PAYS 2023

INTRODUCTION TO PROGRAMMING USING PYTHON

4: Turtle and for loops

Alexandra Papoutsaki

she/her/hers

Lecture 4: Turtle and for loops

- Turtle module
- For loops

Modules

- Module: a collection of functions and variables.
- Modules allow us to use code that other people have written.
- For example, there is a module called `math` that has many of the math functions you might want.
- We can look at the documentation for this module online by searching for "math python" or by going to https://docs.python.org/3/ and browsing searching there.
  - https://docs.python.org/3/library/math.html
    - logs
    - sqrt
    - trigonometric functions
    - constants

Importing modules

- If we want to use a module, we need to tell the program to include it with our program. To do this, we need to "import" it.
- There are many ways of importing modules (some better than others).
- For now, we're going to import the functions and variables into our program as if they were local (i.e. just as if we'd written them in our program).
  - this is convenient for now, but in some situations there are better ways of doing it (more on this later)

```
>>> from math import *
```

- This statement has multiple components:
- `from` is a keyword,
- `math` is the name of the module,
- `import` loads the module into our program,
- * means everything, i.e. load everything included in the math module.

`turtle` module

- The turtle module implements a set of commands similar to the Logo programming language
- The basic idea is that you control the movements of a turtle (in our case, it will be an arrow) through basic commands such as:
  - `forward(distance)`: Move the turtle forward by the specified distance, in the direction the turtle is headed.
  - `backward(distance)`: Move the turtle backward by distance, opposite to the direction the turtle is headed. Do not change the turtle's heading.
  - `right(angle)`: Turn turtle right by angle units.
  - `left(angle)`: Turn turtle left by angle units.
  - ...and many others
- As the turtle moves, it draws a line behind it, so by giving it different commands, we can draw things on the screen!
- Check the documentation for the turtle class online
- You'll be getting more comfortable with this documentation as part of next week's lab.

`turtle` module

- The turtle module implements a set of commands similar to the Logo programming language
- The basic idea is that you control the movements of a turtle (in our case, it will be an arrow) through basic commands such as:
  - `forward(distance)`: Move the turtle forward by the specified distance, in the direction the turtle is headed.
  - `backward(distance)`: Move the turtle backward by distance, opposite to the direction the turtle is headed. Do not change the turtle's heading.
  - `right(angle)`: Turn turtle right by angle units.
  - `left(angle)`: Turn turtle left by angle units.
  - ...and many others
- As the turtle moves, it draws a line behind it, so by giving it different commands, we can draw things on the screen!
- Check the documentation for the turtle class online
- You'll be getting more comfortable with this documentation as part of next week's lab.

Let’s move our turtle!

- How would you create a square?
- `forward(some_length)`

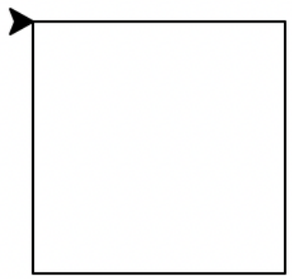
```
right(90)
forward(some_length)
right(90)
forward(some_length)
right(90)
forward(some_length)
```

Let's move our turtle!

turtle-examples.py

```
1   from turtle import *
2   from random import randint
3
4
5   def square(length):
6       forward(length)
7       right(90)
8       forward(length)
9       right(90)
10      forward(length)
11      right(90)
12      forward(length)
13      right(90)
```

```
>>> square(100)
```



- This seems like a lot of repetitive typing. Let's say we can tell the turtle to repeat some statements, would there be a better way of creating a square?
- go forward some length and then turn right, repeat this 4 times

Lecture 4: Turtle and for loops

- Turtle module
- For loops

Python for loops

- Python has a number of different "loop" structures that allow us to do repetition (computers are really good at doing repetitive tasks!)
- The `for` loop is one way of doing this
- There are a number of ways we can use the for loop, but for now the basic structure we'll use is:

```
for some_variable in range(num_iterations):

    statement1

    statement2

    ...
```

Python for loops syntaxes

```
for some_variable in range(num_iterations):
    statement1
    statement2
    …
```

- `for` is a keyword
- `in` is a keyword
- `range` is a function that we'll use to tell Python how many repetitions we want
- `num_iterations` is the number of iterations that we want the loop to do
- `some_variable` is a local variable whose scope (where it can be referred to) is only within the for loop
  - `some_variable` will take on the values from `0` to `num_iterations-1` as each iteration of the loop occurs
    - We’re computer scientists so we start counting at zero :)
  - for example, in the first iteration, it will be 0, the second time 1, the third time 2, etc.we're computer scientists so we start counting at zero :)
- Don't forget the ':' at the end!
- Like with defining functions, Python uses indenting to tell which statements belong in the for loop

What would this code do?

```
>>> for i in range(10):
...     print(i)
... 
```

```
0
1
2
3
4
5
6
7
8
9
```

An iterative square

turtle-examples.py

```
16  def iterative_square(length):
17      for i in range(4):
18          forward(length)
19          right(90)
20
```

Resources

- Textbook: Continue reading Chapter 4.
- turtle-examples.txt