7.5. Lists and `for` loops

It is also possible to perform **list traversal** using iteration by item. A list is a sequence of items, so the `for` loop iterates over each item in the list automatically.

Save & Run 4/30/2021, 8:37:29 PM - 2 of 2 Show in CodeLens

```
1 fruits = ["apple", "orange", "banana", "cherry"]
2
3 for afruit in fruits:     # by item
4     print(afruit)
5
```

```
apple
orange
banana
cherry
```

Activity: 1 -- ActiveCode (ac6_5_1)

It almost reads like natural language: For (every) fruit in (the list of) fruits, print (the name of the) fruit.

7.5.1. Using the range Function to Generate a Sequence to Iterate Over

We are now in a position to understand the inner workings we glossed over previously when we first introduced repeated execution with a for loop. Here was the example:

Save & Run 4/30/2021, 8:37:32 PM - 2 of 2 Show in CodeLens

```
1 print("This will execute first")
2
3 for _ in range(3):
4     print("This line will execute three times")
5     print("This line will also execute three times")
6
7 print("Now we are outside of the for loop!")
8
```

```
This will execute first
This line will execute three times
This line will also execute three times
This line will execute three times
This line will also execute three times
This line will execute three times
This line will also execute three times
Now we are outside of the for loop!
```

Activity: 2 -- ActiveCode (ac_6_5_1a)

The `range` function takes an integer n as input and returns a sequence of numbers, starting at 0 and going up to but not including n. Thus, instead of `range(3)`, we could have written `[0, 1, 2]`.

The loop variable `_` is bound to 0 the first time lines 4 and 5 execute. The next time, `_` is bound to 1. Third time, it is bound to 2. `_` is a strange name for a variable but if you look carefully at the rules about variable names, it is a legal name. By convention, we use the `_` as our loop variable when we don't intend to ever refer to the loop variable. That is, we are just trying to repeat the code block some number of times (once for each item in a sequence), but we are not going to do anything with the particular items. `_` will be bound to a different item each time, but we won't ever refer to those particular items in the code.

By contrast, notice that in the previous activecode window, the loop variable is `afruit`. In that for loop, we *do* refer to each item, with `print(afruit)`.

7.5.2. Iteration Simplifies our Turtle Program

Remember the turtle drawings we made earlier? Let's look again at how we can use for loops there!

To draw a square we'd like to do the same thing four times — move the turtle forward some distance and turn 90 degrees. We previously used 8 lines of Python code to have alex draw the four sides of a square. This next program does exactly the same thing but, with the help of the for statement, uses just three lines (not including the setup code). Remember that the for statement will repeat the `forward` and `left` four times, one time for each value in the list.

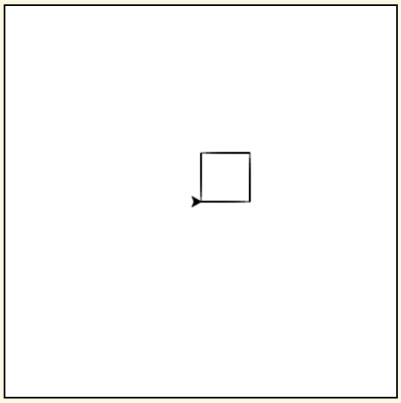
Save & Run 4/30/2021, 8:37:37 PM - 2 of 2

```
import turtle            # set up alex
wn = turtle.Screen()
alex = turtle.Turtle()

for i in [0, 1, 2, 3]:      # repeat four times
    alex.forward(50)
    alex.left(90)

wn.exitonclick()

```



Activity: 3 -- ActiveCode (ac6_5_2)

While “saving some lines of code” might be convenient, it is not the big deal here. What is much more important is that we’ve found a “repeating pattern” of statements, and we reorganized our program to repeat the pattern.

The values [0,1,2,3] were provided to make the loop body execute 4 times. We could have used any four values. For example, consider the following program.

