee is moving very fast on the zigzag course, as if the bee is trembling. Think of a method to make the bee tremble. For this purpose, it is useful to increase the value of the variables for the dance.

### Example solution additional task 1



After the students have already used the same variable to solve different problems or tasks, the next step is to define and use another variable.

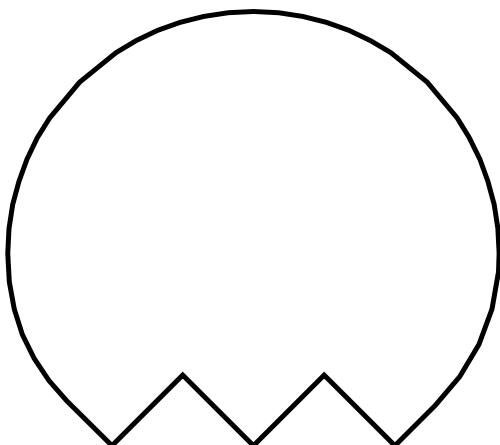


## Lesson talk

## Complete the dance of the bee

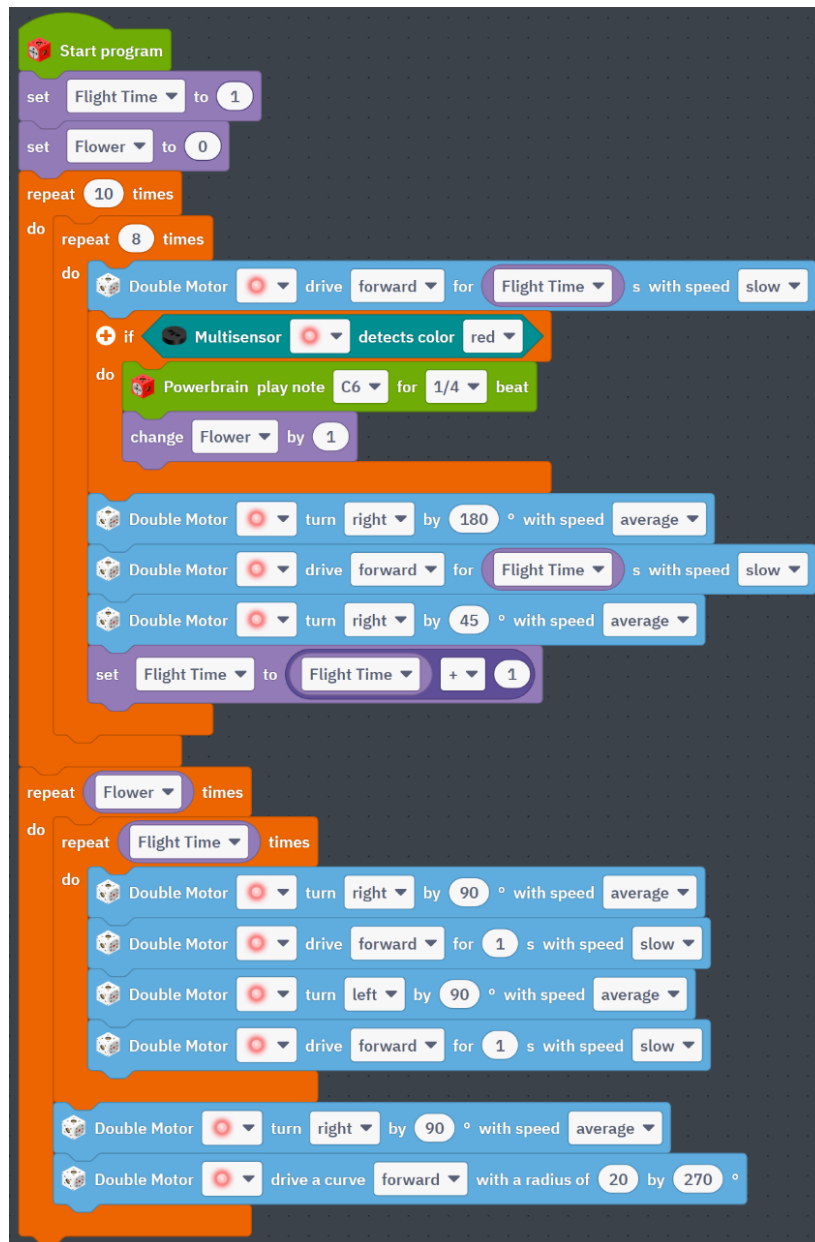
## Task 1

The bee now tells the foragers how rich the food source is, that means, how many flowers it has found. To do this, let the bee dance the zigzag pattern more often. The more flowers found, the more often the bee should repeat the zigzag pattern. Therefore, the bee must turn around and start the zigzag pattern again from the starting position.



