Subprograms

More Inconsistent Terminology

- Function
  - Ideally, a function takes any number of *parameters* (up to a system-dependent limit) and returns a single item.
  - In practice, this is too limiting so programmers and languages have ways around it.
- Subroutine
  - A subroutine takes any number (up to some limit) of parameters and returns any number (up to some limit) of values.
  - The terminology is used formally only in Fortran (as far as I know) but informally in other languages.

More Terminology

- Procedure, Subprogram
  - Sometimes used as generic terms for function or subroutine
- Method
  - A procedure that is only accessible through a defined type (a class or object).

Python

- Syntax

```
def mysub(x,y,z):
    statements
    statements
    return <expression>
```

Python def

- Python functions can return only one <item> but that item can be any object, in particular a tuple, list, or dictionary.
- If you do not specify a return value Python returns the special value None

Optional Arguments

- Syntax:

```
def func(x,y=0,z=2):
    return x+y-z
```

- If y or z is not present in the argument list when func is called, they take the default values.

Lambda Expressions

- Lambda expressions define expressions to be evaluated without giving them explicit function definitions.
- Like inline functions, they must be expressible as a single expression (no statements allowed)

```
newlist=map(lambda x:x+1, mylist)
```

  - This adds 1 to each element of mylist and returns the result in newlist.

Variable Scope

- The *scope* of a variable is the range over which it has a defined value. In Fortran, scope is defined by the program unit. In Python, the scope of a variable is the code block within which it is first referenced. So a calling program may have a variable named x, and a function may also have a variable named x, and if x is not an argument to the function then it will be distinct from the x in the main program.

Python Scope

- Variables defined above a def are global to the functions below it in the module (remember that every Python script is a module).
- Variables defined within a procedure are local to the procedure unless they are declared with the `global` keyword. This is the only declaration used in Python.
- Use globals sparingly if at all

Python Example

Try with and without the global declaration.

```
var = 'foo'
def ex(s):
    global var
    var = 'bar'
    print 'inside the function var is ', var
    return s+var
 print ex(“fu”)
 print 'outside the function var is ', var
```

Recursion

- When a function calls itself this is called *recursion*.
- Make sure you have a stopping condition that *will be* met!!
- Fortran requires the recursion keyword, Python just does it.
- Don't try this until you are completely comfortable with regular functions!! But it's sometimes the best way to express an algorithm.

Everybody's Favorite Recursion

- Fibonacci numbers:
- F(0)=0
- F(1)=1
- F(2)=F(1)
- F(3)=F(2)+F(1)
- F(4)=F(3)+F(2)
- …
- F(n)=F(n-1)+F(n-2)

Python

```
def fib(n):
    if ( n < 0 ):
        print “Illegal value”
        return None
    elif ( n==0 ):
        return 0
    elif ( n==1 ):
        return 1
    else:
        return fib(n-1)+fib(n-2)
```