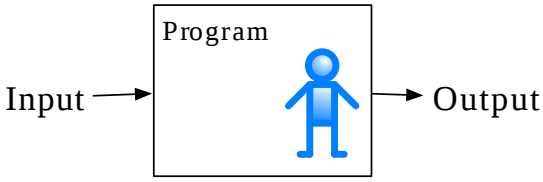
14.4: STARTING WITH PROGRAMS

Contributed by Chuck Severance
Clinical Associate Professor (School of Information) at University of Michigan

A program in its most basic form takes some input, does some processing, and produces some output. Our elevator conversion program demonstrates a very short but complete program showing all three of these steps.

```
usf = input('Enter the US Floor Number: ')
wf = int(usf) - 1
print('Non-US Floor Number is',wf)

# Code: http://www.py4e.com/code3/elev.py
```

If we think a bit more about this program, there is the "outside world" and the program. The input and output aspects are where the program interacts with the outside world. Within the program we have code and data to accomplish the task the program is designed to solve.



A Program

When we are "in" the program, we have some defined interactions with the "outside" world, but those interactions are well defined and generally not something we focus on. While we are coding we worry only about the details "inside the program".

One way to think about object oriented programming is that we are separating our program into multiple "zones". Each "zone" contains some code and data (like a program) and has well defined interactions with the outside world and the other zones within the program.

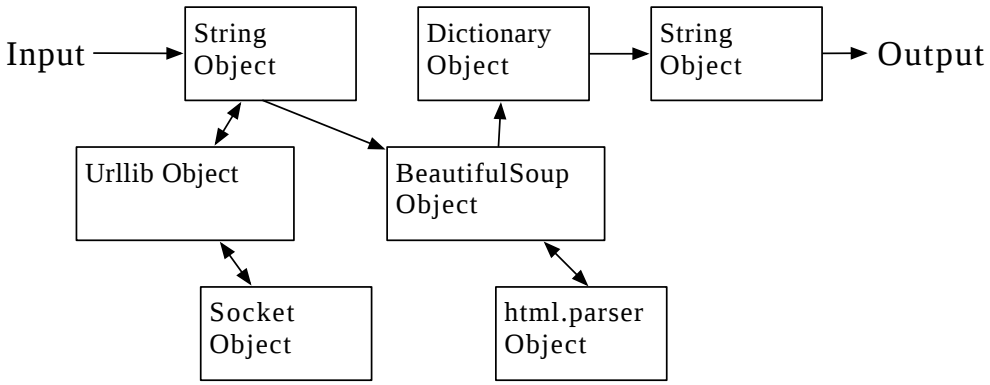
If we look back at the link extraction application where we used the BeautifulSoup library, we can see a program that is constructed by connecting different objects together to accomplish a task:

CODE 14.4.1 (PYTHON):

```
# To run this, you can install BeautifulSoup
# https://pypi.python.org/pypi/beautifulsoup4

# Or download the file
# http://www.py4e.com/code3/bs4.zip
# and unzip it in the same directory as this file

import urllib.request, urllib.parse, urllib.error
from bs4 import BeautifulSoup
import ssl

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

url = input('Enter - ')
```

```
html = urllib.request.urlopen(url, context=ctx).read()
soup = BeautifulSoup(html, 'html.parser')

# Retrieve all of the anchor tags
tags = soup('a')
for tag in tags:
    print(tag.get('href', None))

# Code: http://www.py4e.com/code3/urllinks.py
```

We read the URL into a string, and then pass that into `urllib` to retrieve the data from the web. The `urllib` library uses the `socket` library to make the actual network connection to retrieve the data. We take the string that we get back from `urllib` and hand it to BeautifulSoup for parsing. BeautifulSoup makes use of another object called `html.parser` [1] and returns an object. We call the `tags()` method in the returned object and then get a dictionary of tag objects, and loop through the tags and call the `get()` method for each tag to print out the 'href' attribute.



A Program as Network of Objects

We can draw a picture of this program and how the objects work together.

The key here is not to fully understand how this program works but to see how we build a network of interacting objects and orchestrate the movement of information between the objects to create a program. It is also important to note that when you looked at that program several chapters back, you could fully understand what was going on in the program without even realizing that the program was "orchestrating the movement of data between objects". Back then it was just lines of code that got the job done.