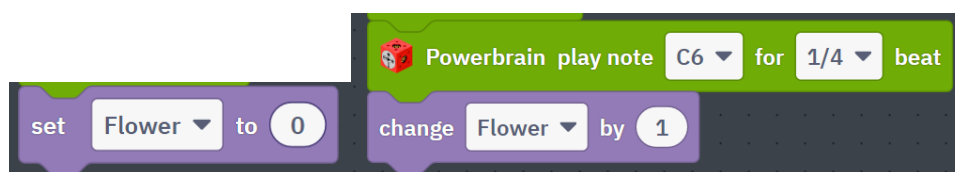


### Example solution 1



#### Tip 1

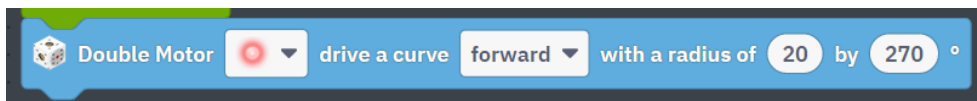
To solve the task, you need another variable which counts how many flowers the bee has found on the way. For this purpose, the program code of the forage must be changed. At the beginning a new variable *Flower* is defined and then inserted at the appropriate place after playing the melody to increase the number of found flowers. Again, the variable must count up.



### Tip 2

The students should try to optimize the code so that the robot bee always moves the jagged line and the turning circle almost exactly over each other.

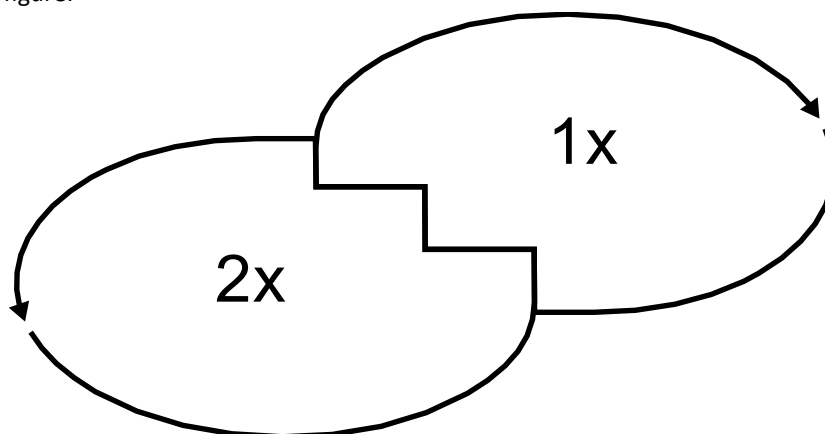
A three-quarter circle with  $270^\circ$  is very suitable for the turning circle. After the zigzag line, the bee may have to align itself again and turn by  $90^\circ$ .



