1.11 Looping

Repeating blocks of code by calling a function more than once, as in example 1.5, can get cumbersome when it needs to be repeated many times. A **loop** repeats a block until some stopping condition is met. One type of loop in Python is a `while` loop, which repeats a block of code while its conditional expression evaluates to `True`. For instance,

```
n = 0           # Initialize n
while n < 5:
    print(n)
    n += 1      # Increment n (i.e., n = n + 1)
```

The loop evaluates the conditional expression `n < 5` and, if in fact `n < 5`, executes the block of code. After the block finishes, the test is repeated and potentially the block of code. This will repeat indefinitely, until the conditional expression evaluates to `False`, in which case the loop exits and execution resumes after the code block. The block will be executed 5 times, printing `0` through `4` to the console.

Another type of Python loop is a `for` loop, which has no explicit conditional expression, instead iterating through an iterable object like a list, , until it reaches the end. For example,

```
l = ["foo", "bar", "baz"]
for s in l:
    print(f"Say {s}")
```

This prints

```
Say foo
Say bar
Say baz
```

It is common to loop through a `range` with a `for` loop, as in the following:

```
for k in range(2, 8):
    print(k, end=" ") # Prints on the same line
```

This prints the following to the console:

```
2 3 4 5 6 7
```

Often, a loop index is required inside a `for` loop. The syntax for this requires an identifier for the index and an `enumerate` type object to be iterated through. The constructor function `enumerate()` assigns an index to each element of its iterable argument (e.g., a list). For instance,

```
names = ["Manny", "Bella", "Amadeus"]
signs = ["Libra", "Virgo", "Sagittarius"]
for i, name in enumerate(names):
    print(f"{name} is a {signs[i]}")
```

This prints the following to the console:

```
Manny is a Libra
Bella is a Virgo
Amadeus is a Sagittarius
```

Looping through a dictionary is similar, but we need the `items()` of the dictionary for the key-value pair, as follows:

```
sounds = {"dog": "woof", "cat": "meow", "fox": "ring-ding-ding"}
for k, v in sounds.items():
    print(f"The {k} says '{v}'")
```

This prints the following to the console:

```
The dog says 'woof'
The cat says 'meow'
The fox says 'ring-ding-ding'
```

1.12 Summary [Outlined]

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.