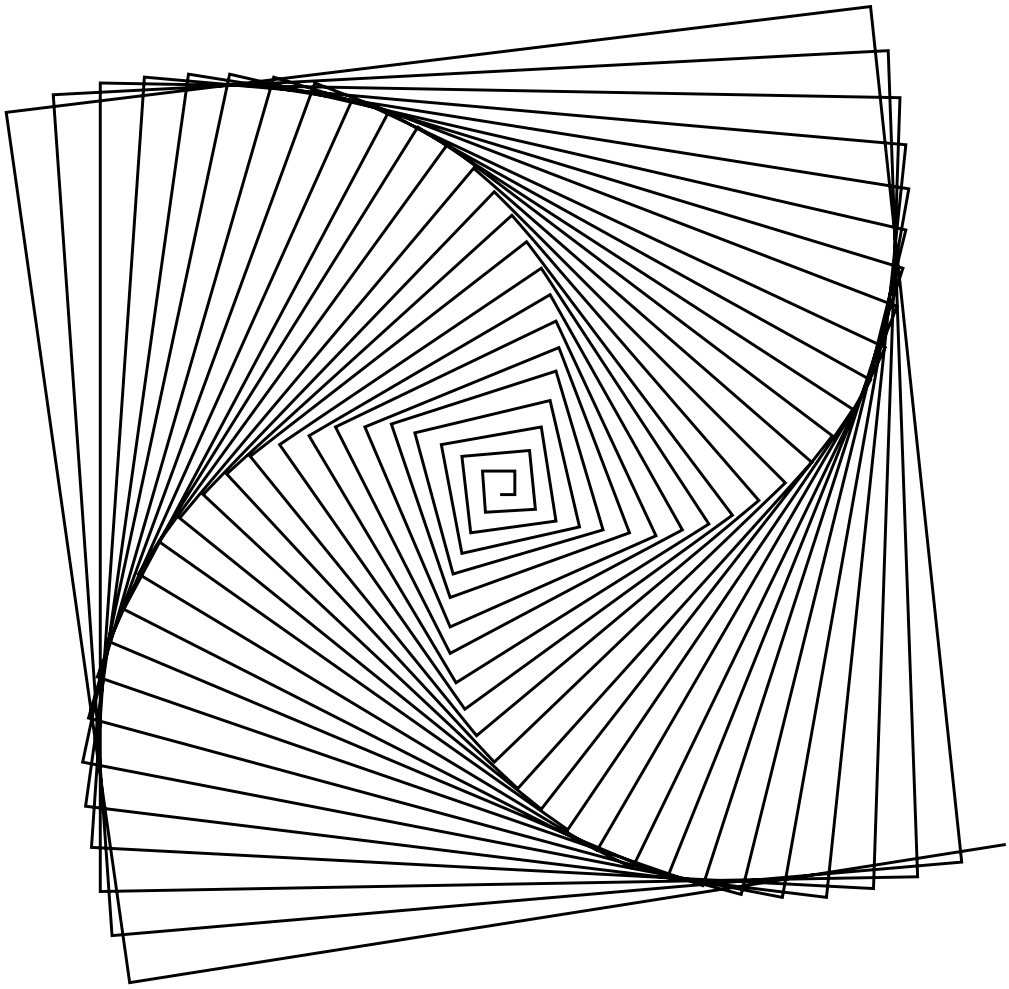
1

Loops and Variables



In this chapter we present how variables and loops can used together. We start by analyzing a simple problem that shows the need of using variables and loops together. We then experiment with some other problems. We introduce the new messages to:do: and to:do:by: that are more adapted to situations where sequences of numbers are needed.

1 A Motivating Example

Try to generate the strange stair shown in Figure 1.1. Start for example by generating a normal stair and modify it. One of the problem that you should be facing is that the length of each step is growing regularly.