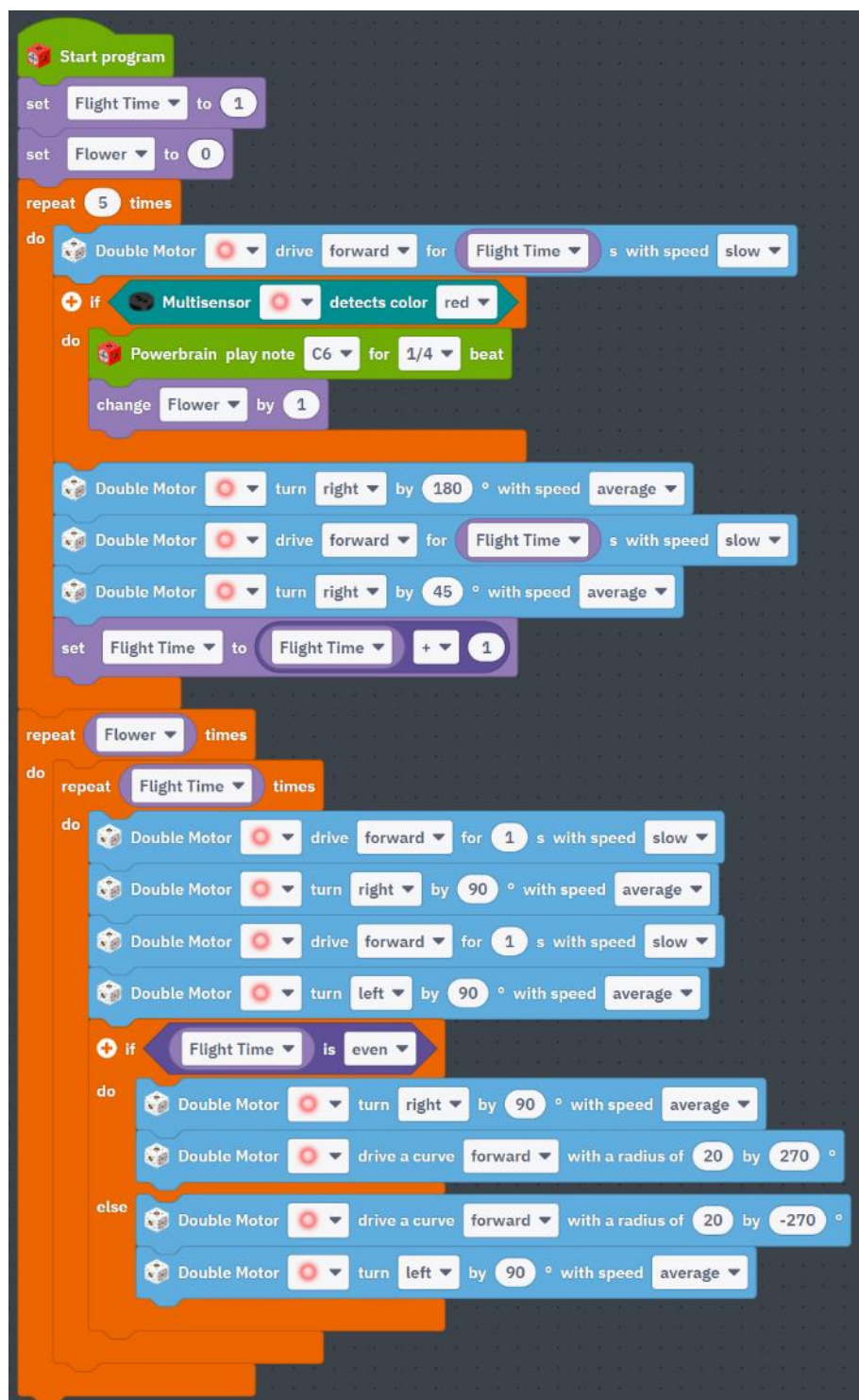
### Additional task: Turning circle

#### Additional task 1

After each zigzag line, the bee changes the direction of the turning circle, with which it returns to the starting position. Extend the code so that the robot bee dances as shown in the figure.



## Example solution additional task 1



```

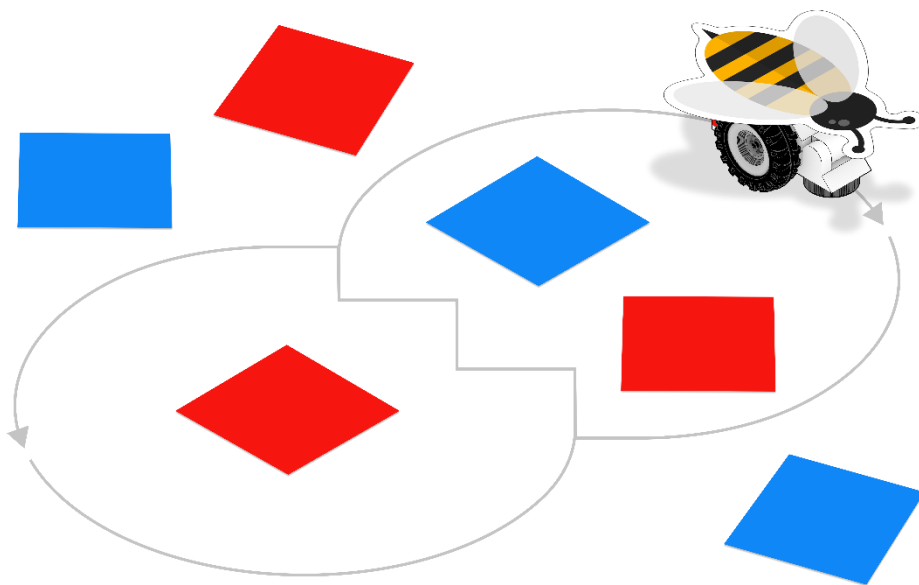
    Start program
    set Flight Time to 1
    set Flower to 0
    repeat 5 times
    do
        Double Motor drive forward for Flight Time s with speed slow
        if Multisensor detects color red
        do
            Powerbrain play note C6 for 1/4 beat
            change Flower by 1
        end
        Double Motor turn right by 180 ° with speed average
        Double Motor drive forward for Flight Time s with speed slow
        Double Motor turn right by 45 ° with speed average
        set Flight Time to Flight Time + 1
    end
    repeat Flower times
    do
        repeat Flight Time times
        do
            Double Motor drive forward for 1 s with speed slow
            Double Motor turn right by 90 ° with speed average
            Double Motor drive forward for 1 s with speed slow
            Double Motor turn left by 90 ° with speed average
        end
        if Flight Time is even
        do
            Double Motor turn right by 90 ° with speed average
            Double Motor drive a curve forward with a radius of 20 by 270 °
        end
        else
            Double Motor drive a curve forward with a radius of 20 by -270 °
            Double Motor turn left by 90 ° with speed average
        end
    end
    end
  
