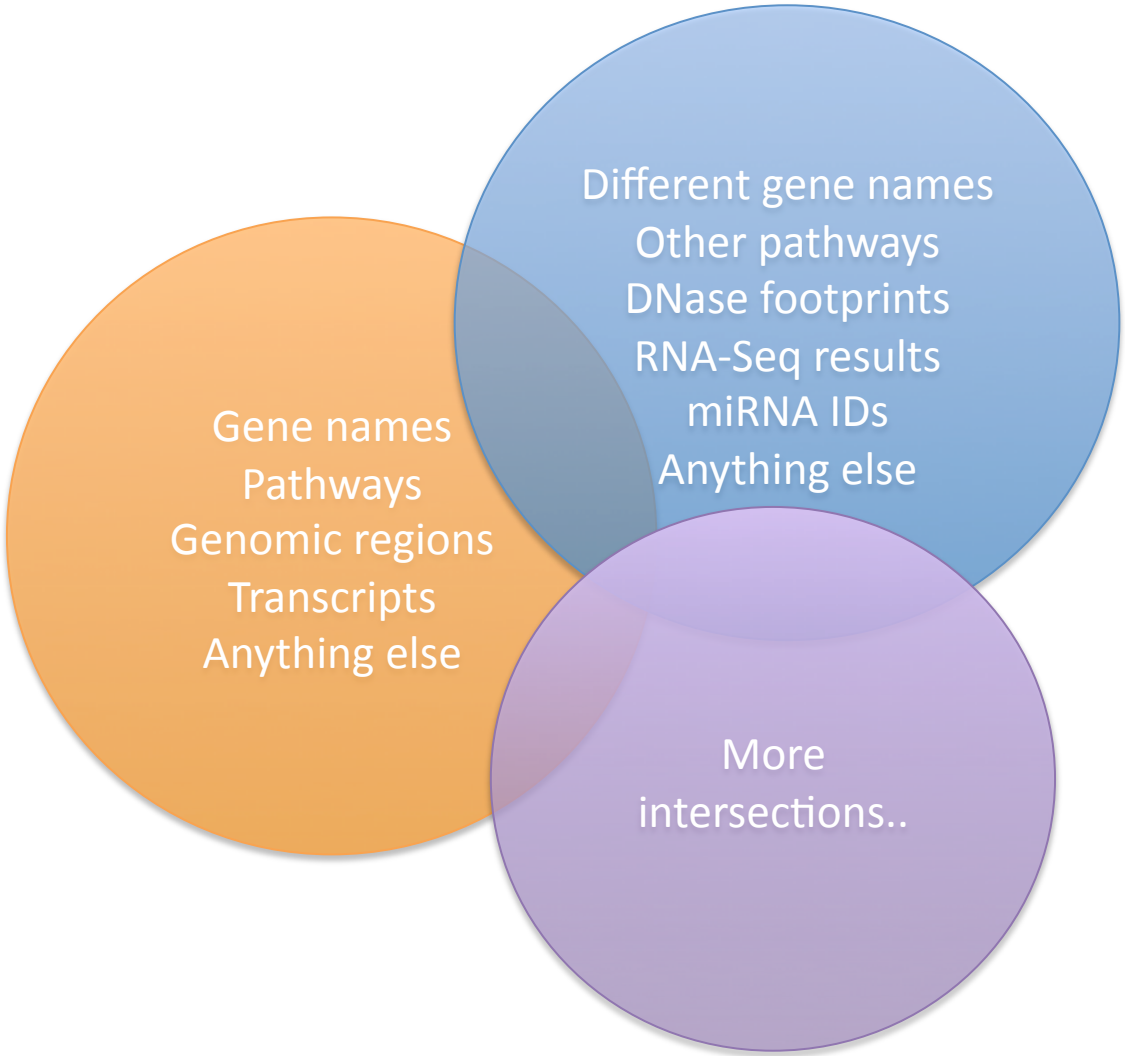
Python I

Arjan van der Velde
arjan.vandervelde@umassmed.edu
https://sites.google.com/site/gsbsbootstrappers/courses-workshops/python
http://bioinfo.umassmed.edu/bootstrappers/bootstrappers-courses/python1/index.html

