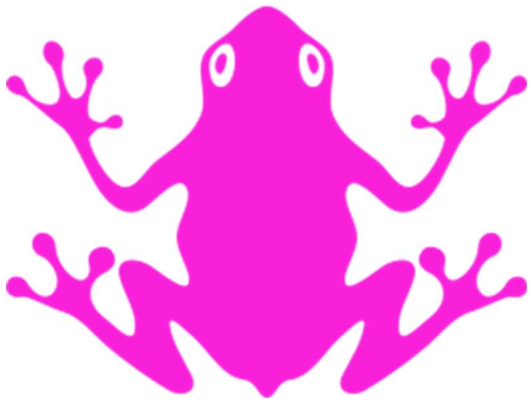
2 What's an algorithm?

An algorithm is essentially a set of instructions to be performed one after another. When we are writing a computer program, we need to specify all the steps of the algorithm in advance of running it. Conveniently, we don’t need to have the complete algorithm from the beginning because we can normally start, stop, and reset the program in a straightforward way.

2.1 Programmable frog

To explore what an algorithm is we can think about a simple robot frog that accepts four commands:



- ↺ Turn left 90 degrees
- ↻ Turn right 90 degrees
- ↑ Move forward one space
- ↓ Move backwards one space

2.1.1 Move forward and backward

If you wanted to make the frog advance forwards then return to the same spot you could issue the commands:

↑ Move forward one space, ↓ Move backwards one space

2.1.2 Spinning around

If you wanted to make the frog spin around on the spot you could issue the commands:

↺Turn left 90 degrees, ↺Turn left 90 degrees,

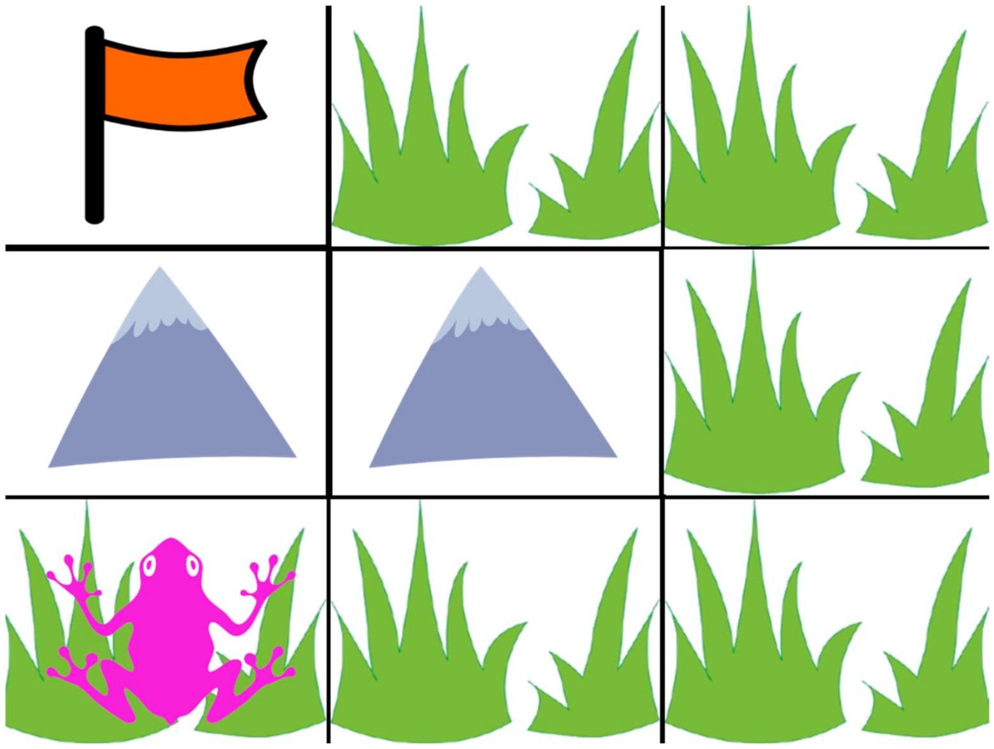
↺Turn left 90 degrees, ↺Turn left 90 degrees

OR

↻Turn right 90 degrees, ↻Turn right 90 degrees,

↻Turn right 90 degrees, ↻Turn right 90 degrees

Exercise



Consider how the frog could move around the maze using the four commands available. You may find it ***useful to move a key or coin*** *(something with a clear direction) around the grid to test your commands out.*

Assume the frog could travel over grass or mountain squares. What commands will move *the frog onto the flag?*

We learn that if the frog tries to move over a mountain square, it will get stuck. What commands will move *the frog onto the flag while only travelling via grass squares?*

The frog is now broken and can only move backwards. Is there a set of commands that will get the frog to the flag without the "move forward one space" command available?