```

The script is a Scratch-style block-based program for a Tinkerbots robot. It begins with a 'Start program' block, followed by two 'set' blocks: 'Flight Time' to 1 and 'Flower' to 0. A large 'repeat 5 times' loop contains several 'do' blocks. The first 'do' block has a 'Double Motor' drive forward for 'Flight Time' seconds at 'slow' speed. This is followed by an 'if' block checking if the 'Multisensor' detects the color 'red'. If true, it plays a 'C6' note for 1/4 beat and increments 'Flower' by 1. Then, it turns right 180 degrees at 'average' speed, drives forward for 'Flight Time' seconds at 'slow' speed, and turns right 45 degrees at 'average' speed. Finally, it increments 'Flight Time' by 1. After the first loop, there is another 'repeat' loop where the number of iterations is the value of 'Flower'. Inside this loop, there is a 'repeat Flight Time times' loop. The inner loop contains four 'Double Motor' blocks: drive forward for 1 second at 'slow' speed, turn right 90 degrees at 'average' speed, drive forward for 1 second at 'slow' speed, and turn left 90 degrees at 'average' speed. After the inner loop, there is an 'if' block checking if 'Flight Time' is even. If true, it turns right 90 degrees at 'average' speed and drives a curve forward with a radius of 20 by 270 degrees. If false, it drives a curve forward with a radius of 20 by -270 degrees and then turns left 90 degrees at 'average' speed.

## Lesson 7 – Get to know functions

Lesson 7 starts a new topic within programming: Defining functions. Using functions is very helpful in two ways:

- They provide an overview: If a program is already very complex, you can split the program into subtasks and combine them into one function each.
- Functions save work: Functions can be used again and again in several places. This way, the same code does not have to be rewritten at a different place but can simply be called again via the function.



## Introduction

## Lesson talk

Take a look at the previous program code with the students and consider together into which different tasks the program could be divided. It helps to look at the different tasks the bee fulfills.

If the program should be divided into two parts, the search for flowers, for example, could be set as a separate task, as well as the "telling" of the bee where it found the flowers. This can now also be made clear in the program code.