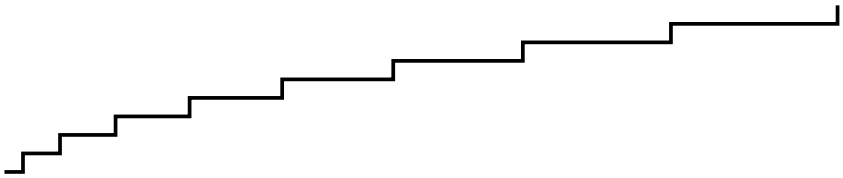


Figure 1.1: A strange stair

One of the simplest solution is described by the Script 1.1 where the length of a step grows 10 by 10 pixels. However, such a solution is not satisfactory because we have to compute manually the length of the next step and that we have to repeat stupidly the same sequence of messages.

Script 1.1 (*Strange stair)*

```
| pica |
pica := Bot new.
pica go: 10.
pica turnLeft: 90.
pica go: 5.
pica turnRight: 90.
pica go: 20.
pica turnLeft: 90.
pica go: 5.
pica turnRight: 90.
pica go: 30.
pica turnLeft: 90.
pica go: 5.
pica turnRight: 90.
pica go: 40.
pica turnLeft: 90.
pica go: 5.
pica turnRight: 90.
...
```

We would like to be able to use the power of variables and combined it with the one of loops. First avoiding to repeat the sequence of messages can be possible using the timesRepeat: method. Second when we analyze the Script 1.1 we see that the length of a step is the length of the preceding one plus 10. Here, 20 = 10 + 10, 30 = 20 + 10, 40 = 30 + 10,.... If we use the variable length to represent the length of a step, we see that the length of the second step is the length of the first step plus 10, that the length of the third step is the length of the second step plus 10... We can express this by the following expression length := length + 10 which increases the value of the variable length by 10.

Let's combine everything! Now if we take the script of a normal stair (Script 1.2), introduce the variable length we obtain a similar method and drawing (Script 1.3). Finally, if we change length

value during each step of the loop, we obtain the stair we want (Script 1.4).

Script 1.2 (*A stair with normal steps)*

```
| pica |
pica := Bot new.
10 timesRepeat: [pica go: 10.
        pica turnLeft: 90.
        pica go: 5.
        pica turnRight: 90]
```

Script 1.3 (*A stair with normal steps using length)*

```
| pica length |
pica := Bot new.
length := 10.
10 timesRepeat: [pica go: length.
        pica turnLeft: 90.
        pica go: 5.
        pica turnRight: 90]
```

Script 1.4 (*The Solution)*

```
| pica length |
pica := Turtle new.
length := 10.
10 timesRepeat: [pica go: length.
        pica turnLeft: 90.
        pica go: 5.
        pica turnRight: 90.
        length := length + 10 ]
```

In the Script 1.4, the sequence of messages in the loop, first make the robot go forward of a distance that is given by the value of the variable length, the first time 10, then the robot turns and draws the missing part of the stair, finally the value of the variable length is increased by 10 and the loop restarts but with the variable length having a value of 20. The same process is repeated 10 times. Note that you should not forget the expression length := length + 10 else the value of the variable would never change.

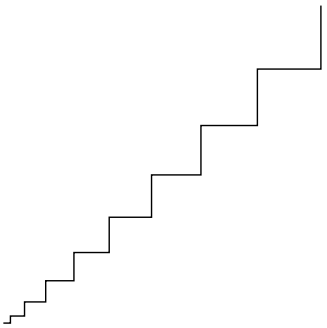
Experiment 1.1
Change the last line of the loop for example put length := length + 15. Change also the place where such a line occurs in the loop. Can you explain what happen when you move the last line of the loop to the beginning of the loop?

If you need more time to understand the solution proposed in the Script 1.4, we suggest you to look carefully at the value of the variable variable length in particular at the beginning and the end of the loop. You can also to simulate the messages sent. If necessary read again the Chapters **??**.

