Lab: Functions

Sometimes code should be reused in several different programs. When that happens we pull the code into a library function and call it from the other programs. We'll do that in this lab.

Creating and calling a function

1. In your bin directory, create a new file called customerfunctions.bash.
2. In it, write a function called listCustomers which should simply print out some test text to the screen. You know, "Hello world" or something like that.
3. Edit the Customers.bash script. Have it call listCustomers. (Hint: You'll need to include customerfunctions.bash).
4. Run and test it from the command line. When you can see your test text, you know your function is being seen and is running properly.

Functions receiving a parameter

Right now you've got Customers.bash creating a table with all our customers. If we want to reuse that code in another shell script, we should probably pull it into a library function. We'll do that next.

5. Change listCustomers to receive a parameter called *format*.
6. If format is blank, just print out the contents of people.txt to stdout.
7. If format is "htmlTable", then print them all out as rows of an HTML table. Here's an example:

```
<tr>
<td>Ronald</td><td>Swanson</td>
<td>Pawnee PARD</td>
<td>769 Farragut Way</td>
<td>Pawnee</td><td>IN</td><td>47149</td>
<td>(812) 873-3340</td>
<td>rswanson@pawnee.ci.in.us</td>
</tr>
```

(Hint: this is exactly what you're doing in Customers.bash already. Feel free to copy and paste from Customers.bash into listCustomers.)

8. Change Customers.bash to call your new listCustomers function.
9. Run and test. Congratulations! Your code can now be used in any number of other shell scripts.

Creating more functions

10. In that same library file create functions called ...

- CreateCustomer
- ReadCustomer
- DeleteCustomer
- UpdateCustomer

11. Make each of those functions simply print "Not implemented yet". You've just created a bunch of function stubs.

12. Test that each of them are callable by another script.

Once you get them all callable, you can be finished.