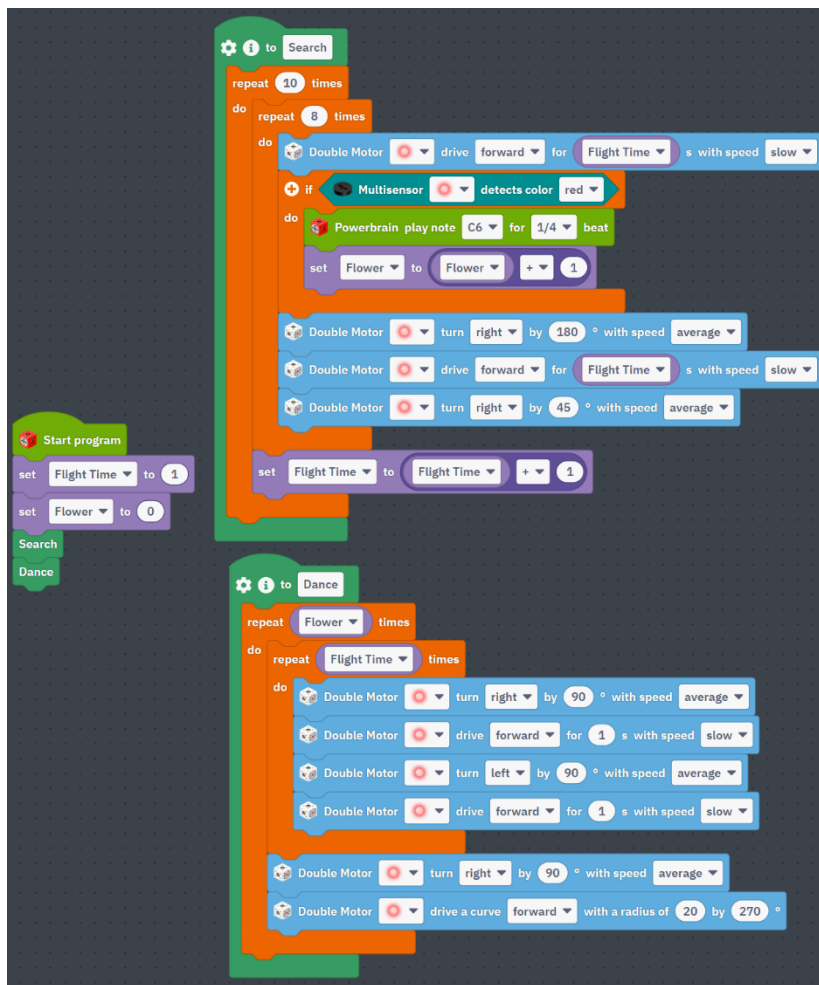
## Structure the tasks of the bee

## Task 1

Think about which two major subtasks the bee has and which blocks of your program belong to each.

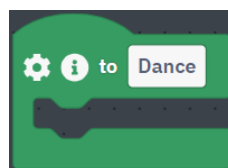
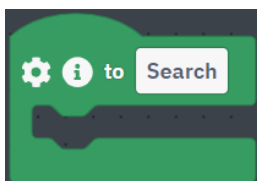
Create a new function for each of the two subtasks and move all blocks that belong to this subtask into this new function.

## Example solution 1



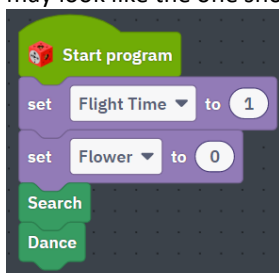
### Tip 1

You can give meaningful names to the functions as well as the variables. So, you can name the functions of the subtasks e.g., *search* and *dance*.



### Tip 2

It is important that the function is called at the right place in the program code. A function call may look like the one shown in the figure.

