

Blockly Learning About Algorithms & Loops

Teaching Guide

Overview

Learning About Loops is an activity to teach about the concept of algorithms including loops and conditionals. It is taught with the assistance of Blockly, a website teaching CS concepts through block programming games. It specifically uses Blockly's "Maze" game, which is an introduction to algorithms including loops and conditionals. It starts simply, but every level is more challenging than the last.

Teaching Guide Sections

1. Setup
2. Activity Instructions
 - a. Class Discussion
 - b. Students play "Maze"
 - c. Wrap Up
3. Walkthrough / Key
4. Extra Resources
5. Optional Extension

Vocabulary

- **Algorithm** - A set of instructions that will always result in the same answer when given the same input. They are executed sequentially.

- **Conditional** - A control structure that executes one of two possible actions depending on a condition. Those two possible actions are the true or false branches.
 - **Control Structure** - Components of an algorithm / code that don't do work on their own, but organize the work of other steps. Like telling you which path to go on.
 - **If-else** - A conditional statement that basically asks "if this is true... then do this. Or else... it's not true, so do this."
 - **Loop** - A repetitive control structure that repeats actions within for as many times as you specify, or until a condition is reached.
-

Setup

Requirements:

Each student will need to use a laptop to run this activity. They will be playing on a browser, any will do. They will use the Blockly website, link below.

Activity Instructions

1. Class discussion

The teacher can pose a few of the following questions to the class, and have an open discussion, providing examples and letting students give answers of their own for a few minutes. The teacher can use a few of the provided responses to provide information and segue into the activity.

Provided Questions & Answers:

- What is an algorithm?
 - A set of instructions that will always result in the same answer when given the same input. They are executed sequentially.

- What a loop is?

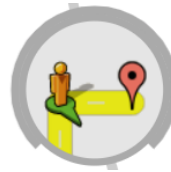
- A loop is a set of instructions that repeats as many times as you specify, like three or ten or even twenty times. It can also repeat until a condition is reached, like something specific happening that tells it to stop.
- Conditions, or conditionals, are questions you can ask yourself, then answer with “yes” or “no.” If the answer is yes, you would do one thing, like continue repeating, or if it is no, you would do something else.

- An example of a loop is washing the dishes. Have any of you ever had to wash the dishes? What are the steps?

- You could probably describe the steps as: you take some soap, scrub a dish, wash it off, then put it on the drying rack. Those would be the actions, and the loop is that you repeat them for every dish.
- But when do you stop? You’d stop when there are no more dishes left to wash. That is your end condition.
- Speaking of loops, today we are going to be practicing writing loops and conditional statements. Go ahead and open up your computers...



2. Students play Blockly’s “Maze”



Blockly “Maze” Link: <https://blockly.games/maze>

Let students open their laptops and go to the link, which will send them directly to the “Maze” game on Blockly. The online game does not have audio, so headphones are not required. It saves progress for any of its games, if opened in the same browser, so if the site is refreshed or closed, students will not have to restart. There are ten problems in the “Maze” game.

If students end up going to the Blockly home page, have them click the “Maze” icon, which is the second from the left. Above is a picture of the icon.

Refer to the Walkthrough / Key section below for specific instructions per section.

3. Wrap up

Have another short conversation with the students, in the same manner as the first class discussion. Here, teachers will ask questions about what the students learned, how they felt about the activity, etc. They will also suggest that when students have free time, they can look at the other activities on the site that they didn't get a chance to look at today, if they want.

Provided Questions & Answers:


- So, how was it? Did you guys think it was hard?
 - Loops are a difficult topic in computer science that can get really complicated. You guys only had one loop, but a lot of times programmers put loops inside other loops! That's called nested loops.
- What did you learn about algorithms?
 - Algorithms are very precise. It takes a great deal of thought to get them correct.
- What did you learn about loops?
 - You can put a lot inside of loops! You can put conditions, regular actions, or even other loops.
 - Loops can end whenever you want them to. In our case, we had the loop end when the person reached their goal on the map.
- What did you learn about conditionals?
 - Conditional statements like the one you used, if-else, are really important in programming. They let you tell your code what to do in different situations, which is really important.

- Let's say you told someone to step on bugs inside, to kill them. That's fine, right? But what if the bug was a wasp? You don't want to step on those! So you would have to give them a condition... If it's not a wasp, step on bugs inside to kill them. And you might also have other conditions too, like if it is a ladybug, pick it up and let it go outside.


Walkthrough / Key

Question 1:

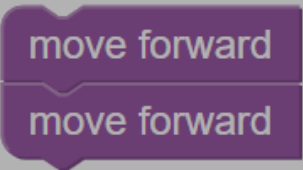
Instructions...



Stack a couple of 'move forward' blocks together to help me reach the goal.



Q1 Answer...



Congratulations!

