

Functions: passing and returning lists

Look at how lists work with functions:

1. Returning lists from functions
2. Passing lists to functions
3. Passing lists to functions and changing them
4. Different types of object

Returning lists from functions I

- Example: create a list in a function and return it to main()

```
import math
import random

def randomPoint(d):
    x = d*math.cos(random.uniform(0.0,math.pi))
    y = d*math.sin(random.uniform(0.0,math.pi))
    pt = [x,y]
    return pt # Return a list

def main():
    p = randomPoint(10.0) # Get a point
    print("x is " + str(p[0]) + " y is " + str(p[1]))

main()
```

- Behaves as you would expect

Returning lists from functions II

- Python tries to make things 'easy', so alternatively you can use this syntax

```
import math
import random

def randomPoint(d):
    x = d*math.cos(random.uniform(0.0,math.pi))
    y = d*math.sin(random.uniform(0.0,math.pi))
    return x, y # automatically returns a list

def main():
    a, b = randomPoint(10.0) # assigns values from the list
    print("x is " + str(a) + " y is " + str(b))

main()
```

- Does exactly the same thing as previous example, but is perhaps easier to read?
- You may find this a useful structure for checkpoint 5

Passing lists to functions

- Passing a list to a function and accessing the list elements works as you would expect:

```
def sumSquares(vals):  
    total = 0.0  
    for i in range(0, len(vals)):  
        total = total + vals[i]**2    # Access list elements  
    return total  
  
def main():  
    v = [2.0, -4.6, 6.7, -3.4, 2.6]    # Create a list  
    t = sumSquares(v)  
    print("Sum of squares is " + str(t))  
  
main()
```

Passing lists to functions and **changing** them

- This is where things get tricky:

```
def add_const(vals):  
    c = 5.0  
    for i in range(len(vals)):  
        vals[i] = vals[i]+c # add constant to elements of vals  
  
def main():  
    xdata = [0.0]*10 # Create list of 10 zeros  
    add_const(xdata)  
    print(xdata)  
  
main()
```

- Result of `print(xdata)` is: `[5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0]`
- You have created a list in `main()`, passed it to a function and changed the values in that function...
- but the values of the list elements in `main()` are changed!

Different types of object

- The full story here is rather complex
- There are two type of object in Python
 - **Mutable**: which means objects can be altered
 - **Immutable**: which means objects can't be altered, and attempting to do so generates a new object
- Details are beyond this course, but will be covered in
 - Computer Simulation (option in Semester 2)
 - Computer Modelling (In Junior Honours for Physics students)