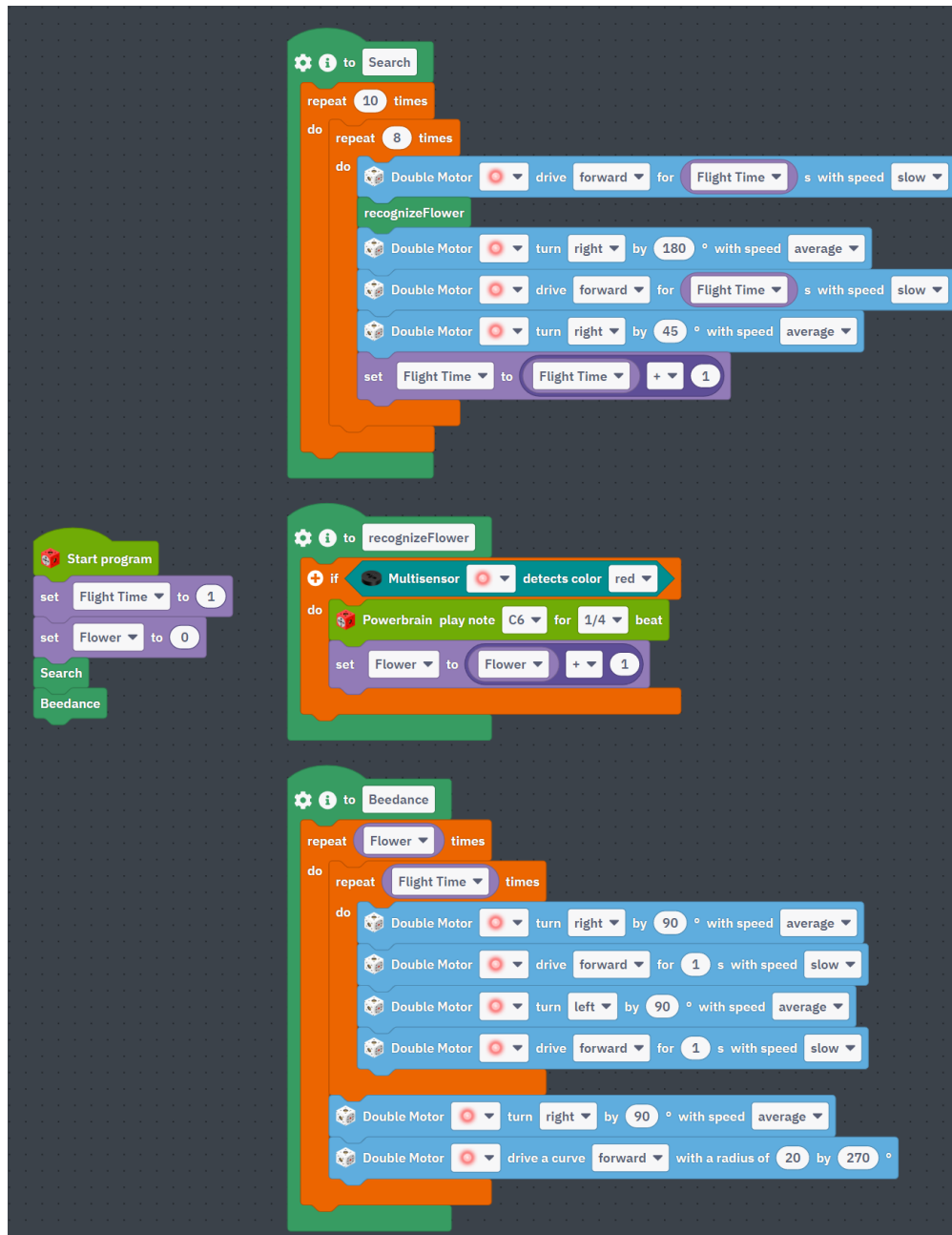


## Additional task: Swap out even more blocks to functions

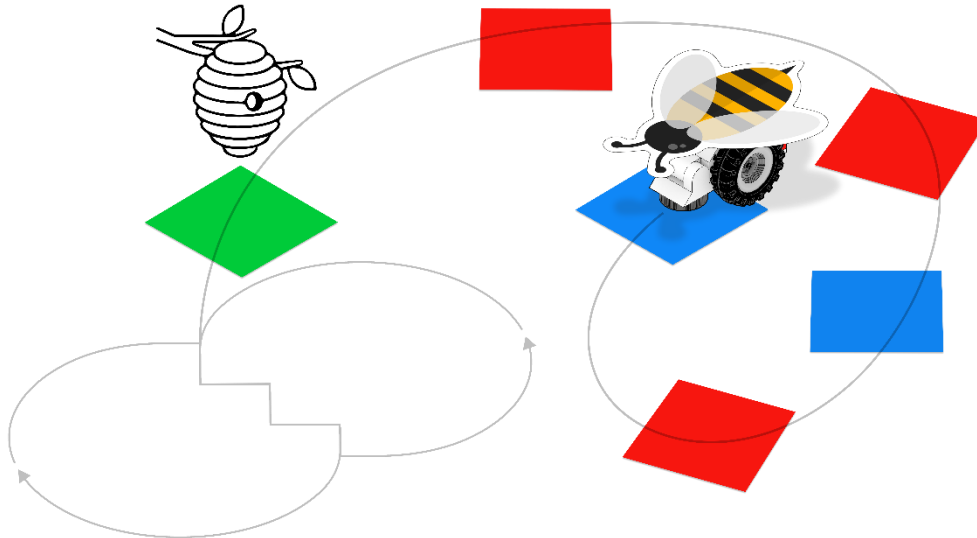
### Additional task 1

Think of another subtask and create a new function to swap out the corresponding blocks.  
Think about which division makes sense.

### Example solution additional task 1



## Lesson 8 – Adjust program code with functions



### Preparation

The students now know what variables and functions are in programming and when they are used. They can use the variables and functions to build their program.

In the next step, we refine the simulation of bee communication by another module.

This requires a little preparation. In the following task the beehive comes into play in addition to the bees and flowers. This will be symbolized by green cardboard paper.

