* Included editor and debugger is included with Python, load with "py –m idlelib" (PyCharm recommended)
* Chris' github link <https://github.com/topherCantrell/class-IntroPython>
* "Py" interpreter notes
  + Using name of function shows pointed (e.g., "print" gives name of function, "a" will display the a pointer on the screen
  + Can use single or double quotes – no difference
  + Basic commands are function()
  + Quotes and ticks can be escaped with "\"

|  |
| --- |
| >>> print("Hello World")  Hello World  >>> asdf = 5  >>> print(asdf)  5  >>> qwer=(asdf+2)  >>> print(qwer)  7 |

* + Help functions available with help(print); using help(a) will give info on object type (e.g., integer)
  + Python is compiled at run-time – use "py –m py\_compile <file>.py" to force python to preserve bytecodes – stored in \_\_pycache\_\_ directory
  + To read python bytecodes, you load the disassembler with "import dis" then run "dis.dis(<FUNCTION>)"
  + Built-in help functions available using "help(<FUNCTION>)"
* PyCharm notes
  + Can open a directory rather than a virtual environment – will automatically select correct interpreter
  + Can create break points in a specific line to assist with debugging – will cause execution to stop at specified line
  + Can run or debug files by right-clicking them or using "Run" menu or using buttons at top right of IDE window
  + Terminal function on bottom of screen will create a Command Prompt inside project directory
  + TODOs can be created inside comments – indicated by "TODO" inside a comment (e.g., "# TODO This is a test TODO"
  + Can locate a file on system by right-clicking file and selecting "Show in Explorer"
* Functions
  + All functions return something, even if it is “None”
    - To have a function return a variable, use the command “return <variable>”
  + For example, the print functions returns "None"
  + Now x points to the object "None"

|  |
| --- |
| >>> x = print('hello')  hello  >>> x  >>> print(x)  None  >>> id(None)  140705890392192  >>> id(x)  140705890392192 |

* + Define your own function using "def <name>(<parameters>):"
    - Name must start with a letter or underscore
    - Parameters are not required
  + Functions are closed when the next line has a different indention level – Python is white-space sensitive

|  |
| --- |
| # define your function  def say\_hi():      print('hello')      print('there')     # call function  say\_hi() |

* + "Snake case" is preferred nomenclature for python functions – I.e., use of underscores in lieu of upper case letters - e.g., "say\_bye" instead of "sayBye"
  + Call stacks illustrated in snippets1.pptx - more info <https://sites.cs.ucsb.edu/~pconrad/cs8/topics.beta/theStack/02/>

* Comments are included in files by prepending # at the beginning of the line, or in the middle of the line
* Trivia – integers, strings, math functions can be included in files – will not cause syntax error but service no purpose
* New lines are indicated by "\n", e.g., print('hello\nthere\nworld')
* Can create multi-line strings with multiple tick marks, e.g.,

|  |
| --- |
| a= '''hello  there  world''' |