Sessions:
Tuesday, December 2nd, 5pm-7pm
Thursday, December 4th, 5pm-7pm
Tuesday, December 9th, 5pm-7pm
Thursday, December 11th, 5pm-7pm
Amp III S6-102

Basic Bioinformatics



Examples

- Find pathways overlapping with upregulated genes
- Find promoters overlapping with ChIP-Seq reads
- Find upregulated genes that are novel (that don't overlap) relative to some other treatment
- Find primers that don't overlap with high GC-content regions of the genome

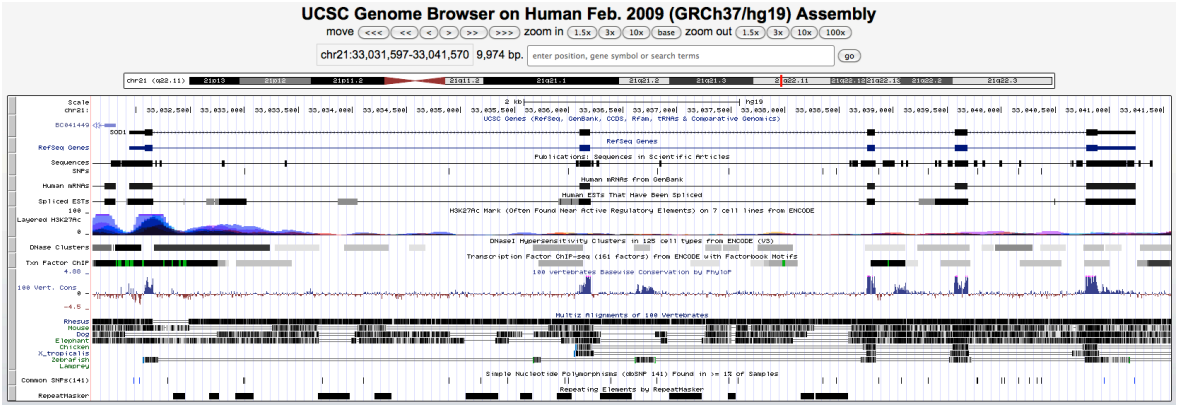
Basic Bioinformatics

chr9	4974730	4974744	CEBPB	14.6061	+	6.11e-06	GATTGCGCCACTGC	-1	K562	CEBPB	ENCFF002CLR	0
chr13	25037026	25037040	CEBPB	14.8182	+	4.55e-06	TATTGCACCACAGC	-1	K562	CEBPB	ENCF
chr2	204064308	204064322	CEBPB	14.6061	+	6.11e-06	GATTGCGCCACTGC	-1	K562	CEBPB	ENCF
chr9	129127385	129127399	CEBPB	14.5758	+	6.38e-06	GATTGCACCACTGC	-1	K562	CEBPB	ENCF
chr1	147043983	147043997	CEBPB	14.6061	+	6.11e-06	GATTGCGCCACTGC	-1	K562	CEBPB	ENCF
chr12	98781472	98781486	CEBPB	13.9394	-	1.34e-05	GATTGCAAAACTCC	-1	K562	CEBPB	ENCF
chr7	7157279	7157293	CEBPB	14.6061	+	6.11e-06	GATTGCGCCACTGC	-1	K562	CEBPB	ENCFF002CLR	0
chr19	17969839	17969853	CEBPB	14.0758	-	1.15e-05	GATTGCGCCACTGT	-1	K562	CEBPB	ENCF
chr2	238986128	238986142	CEBPB	14.3788	-	8.21e-06	TGTTGCATCATACT	32.2695238042559			K562
chr4	72514304	72514318	CEBPB	13.6212	+	1.82e-05	TGTTGCATCATTTC	-1	K562	CEBPB	ENCF
chr21	45973113	45973127	CEBPB	13.6818	-	1.73e-05	GATTGCTCAACCTC	-1	K562	CEBPB	ENCF
chr5	62157235	62157249	CEBPB	14.1818	-	1.03e-05	GCTTGCACCATCTC	-1	K562	CEBPB	ENCF
chr11	129734538	129734552	CEBPB	13.3182	-	2.36e-05	GGTTGCATCATTGT	-1	K562	CEBPB	ENCF
chr4	154492332	154492346	CEBPB	14.1667	+	1.05e-05	CGTTGCATAATCTT	-1	K562	CEBPB	ENCF
chr3	42644802	42644816	CEBPB	14.5758	+	6.38e-06	GATTGCACCACTGC	-1	K562	CEBPB	ENCF
chr11	27165617	27165631	CEBPB	14.6061	+	6.11e-06	GATTGCGCCACTGC	-1	K562	CEBPB	ENCF
chr12	22628393	22628407	CEBPB	14.5455	-	6.62e-06	TATTGCACCATACA	-1	K562	CEBPB	ENCF
chr2	171532292	171532306	CEBPB	14.5758	+	6.38e-06	GATTGCACCACTGC	-1	K562	CEBPB	ENCF
chr6	134913938	134913952	CEBPB	13.4242	-	2.17e-05	TATTGCATAATAAA	-1	K562	CEBPB	ENCF

