Poynting’s Python Society
Functions

This worksheet is aimed at introducing functions. The website CodingBat has lots of practice problems to help you to become familiar using functions. It doesn’t require you to have Spyder or install anything.

Functions in Python are a block of code that only runs when you call them. They are very useful, make your code more readable and save time.

To create a function, use this formatting:

```python
def function(variable):
    #write your function
    return result
```

Don’t forget to return!!
Otherwise, your function is useless (it will return None). If your code isn’t working and you can’t figure out why, it’s a good idea to check that you have a return statement in all of your functions.

Here’s a simple example:

```python
def sqrt(x):
    square_rooted = x ** (1/2)
    return square_rooted
```

We define a function that will square root whatever value you give it.

We can now use this function to make calculations quicker (barely in this case):

```python
sqrt(2) #<-- this calls the function for x=2
print(sqrt(9)) #<-- this prints the result in the console
print(5 + 2 * sqrt(25))
```

We call the square root of 2 and print the square root of 9. The output in the console is

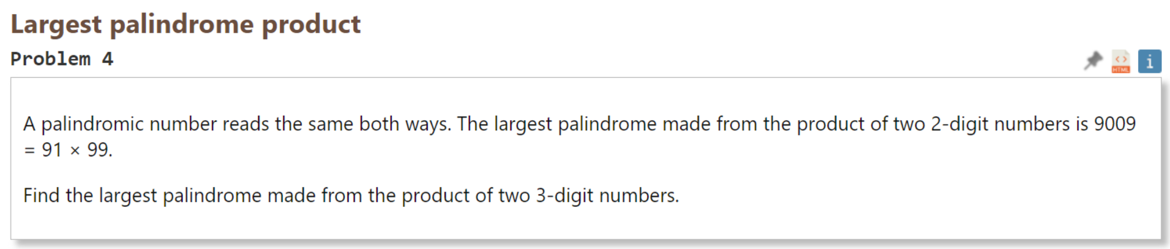
```
3.0
15.0
```

Your ‘variable’ and ‘result’ are dummy variables. This means they can be anything, as long as they are consistent with the rest of your code:

```python
def sqrt(lama):
    wombat = lama ** 2
    return wombat
```

This will give you the same results as before

Here’s an example where using a function is useful:



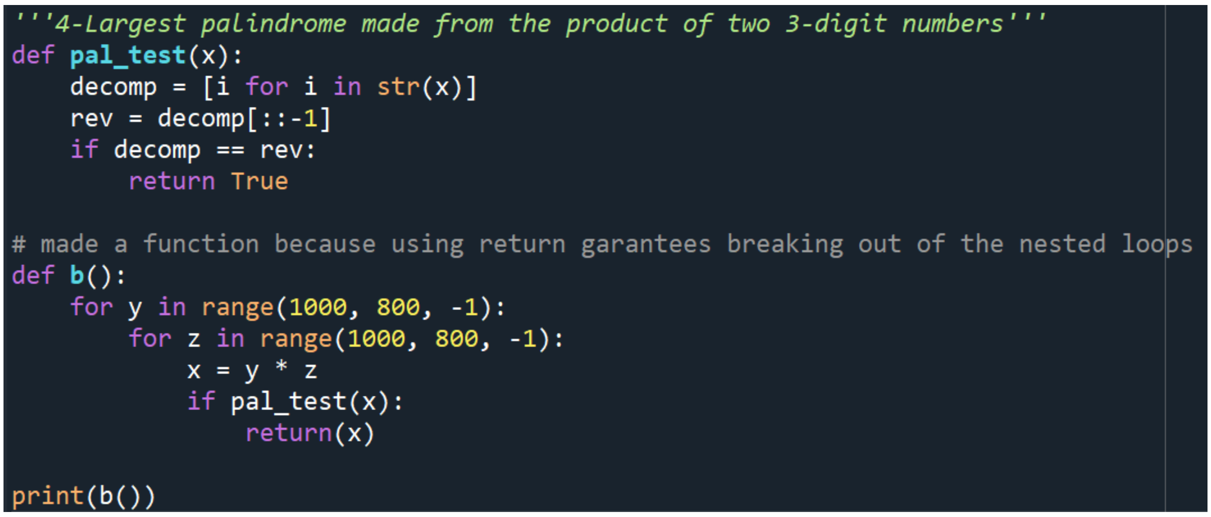
Largest palindrome product

Problem 4

A palindromic number reads the same both ways. The largest palindrome made from the product of two 2-digit numbers is 9009 = 91 × 99.

Find the largest palindrome made from the product of two 3-digit numbers.

This is problem 4 of Project Euler that we have seen last week.
To solve it, I used nested loops (loop in a loop) (there is probably a better way to do this!!). An easy and sure way to break out of nested loops is by using a function:



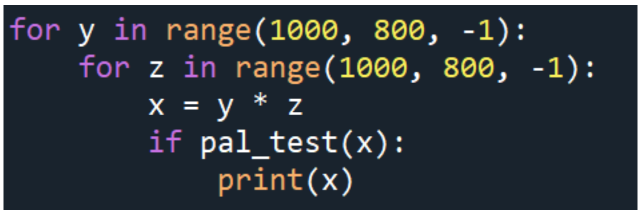
```python
'''4-Largest palindrome made from the product of two 3-digit numbers'''
def pal_test(x):
    decomp = [i for i in str(x)]
    rev = decomp[::-1]
    if decomp == rev:
        return True

# made a function because using return garantees breaking out of the nested loops
def b():
    for y in range(1000, 800, -1):
        for z in range(1000, 800, -1):
            x = y * z
            if pal_test(x):
                return(x)

print(b())
```

Here the function pal_test(x) determines if a given number is a palindrome or not. It will return True if the value we feed it is a palindrome. It is useful to define this function because other problems in Project Euler ask us to work with palindromes. So, we can use this function defined for problem 4 to solve different problems.

The function b() guarantees that the output will be the single value we are looking for. If instead, we had this:



```python
for y in range(1000, 800, -1):
    for z in range(1000, 800, -1):
        x = y * z
        if pal_test(x):
            print(x)
```

The output would be a bunch of values, and it would not be clear which is the one we want

In the function b(), we are looping through a range of values and then checking if their multiple is a palindrome using the previous function. Note "if pal_test(x):" is equivalent to "if pal_test(x) is True:"

The function b() doesn't have a variable. Functions don't necessarily need to be fed a value.

Try doing some problems in Warmup-1 in CodingBat to get used to using functions!