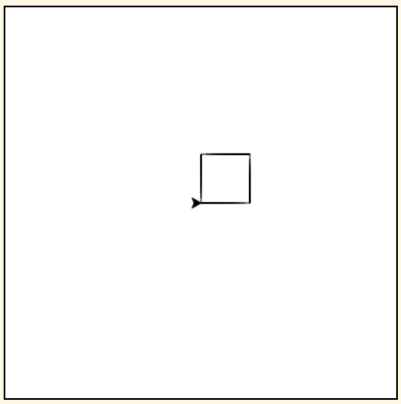
Save & Run 4/30/2021, 8:37:41 PM - 2 of 2

```
import turtle            # set up alex
wn = turtle.Screen()
alex = turtle.Turtle()

for aColor in ["yellow", "red", "purple", "blue"]:      # repeat four times
    alex.forward(50)
    alex.left(90)

wn.exitonclick()

```



Activity: 4 -- ActiveCode (ac6_5_3)

In the previous example, there were four integers in the list. This time there are four strings. Since there are four items in the list, the iteration will still occur four times. `aColor` will take on each color in the list. We can even take this one step further and use the value of `aColor` as part of the computation.

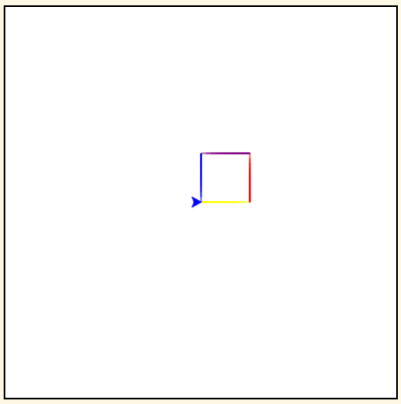
Save & Run 4/30/2021, 8:37:45 PM - 2 of 2

```
import turtle            # set up alex
wn = turtle.Screen()
alex = turtle.Turtle()

for aColor in ["yellow", "red", "purple", "blue"]:
    alex.color(aColor)
    alex.forward(50)
    alex.left(90)

wn.exitonclick()

```



Activity: 5 -- ActiveCode (ac6_5_4)

In this case, the value of `aColor` is used to change the color attribute of `alex`. Each iteration causes `aColor` to change to the next value in the list.

The for-loop is our first example of a **compound statement**. Syntactically a compound statement is a statement. The level of indentation of a (whole) compound statement is the indentation of its heading. In the example above there are five statements with the same indentation, executed sequentially: the import, 2 assignments, the *whole* for-loop, and `wn.exitonclick()`. The for-loop compound statement is executed completely before going on to the next sequential statement, `wn.exitonclick()`.

Check your Understanding

iter-5-1: How many times will the for loop iterate in the following statements?

```
p = [3, 4, "Me", 3, [], "Why", 0, "Tell", 9.3]
for ch in p:
   print(ch)
```

- A. 8
- B. 9
- C. 15
- D. Error, the for statement needs to use the range function.

Check me Compare me

✔ Yes, there are nine elements in the list so the for loop will iterate nine times.

Activity: 6 -- Multiple Choice (question6_5_1)

iter-5-2: How does python know what statements are contained in the loop body?

- A. They are indented to the same degree from the loop header.
- B. There is always exactly one line in the loop body.
- C. The loop body ends with a semi-colon (;) which is not shown in the code above.

Check me Compare me

✔ The loop body can have any number of lines, all indented from the loop header.

Activity: 7 -- Multiple Choice (question6_5_2)

iter-5-3: Consider the following code:

```
for aColor in ["yellow", "red", "green", "blue"]:
   alex.forward(50)
```

```
alex.left(90)
```

What does each iteration through the loop do?

- A. Draw a square using the same color for each side.
- B. Draw a square using a different color for each side.
- C. Draw one side of a square.

Check me | Compare me

✔ The body of the loop only draws one side of the square. It will be repeated once for each item in the list. However, the color of the turtle never changes.

Activity: 8 -- Multiple Choice (question6_5_3)

You have attempted 9 of 8 activities on this page

7.4. Strings and for loops">
ngs and for loops">
7.6. The Accumulator Pattern">

✔ Completed. Well Done!

© Copyright 2017 bradleymiller. Created using Runestone 4.1.17.

| Back to top