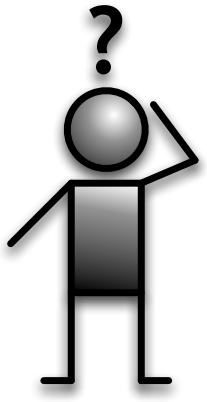
+

```
AGGACCATAAAACTCCAGTCAGTGAAC
AAACAAGTTAATAAACTAAAACTTTCA
TGGTTCTGGCATCGATGAAGAACGCAG
GTAATGTGAATTGCAGAATTCAGTGAA
GAACGCACATTGCGCCCCTTGGTATTC
TGTTCGAGCGTCATTTCAACCCTCAAG
TGGGCTCCGTCCTCCACGGACGCGCCT
GGTGGCGTCTTGCCTCAAGCGTAGTAG
TTGGAGCGCACGGCGTCGCCCGCCGGA
TATTTCTCAAGGTTGACCTCGGATCAT
AAGGTAAGAAAGTTTTTCCTTCCGCTG
CTGGGTGCTGGGTGCTGGGTGCTGGGT
TTGCCTTATCGCTTCGGTGAGGGGCAT
TTGGCCCGCGCTAAGCCTCGTTCGGGC
CGCATCTGGTTTTTTTGCGACCGGCGT
```

+



=



Objectives

- Learn the logical structure of programming languages
- Open, analyze and close large data files
- By the end, write simple programs to calculate statistics, compare large data files to each other and extract relevant data
- Know how to apply programming to simple large scale tasks

Python is a great place to start

- The command are like natural English
- Many modules exist that enable common tasks, such as statistics, graphing, etc.
- Python is an open language and a large community exists that supports and continuously improves the language
- Python is a good language for text processing and therefore for many Bioinformatics tasks

Course Layout

Session I

- Intro
- What is programming?
- Introduction iPython Notebooks
- Variables and data types, simple conversions
- Expressions and conditionals

Session II

- Lists and dictionaries
- Simple iteration
- Splitting and sorting
- Functions

Course Layout

Session III

- Scripting
- Reading and writing files
- How are (text) files represented?
- Delimiters, column based text data
- Data transformations

Session IV

- Sample problems
- Using existing code (modules)
- Loading modules
- Read a genome?
- How about sequence data?

Course Layout

Session V

Learning by doing

- Programming is very practical
- It's a craft
- Short explanations (5-10 min) followed by a set of instructive exercises (10-30 min)
- 3-5 blocks each session

What is Python?