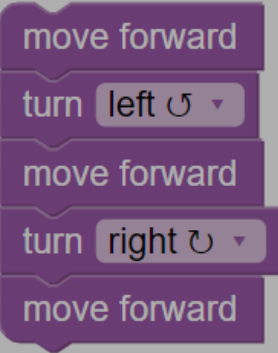
You solved this level with 2 lines of JavaScript:

```
moveForward();  
moveForward();
```

Question 2:

Instructions... same as Question 1, but with left and right turns now.

Q2 Answer...



move forward
turn left ↶
move forward
turn right ↷
move forward

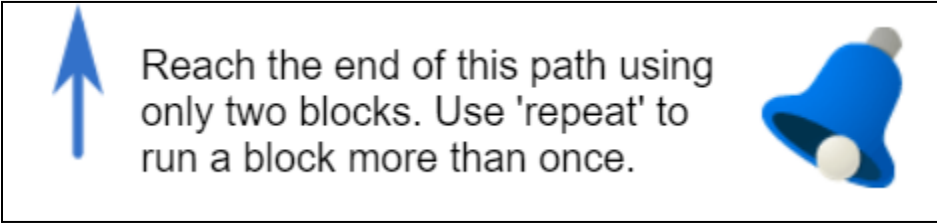
Congratulations!



You solved this level with 5 lines of JavaScript:

```
moveForward();  
turnLeft();  
moveForward();  
turnRight();  
moveForward();
```

Question 3:

Instructions...



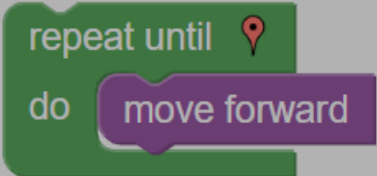
 Reach the end of this path using only two blocks. Use 'repeat' to run a block more than once. 

Note... If there are already 2 blocks in play, none can be added until one is dragged off to the side. The blocks on the side will look like this when no more can be added:



turn left ↶
turn right ↷
repeat until 
do

Q3 Answer...




Congratulations!

You solved this level with 3 lines of JavaScript:


```
while (notDone()) {  
  moveForward();  
}
```

Question 4:

Instructions...



You can fit more than one block inside a 'repeat' block.



Q4 Answer...



Congratulations!

You solved this level with 6 lines of JavaScript:

```
while (notDone()) {  
  moveForward();  
  turnLeft();  
  moveForward();  
  turnRight();  
}
```

Are you ready for level 5?

Question 5:

Instructions... same as Question 4.

Note... In the top right, there is a person icon which can change the appearance.



Q5 Answer...



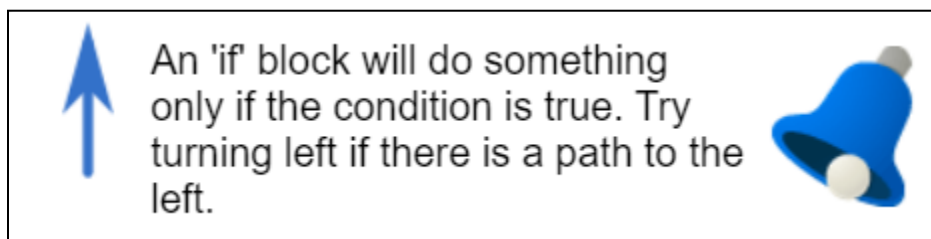
The screenshot shows a code editor with a block-based code editor on the left and a JavaScript code block on the right. The block-based code consists of two "move forward" blocks, a "turn left" block, a "repeat until" block with a location pin icon, and a "do" block containing a "move forward" block. The JavaScript code block contains the following code:

```
moveForward();
moveForward();
turnLeft();
while (notDone()) {
  moveForward();
}
```

Below the code, the text reads: "Congratulations! You solved this level with 6 lines of JavaScript:"

Question 6:

Instructions...



Q6 Answer...




Congratulations!

You solved this level with 6 lines of JavaScript:


```
while (notDone()) {  
  moveForward();  
  if (isPathLeft()) {  
    turnLeft();  
  }  
}
```

Question 7:

Instructions...



Click on ahead ▾ in the 'if' block to change its condition.



Q7 Answer...



Congratulations!

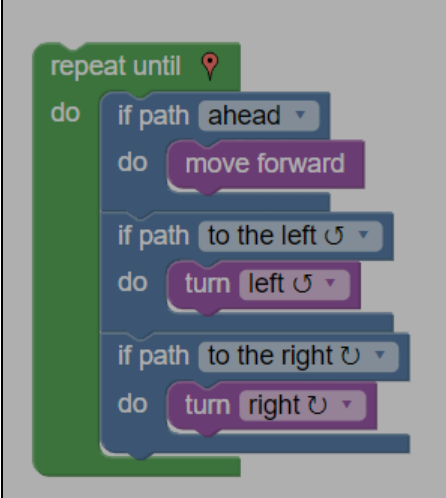
You solved this level with 8 lines of JavaScript:

```
while (notDone()) {  
  if (isPathRight()) {  
    turnRight();  
  }  
  if (isPathForward()) {  
    moveForward();  
  }  
}
```

Question 8:

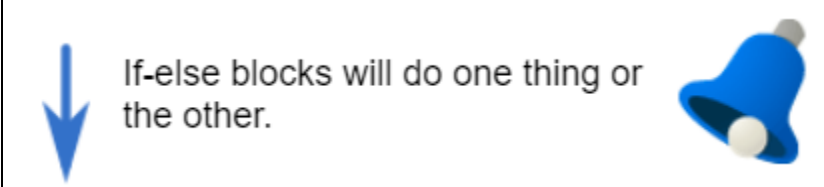
Instructions... same as Question 7.

