**Python for Everyone: Notes**

**Chapter 1: Why Program**

* Computers want to be helpful
  + Computers are built for one purpose – to do things for us
  + But we need to speak their language to describe what we want done
  + Users have it easy – someone already put many different programs (instructions) into the computer and users just pick the ones they want to use
* Programmers anticipate needs
  + Iphone applications are a market
  + Iphone applications have over 3 billion downloads
  + Programmers have left their jobs to be full-time iphone developers
  + Programmers know the ways of the program
* Users vs Programmers
  + Users see computers as a set of tools – word processors, spreadsheet, map, to-do list, etc
  + Programmers learn the computer ways and the computer language
  + Programers have some tools that allow them to build new tools
  + Programmers sometimes write tools for lots of users and sometimes programmers write little helpers for themselves to automoate a task
* Why be a Programmers
  + To get some task done – we are the user and programmer
    - Clean up survey data
  + To produce something for others to use
    - Fix performance problem in software
    - Add guestbook to a website
  + From a software creator’s point of view, we build the software. The end users (stakeholders) are our masters – who we want to please 0 often they pay us money when they are pleased. But the data, information, and networks are our problem to solev on their behalf. The hardware and software are our friends and allies in this quest
* What is code. Software, program
  + A sequence of stored instructions
    - It is a little piece of our intelligence in the computer
    - We figure something out then we encode it and then give it to someone else to save them the time and energy of figuring it out
  + A piece of creative art – particularly when we do a good job on user experience
* Definitions
  + Central processing unit: Runs the program – the cpu is always wondering what to do next. Not the brains exactly, very dumb but very fast
  + Input devices: keyboard, mouse, touch screen
  + Output devices: screen, speakers, printer, dvd burner
  + Main memory: fast small temporary storage, lost on reboot, aka ram
  + Secondary memory: slower large permanent storage – last until deleted, disk drive/memory stick
* Early learners: syntax errors
  + We need to learn the python language so we can communicate our instructions to python. In the beginning, we will make a lot of mistake and speak gibberish like small children.
  + When you make a mistake, the computer does not think you are cute, it says syntax error, given that it knows the language and you are just learning it. It seems like python I cruel and unfeeling.
  + You must remember that you are intelligent and can learn. The computer is simple and very fast but cannot learn. So it is easier for you to learn python than for the computer to learn englsh
* Elements of python
  + Vocabularly, words: variables and reserved words
  + Sentence structures: valid syntax patterns
  + Story structure: constructing a program for a purpose
* Reserved words
  + You cannot use reserved words as variable names identifiers
  + False, None, True, and, as, assert, break, class, if, return, def, del, elif, else, except, return, for, from, global, try, import, in, is, lambda, while, not, or, pass, raise, finally, continue, nonlocal, with, yield
* Sentences or lines
  + X = 2 <- assignment statement
  + X = x + 2 <- assignment with expression
  + Print(x) <- print statement
* Programming paragraphs, Python scripts
  + Interactive python is good for experiments and programs of 3-4 line
  + Most programs are much longer, so we type them into a file and tell python to run the commands in the file
  + In a sense, we are giving python a script
  + As a convention, we add .py as the suffix on the end of these files to indicate they contain python
* Interactive vs script
  + Interactive: type directly to python one line at a time and it responds
  + Script: you enter a sequence of statements (lines) into a file using a text editor and tell python to execute the statement in the file
* Program steps or program flow
  + Like a recipe or installation instructions, a program is a sequence of steps to be done in order
  + Some steps are conditional they must be skipped
  + Sometimes a step or group or steps is repeated
  + Sometimes we store a set of steps to be used over and over as needed several plaes through the program
* Sequential steps
  + When a program is running, it flows from one step to the next. As programmers, we set up paths for the program to follow
* Repeated steps
  + Loops (repeated steps) have iteration variables that change each time through a loop