- Python is a programming language that came about in the 90s and has become very popular over the past ten years
- There is a solid community and there are many online resources
- Currently the are two main branches of Python, version 2 and version 3. Python 3 is not fully backward compatible and 2.7 is still the most widely used version
- Python is an interpreted language; programs written in the Python language are run by the Python interpreter (called *python*)
- Python commands can be typed directly into the interpreter but are usually saved in a file (a script) that is then read and executed by *python*.
- In the first two sessions we will be using iPython Notebook to interact with *python*

Getting help

- Python comes with a built-in help system, called *pydoc*, as well full documentation online.
- https://docs.python.org/2.7/reference/
- https://docs.python.org/2.7/library/

Troubleshooting errors

- In programming, most things don't work the first time
- By breaking a complex statement into parts, you can often see where the programming language didn't understand you

- Example: 2 / (4 + 6) * 4 should give 0.8 but it doesn't

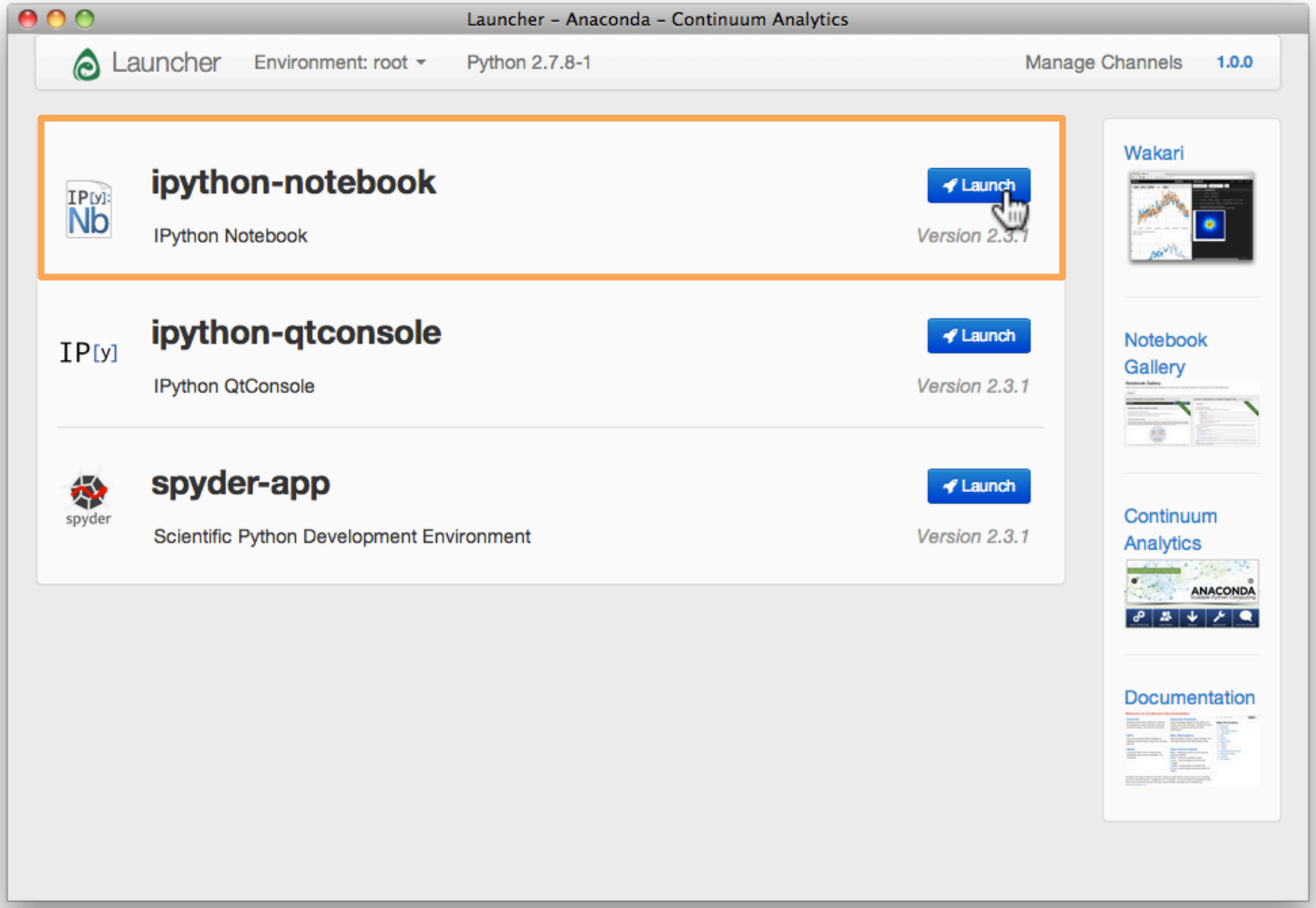
  4 + 6

  2 / 10

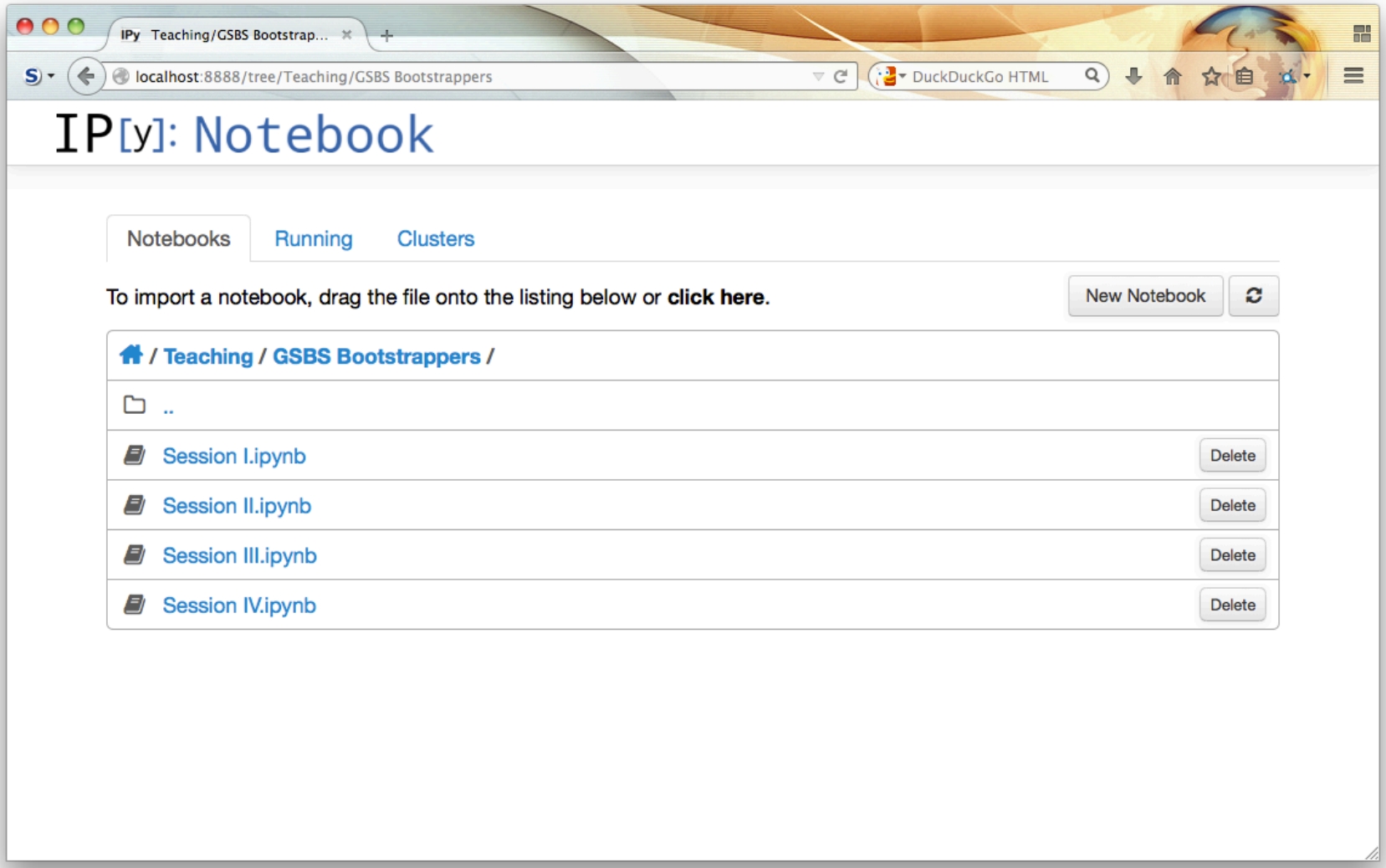
  0.2 * 4

Let’s start


Launcher


Launcher – Anaconda – Continuum Analytics
Launcher
Environment: root
Python 2.7.8-1
Manage Channels
1.0.0
ipython-notebook
IPython Notebook
Launch
Version 2.3.1
ipython-qtconsole
IPython QtConsole
Launch
Version 2.3.1
spyder-app
Scientific Python Development Environment
Launch
Version 2.3.1
Wakari
Notebook Gallery
Continuum Analytics
Documentation

iPython Notebook



Exercises 1

- The goal of these exercises is to get familiar with the Python interpreter and with iPython notebook