Q8 Answer...

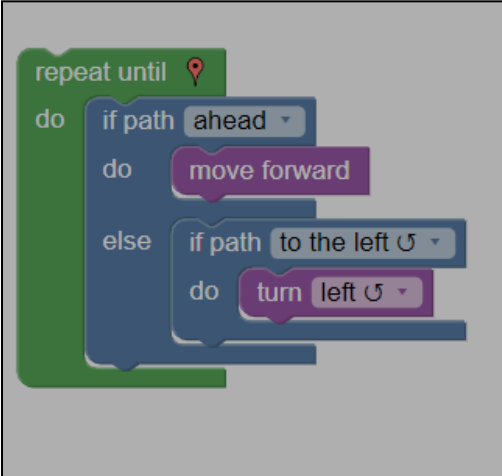
	<p>Congratulations!</p> <p>You solved this level with 11 lines of JavaScript:</p> <pre data-bbox="699 527 1062 867">while (notDone()) { if (isPathForward()) { moveForward(); } if (isPathLeft()) { turnLeft(); } if (isPathRight()) { turnRight(); } }</pre>
---	---

Question 9:

Instructions...



--

Q9 Answer...

	<p>Congratulations!</p> <p>You solved this level with 9 lines of JavaScript:</p> <pre data-bbox="760 1583 1133 1892">while (notDone()) { if (isPathForward()) { moveForward(); } else { if (isPathLeft()) { turnLeft(); } } }</pre>
---	---

Question 10:

Instructions...



Can you solve this complicated maze? Try following the left-hand wall. Advanced programmers only!

OK

Answer... There is more than one way to do this. Here are some ways:

Going to the right first...

```
move forward
move forward
turn left 90
repeat until
do
  if path to the right 90
  do
    turn right 90
    move forward
  else
    if path ahead
    do
      move forward
    else
      turn left 90
```

```
repeat until
do
  if path ahead
  do
    move forward
  if path to the right 90
  do
    turn right 90
    move forward
  else
    if path ahead
    do
      move forward
    else
      turn left 90
```

Going to the left first...

```
move forward
move forward
repeat until
do
  if path to the left 90
  do
    turn left 90
    move forward
  else
    if path ahead
    do
      move forward
    else
      turn right 90
```

END OF ACTIVITY

Extra Resources

Unplugged activity about For Loops:

<https://studio.code.org/s/coursef-2021/lessons/13#activity-75728>

Lesson Plan about For Loops:

<https://www.flocabulary.com/lesson/coding-for-loops/>













Optional Extension

There are other games on Blockly's website that you can let students play through to learn more about different programming topics. I most recommend letting them play the "Bird" game, as it is more in-depth about conditionals, or the "Turtle" game, as it is more in-depth about loops. Puzzle is also acceptable, but it is unrelated to conditionals or loops:



Puzzle

- A quick introduction to Blockly's shapes and how the pieces snap together.
- It is essentially a puzzle made up of animals, their characteristics, and number of legs. Below is the answer:

<p>Bee</p> <p>picture: </p> <p>legs: 6</p> <p>traits:  </p>	<p>Cat</p> <p>picture: </p> <p>legs: 4</p> <p>traits:  </p>	<p>Snail</p> <p>picture: </p> <p>legs: 0</p> <p>traits:  </p>	<p>Duck</p> <p>picture: </p> <p>legs: 2</p> <p>traits:  </p>
--	--	---	---



Bird

- A deep-dive into conditionals. Control-flow is explored with increasing complexity.



- Taught by moving a bird to collect a worm, then go to its nest with directional angles.



Turtle

- A deep-dive into loops. Use nested loops to paint a picture.
- Levels get extremely hard, and build upon code used in previous levels.
- Requires a bit of math (how many degrees to turn).

I would not recommend letting them go into the following games, as they are more complex, or just unrelated to the current activity, and as there is no teaching guide attached, it will be difficult to help them:

- **Movie**

- An introduction to mathematical equations. Use math to animate a movie.



Music

- An introduction to functions. Use functions to compose music.



Pond Tutor

- Introduces text-based programming. Levels switch back and forth between blocks and actual JavaScript in a text editor.



Pond

- An open-ended contest to program the smartest duck. Use either blocks or JavaScript.
- Must do “Pond Tutor” before doing this.