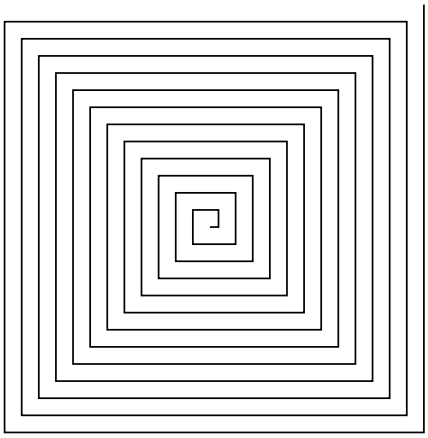
2 Practicing: Mazes, Spirals and Other

Now let us see how combining variables and loops helps us to solve some other problems.

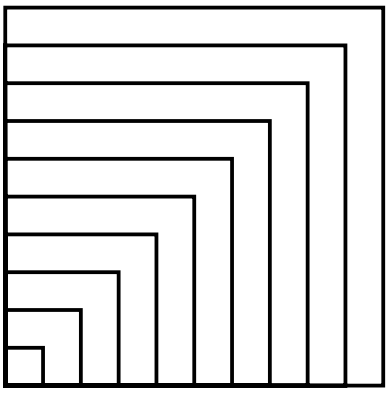
Experiment 1.2
Change the Script 1.4 to produce the picture shown on the right.



Experiment 1.3 (*Maze)*
Define a script that reproduces the drawing shown on the right. By turning from a different angle you should be able to recreate the first picture of the chapter as well as the spiral shown in the Figure 1.2.



Experiment 1.4 (*Russian squares)*
Build the squares of different sizes as shown on figure on the right.



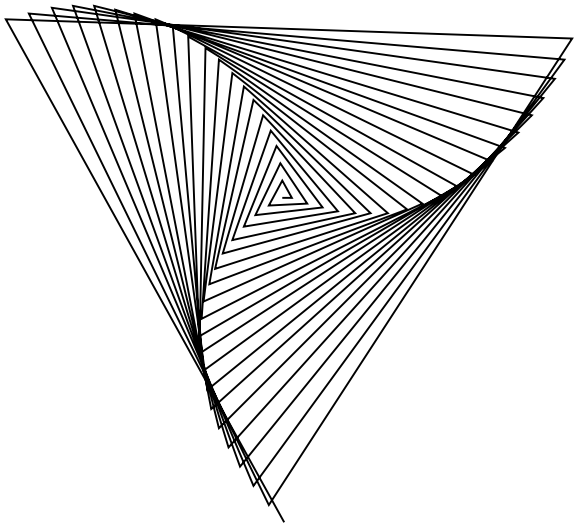
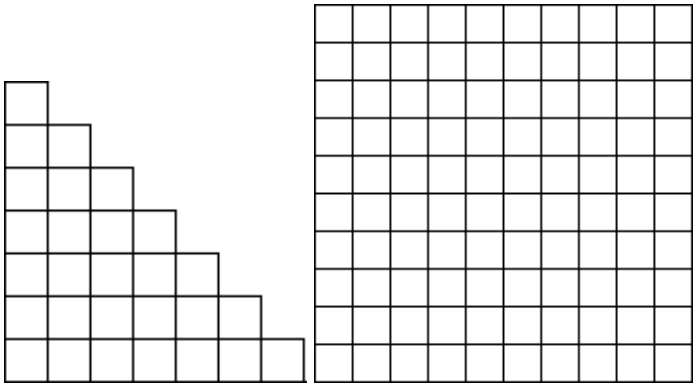


Figure 1.2: A nice spiral

Experiment 1.5 (*Spirale)*
Define two scripts that create the constructions shown on the right. Note that this task is a bit difficult and that there are different ways to solve it.



Experiment 1.6 (*Corridor)*
Build the squares of different sizes as shown on figure on the right.