Exercises 1 Explanation

- Characters with quotes around them are strings
- Adding two strings together concatenates them
- Numbers without quotes are integers
- Adding two integers together returns the sum of the two integers
- Adding an integer and a string together is not allowed because python doesn't know which type of addition is meant.
- Strings and integers each have certain operations that can be done on them whose results can be built into more complex expressions
- Trying to do math operations on a string or string operations on numbers is an extremely common bug in bioinformatics programs.

Information storage

- All information entered into python has a data type or 'type'
- Simplest data types: *str*, *int*, and *float* (strings, integers, and decimals)
- Examples:
- 'hello' (string, known to python as *str*)
- 52 (integer, known to python as *int*)
- 35.4 (decimal, known to python as *float*)
- Python automatically guesses each information type and has certain operations it can do on it.
- Information is stored in variables using an “=” sign. From then on, when you type the variable, python retrieves the value.

Things you can do with *int*, *str*, and *float*

- Integers:
  - Add, subtract, multiply, divide, take exponents, compute remainders, sort numbers by size, anything mathematical, results will always be integers (explains division problem from Exercises 1)
- Floats:
  - Same as integers, but results will be decimals (python recognizes the decimal point)
- Strings:
  - In programming, a string is a series of letters. A sentence in English would be an example (python recognizes the quotes).