### Return to the beehive

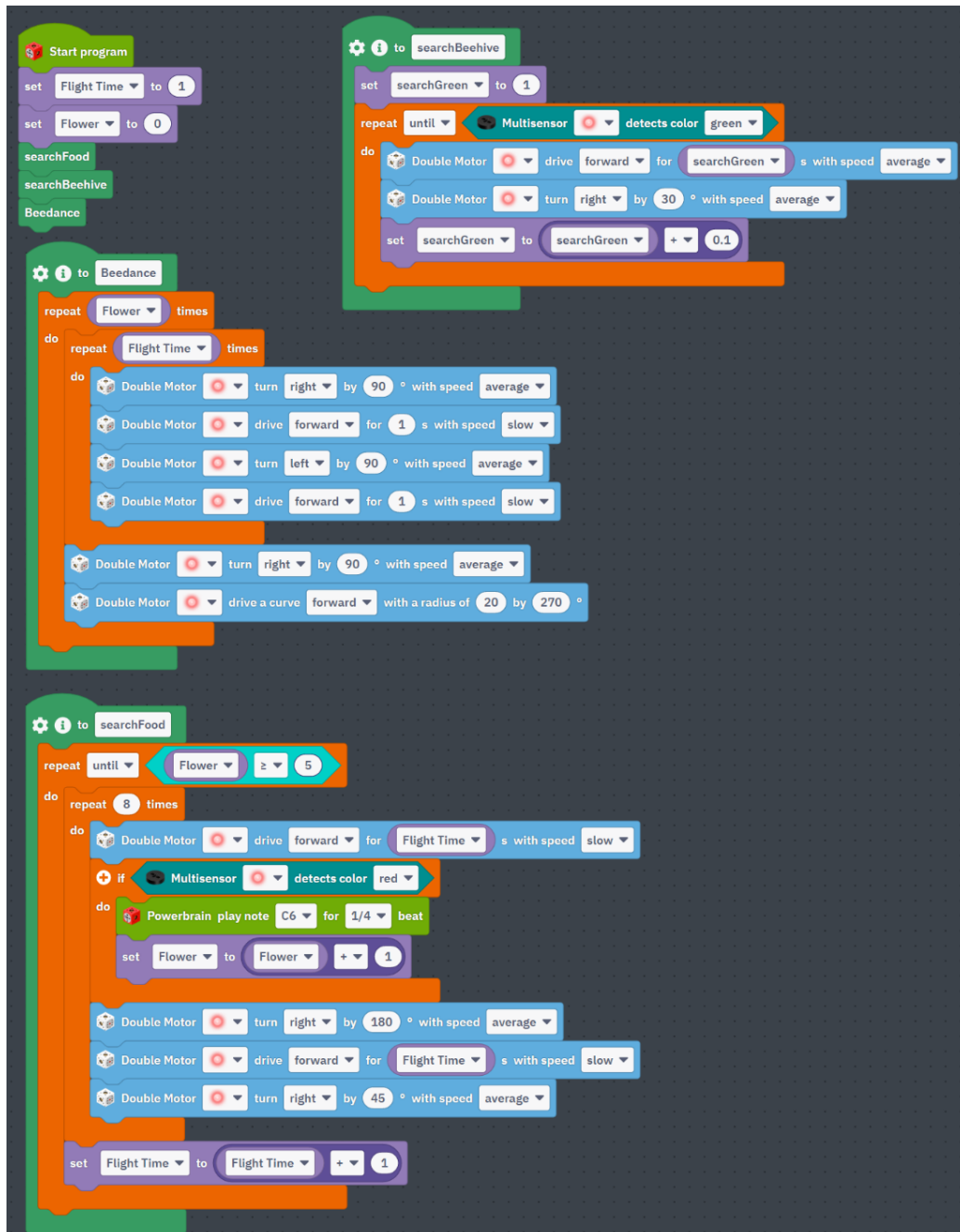
#### Task 1

The robot bee now receives the following order:

It should return to the beehive when it has found five flowers.

It recognizes the beehive by the green color. Only when it has found its hive it should perform the bee dance.

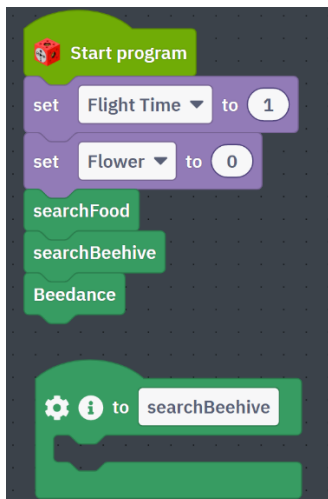
## Example solution 1



### Tip 1

Give the students the hint that a new function should be created. The placement of the function is important here. The bee should first search for the hive and only then perform the dance. The functions can be called in exactly this order.

