# Computing for mathematics handout 3 - Functions, Lists and For Loops, Iteration versus Recursion

Lecturer: Vince Knight

Office: M1.30

email: knightva@cf.ac.uk

**Office hours: Thursday 1300-1500**

## What you have learnt this week:

* Lists, appending variables to lists, using list comprehensions;
* Dictionaries;
* Writing and reading data to file;
* Recursion versus iteration.

## Functions

* When you define a function you do not **use** it:
* def mean(lst): """ A function to return the mean of a list Arguments: lst: A list of numbers Outputs: The mean """ sumofelements = sum(lst) N = len(lst) return lst / float(N)
* The above just creates a **tool** that we can use if we want to:
* print mean([1,2,3,4,5])

## Lists and for loops

* A list is a python object that **contains** other python objects:

someoddnbrs = [1,3,5,7,9,11]

* We can use a for loop (see sheet 1) to 'iterate' (ie 'go through') the elements of that list:

for k in someoddnbrs: print k

* We can apply a function to a list:

def makeeven(k): """ A function to minus 1 from a number Arguments: k: an odd number Output: k - 1 """ return k - 1someevennbrs = []for k in someoddnbrs: someevenbrs.append(makeeven(k)) print someevennbrs

* We can do this in 1 line using list comprehensions:

someevennbrs = [makeeven(k) for k in someoddnbrs]

## Iteration versus recursion

* Iteration is an approach for defining a function that loops through elements.
* Recursion is an approach for defining a function that 'calls itself' until a base case is reached.

## What you should do next:

* You have had a lot of information delivered to you in a short amount of time, go back through the previous sheets to make sure you understand the basics.
* **Get started on the third sheet!**
* To make the best use of the lab sessions turn up having finished your sheets;
* If anything is still unclear **please** come and see me during office hours.