  - Strings can concatenate two strings into a bigger one (like DNA), grab parts of strings, insert extra letters on the end of strings, change parts of a string into something else (like translating cDNA), sort things alphabetically, reverse strings, and much more
- Try Exercises 2

Exercises 2 conclusions

- Data types can sometimes be converted between each other, allowing formerly impossible operations to be performed
- When math is done involving both a decimal and an integer, the result will be a decimal
- A variable name can't start with a number or be anything that already has an intrinsic meaning to python (like an '=' sign, the addition sign, the quote symbol, or anything else that python already understands)
- Variables that have been assigned to data take on the properties of the data they have in them (addition of two variables that have numbers assigned is different from addition of two variables holding strings)
- Variables can be reassigned to different values and data types, but can't be used without being initially assigned to a value. Be careful not to confuse strings and variables!

Logic

- Python knows <, >, ==, !=, and other tests
- Strings and numbers can be compared and strings can even be compared to numbers
- Logic operators can be combined with 'and' and with 'or'
- You can even test whether some sequence is found within another sequence.

Conditionals

- Test whether some expression evaluates to *True*
- Conditional statement starts with '*if*' followed by the thing to test followed by a ':' symbol
- Execute conditional commands only when the thing to test is *True*
- Conditional commands are indented under the conditional statement
- First unindented line after the ':' is where normal code resumes (that happens regardless of the conditional)
- If the conditional is not *True*, an '*else*' statement can have its own commands that execute