

# Quick introduction of survival kit

# Control Flow

```
a=5; b=6
```

```
if a > b:  
    x = 2  
else:  
    x=3  
print x
```

```
y=2 if a >b else 3  
print y
```

# Functions

- Blocks of code that perform a specific task
- We have already seen many :- list(), type() etc.
- You can do your own:- Defined using the *def* keyword

```
myfun()  
print 'heres a function working'  
  
def myfun():  
    print "Hello"
```

```
myfun(5)  
print 'a function with an arg'  
  
def myfun(a):  
    print "inside", a
```

```
x = add(2,3)  
print x  
print 'heres function with a return'  
  
def add(a,b):  
    return a+b
```

# Functions

Function with arbitrary number of arguments

```
sumitall(2,3,4,5)
```

```
sumitall(9,15,17)
```

```
def sumitall(*values):  
    total = 0  
    for i in values:  
        total +=i  
    return total
```

Functions are objects. You can pass one function as an argument of another

Function documentation

```
help(sumitall)
```

```
def sumitall(*values):  
    """  
    This function illustrates  
    the use of arbitrary num  
    of arguments  
    """  
    total = 0  
    for i in values:  
        total +=i  
    return total
```

# Modules

- Collections of objects
- *math* module has 42 objects (*pi*, *cos* etc.)
- You can write your own module
- Different ways of importing a module

# Example Module - Example.py

```
"""
This is a custom module.
Containing some functions for the purpose of demonstration.
"""

def fun1():
    print "Inside fun1"

def fun2():
    print "Inside fun2"

pi = 3.14
e = 2.7

print "I am a Custom Module"
```

# Numpy

- How do I add two arrays in python?

```
a1 = [1,2,4]  
a2 = [10,20,40]
```

- Numpy way, the fast way

```
import numpy as np  
a1 = np.array([1,2,4])  
a2 = np.array([10,20,40])  
a3 = a1+a2  
print a3
```

- A numpy array is *not a list*, but a table of elements, all of the **same type**.

# Array operations

- Addition +
- Subtraction -
- multiplication \*
- Division /
- Modulo %
- Square \*\*

```
ai = numpy.array([1.,2., 5.])
```

```
print ai.dtype
```

See difference between  
`numpy.empty(N1)`, `numpy.zeros(N2)`,  
`numpy.ones(N3)`

```
a1 = numpy.array([[1,2,3],[4,5,6]])
```

What is `a1[0][1]`, `a1[0,1]`, `a1[0,1:]`,  
`a1[:,0]`

Attributes:- `ndim`, `shape`, `size`,  
`dtype`



# Array Manipulation

- Splitting [split, vsplit, hsplit, ...]
- Adding / removing [delete, insert, append, resize....]
- Rearrange [reshape,...]
- Joining[concatenate, stack, column\_stack... ]



*scipy*

- python library for scientific and technical computing
- has modules for optimization, linear algebra, integration, FFT, solvers of differential eqns, statistical functions, etc.



*astropy*

- Collection of software packages for astronomy
  - Coordinates, units, date / time, FITS file handling, model fitting, astrostatistics tools, cosmological calculations etc.

# Let us plot stuff!

- *matplotlib* : a plotting library for

```
>>> import numpy as np
```

```
>>> import matplotlib.pyplot as plt
```

```
>>> ...plot your stuff...
```

```
>>> plt.plot(x,y) #linear plot
```

```
>>> plt.loglog(x,y) #log scale
```

```
>>> plt.savefig('fig.pdf')
```

```
>>> plt.show()
```