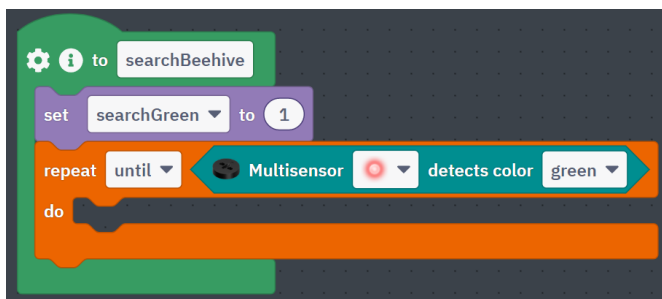




## Tip 2

Next, the new function still needs to be filled. The bee is supposed to search for the color *green* here. To do this, it again needs a search pattern and a method that it stops searching as soon as it has found the hive. The *repeat until* loop is suitable for this. It will stop as soon as the sensor detects the color green.

Also, as before, we want to change the search pattern so that it does not always search in the same place. For this purpose, you can introduce a new variable.

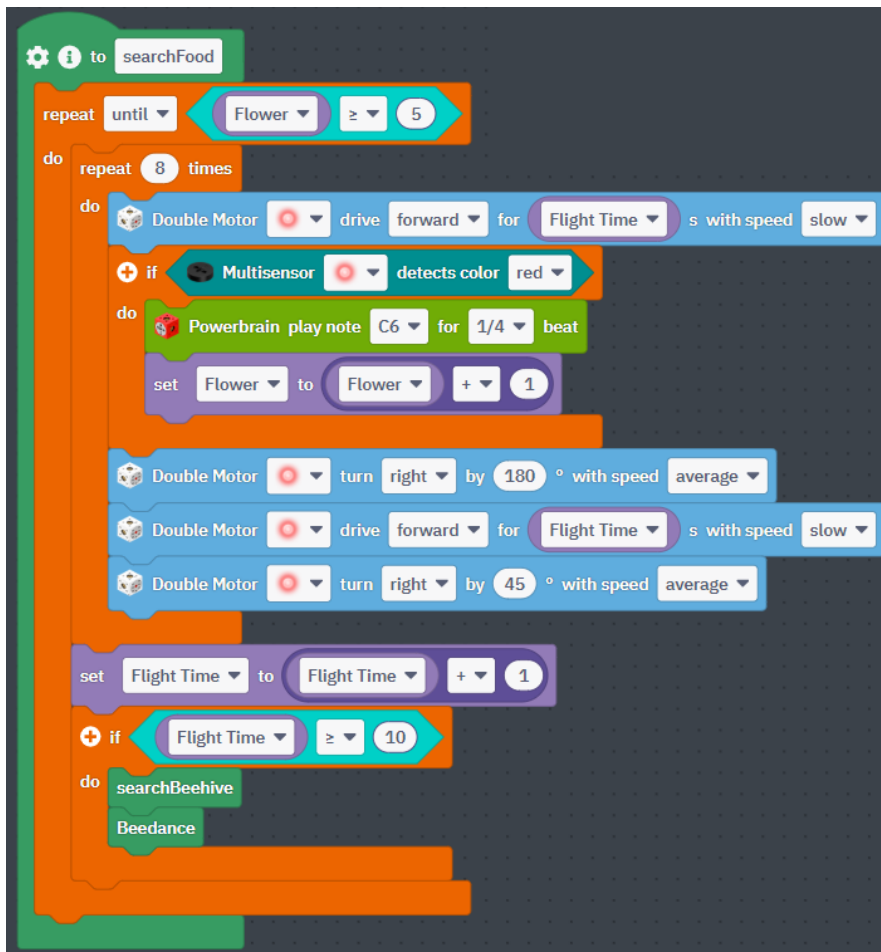


## Additional task: Unsuccessful search

### Additional task 1

There are days when the robot bee does not find enough flowers on its search. If it has not found five flowers even after a long time, it should still fly to its hive and let the other bees know how many flowers it has found.

### Example solution additional task 1



### What is the next step?

The students are now very advanced and have proven that they are able to carry out even difficult tasks.

Of course, there are many more possibilities, which can be programmed in the context with the bee:

- Use a different function to let the bee search its hive than for flower search.
- Find a method for the robot bee to make the zigzag line in the direction it came from.
- When your bee has performed its dance, let it take a short break and then start searching again. During the break, the bee could also sing a short song for us. After the break it should start counting again from the beginning.
- Build a function: Whenever the robot bee is lifted, it should complain with the police signal or give another warning tone.
- Let the robot bee perform a little dance of joy when it has found five flowers. Be sure to use the pivot module for the dance!