**Loops and lists 2**

Here is another complex example that makes use of all that we’ve learned so far, specifically, functions, if statements, while loops and lists.

This program will take a string and shift each letter along one in the alphabet. In order to do this we need to compare each letter to the alphabet to find its position which we can then use to determine the new letter to place into our new, shifted string. We also need to take into account that not all characters in our string will be a letter and place them into our shifted string too.

**String\_shift.py**

# Attempt to match a letter to its position in the alphabet and return that number

def find\_letter(letter, my\_list):

position = 0

# Go through the whole alphabet list

while(position < len(my\_list)):

# If there is a match at that position

if(letter == my\_list [position]):

# If there is a match, return position

return position

else:

# Otherwise go to next position

pass

position += 1

# If no match, return original character

return letter

# Take each letter in a string and shift each along to the next letter in the alphabet

def shift\_one(string):

alphabet = ["a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z"]

position, new\_string = 0, ""

# Iterate through each letter in the string

while(position < len(string)):

# Check if letter is in the alphabet

char\_pos = find\_letter(string[position], alphabet)

# If it is, char\_pos will be an integer

if(isinstance(char\_pos, str) == False):

# If position 25 (z) position = 0 (a)

if(char\_pos == 25):

char\_pos = 0

# Increment position by 1

else:

char\_pos += 1

# Insert char + 1 to new string

new\_string = new\_string + alphabet[char\_pos]

# Otherwise it is a string

else:

new\_string += char\_pos

position += 1

print("Original string: {}".format(string))

print("Shifted string: {}".format(new\_string))

shift\_one("shift each letter along one.")