Unit 15

Functions & Modules

Asg 15.2 (Coding)

In [12]:
```
# set up notebook to display multiple output in one cell

from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"

print('The notebook is set up to display multiple output in one cell.')
```

```
The notebook is set up to display multiple output in one cell.
```

Practice Problem #1: Write a fruitful function sumTo(n) that returns the sum of all natural numbers up to and including n. So sumTo(10) would be 1 + 2 + 3 + ... + 10 which would return the value 55. Use the equation (n * (n + 1)) / 2.

In [4]:
```
def sumTo(n):
    return (n*(n+1))/2

int(sumTo(10))
```

Out[4]:
```
55
```

Practice Problem #2: Rewrite the function sumTo(n) that returns the sum of all integer numbers up to and including n. This time use the **accumulator pattern**.

In [10]:
```
def sumTo(n):
    runningtotal = 0
    for counter in range(1, n+1):
        runningtotal += counter
    return runningtotal
int(sumTo(10))
```

Out[10]:
```
55
```

Practice Problem #3: Write a fruitful function that adds the first n positive odd integers without using an **accumulator function**.

In [16]:
```
def sumTo(n):
    return n**2
sumTo(5)
```

Out[16]: 25

Practice Problem #4: Write a fruitful function that adds the first n positive odd integers that uses an **accumulator function**.

In [23]:
```
def sumTo(n):
    total = 0
    for x in range(1,n+1,2):
        total += x
    return total

sumTo(10)
```

Out[23]: 25

Practice Problem #5:
Write a function called mySqrt that will approximate the square root of a number, call it n, by using Newton's algorithm. Newton's approach is an iterative guessing algorithm where the initial guess is n/2 and each subsequent guess is computed using the formula: newguess = (1/2) * (oldguess + (n/oldguess)).

In [24]:
```
def mySqrt(n, guesses):
    oldguess = n/2
    for i in range(guesses):
        newguess = (1/2)*(oldguess + (n/oldguess))
        oldguess = newguess
    return newguess

int(mySqrt(9,6))
```

Out[24]: 3