2.2 Giving instructions

Exercise

Follow these instructions:

- *Draw a square*
- *Draw a triangle on top of the square*

Check the image on the last page of this section. Does it match what you have drawn above? If not, think about how the instructions could be improved to get the desired result?

2.2.1 Algorithms must be precise

Precise sets of ordered steps that can be followed by a human or a computer to do a task are known as algorithms. There are many ways that algorithms can be represented, for example annotated drawings, flowcharts, or listed instructions. Getting the desired result from an algorithm often requires precise instructions.

2.2.2 Creating algorithms to guide programming

It can be a valuable exercise to write instructions before turning them into code, particularly in an unfamiliar programming environment. Your steps do not need to match exactly, but they should have a close resemblance to what you want to achieve. Try to focus on key details and do not be afraid to draw pictures or flowcharts as well.

Exercise

Write an algorithm for something that you do on a day-to-day basis. ***For example****: your workday morning routine or instructions to reach your house from a common landmark.*

Can you think of any ways to improve your algorithm? Perhaps using a diagram or adding more precise instructions? Consider showing somebody else and write down any ideas you come up with.

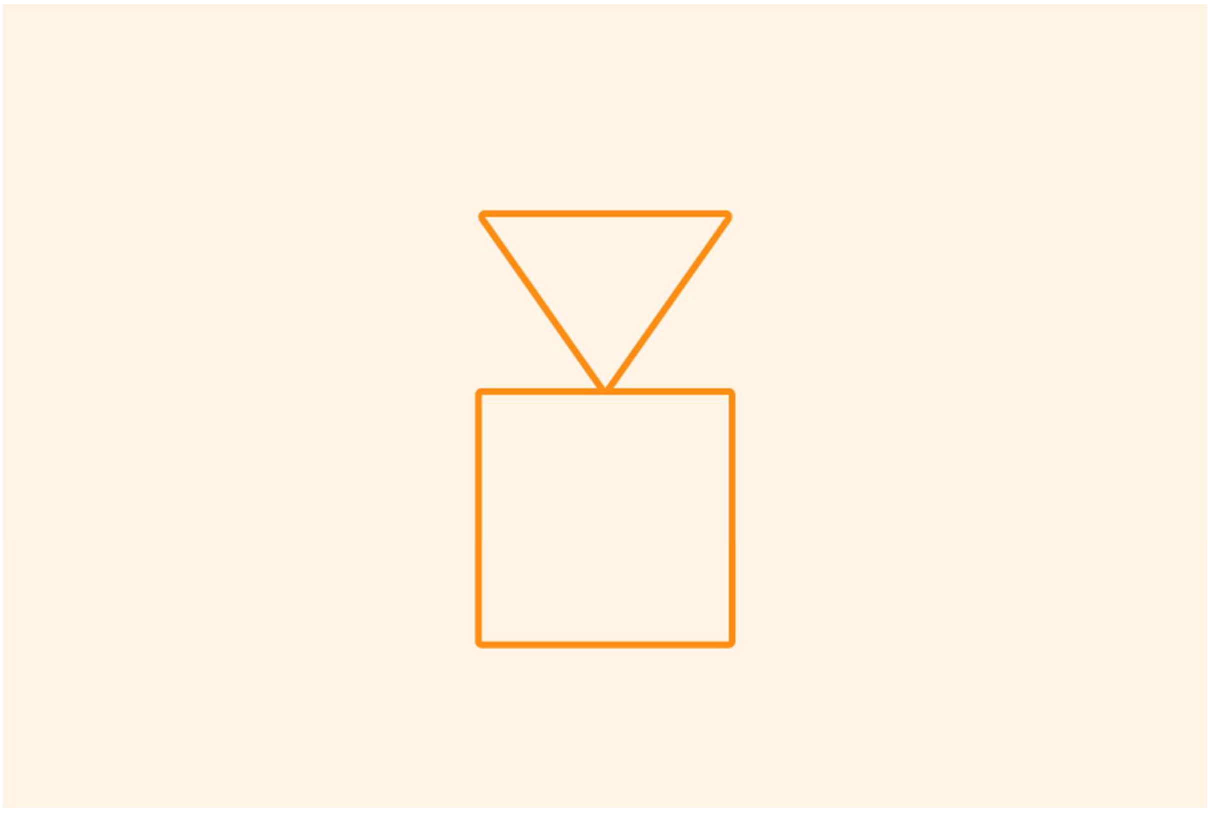


Figure 1. Solution to "2.2 Giving instructions" exercise