Python notes

Python Intro

Python is an extremely powerful and dynamic object-oriented programming. There are several reasons to use Python. One of the biggest draws is portability. Python runs on almost every operating system. In addition python is fairly easy to pick up

Invoking Python

One *windows* you invoke the python interpreter by launching the `cmd` utility and entering in `python` which should give you a prompt similar to the one bellow.

```
Python 2.6.6 (r266:84297, Aug 24 2010, 18:46:32) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information
>>>
```

Functions

One of the most basic building blocks of any programming language is the function. In python functions are defined with the def keyword. Functions have two main parts, the function declaration and the function body.

```
>>> def functionName():          #Decleration
>>>    print "Hello World!"      #Function body
...
>>> functionName()
>>> Hello World!
```

Functions can also take take arguments. These arguments can be passed in between the parentheses of the function definition. These arguments can then be accessed in the body of the function.

```
>>> def argumentFuction(a, b):
>>>    print "A=" + a + " B=" + b
...
>>> argumentfuction(1,2)
>>> A=1 B=2
```

Variables

Another basic building block of any programming language is variables. Variables consist of three components type, indentifier, and a value. `Python` is what is known as a `dynamic` language. What this means is that you are not required to include a type delcration, in contrast to strongly typed languages such as `Java` or `C++`.

Java

```
>>> int number = 5;
...
>>> System.out.print(number);
...
>>> 5
```

Python

```
>>> number = 5
...
>>> print(5)
...
>>> 5
```

Conditional

Python supports conditional statements in the form of if, else and elif. The way if statements work is the code below the if statement is executed if the condition is true.

```
>>> x = True
>>> if x == True:
>>>     print "x is true"
>>> elif x == False:
>>>     print "x is false"
>>> else:
>>>     print "x is neither"
```

Looping Constructs

Python features several constructs to repeat a statement. The first of these explressions is the `while` loop. Thie while loop repeats the code below the loop while the condition is true.

While loop

```
>>> x = 1
>>> while x < 10:
>>>     print(str(x) + ","),
>>>     x += 1
...
>>> 1,2,3,4,5,6,7,8,9,
```

Another type of loop in Python is the `for` loop. The syntax of the for loop is `for item in sequence: block`. Each loop item is set to the next item in the sequence and the block of code is executed.

For loop

```
>>> word = "Python"
>>> for ch in word:
>>>     print(ch + "-"),
...
>>> P-y-t-h-o-n-
```

Python Classes

Object oriented programming *(O.O.P.)* is a programming paradigm that is used to model real world objects and situations. Two of the main concepts involved in *O.O.P.* are classes and objects. To make an analogy to the real world classes are like a blue print for a house and objects would be instances of that house.

Classes in Python are basically a collection of variables and functions. In the context of object oriented programming functions are referred to as methods and variables are referred to as fields, or attributes. Lets take a look at an example bellow.

Classes

```
>>> class test():
>>>    word = "Hello World"
>>>    def printWord(self):
>>>       print self.word
>>>
>>> t = test()
>>> t.printWord()
>>>
>>> Hello World
```

In the example above the class declaration starts with the keyword `class` followed by an identifier and a pair of parentheses. Following the declaration we have the variable definition `word` which is attached to the `test` class. Similarly we have a function attached to the test class called `printWord` which will print the contents of the variable word.