Python Programming for FRC Scouting and Data Analysis

Training Objectives

# Overview

This course is intended for students with little or no prior computer programming experience.

Students who complete this course will be able to use the Python programming language and PostgreSQL database server to store, share, manipulate, format, analyze, and visualize data. This course provides most of the skills necessary to maintain, operate, and extend the Issaquah Robotics Society’s (IRS) scouting system. (The course does not address constructing Android applications, which are also used in the IRS’s scouting system.)

The concepts and technologies learned in this course are applicable to any sort of data analysis used in engineering, business operations, and the sciences.

# Necessary Materials

Each student should have a personal computer with the Windows operating system and internet access.

# References

1. Windows Command Prompt: <https://www.computerhope.com/issues/chusedos.htm>
2. Windows Environment Variables: <https://windowsreport.com/edit-windows-path-environment-variable/>
3. Anaconda: <https://www.anaconda.com/download/>
4. Python Tutorial: <https://docs.python.org/3/tutorial/index.html>
5. Python Library Reference: <https://docs.python.org/3/index.html>
6. Google Python Style Guide: <https://google.github.io/styleguide/pyguide.html>
7. Atom Text Editor: <https://atom.io/docs>
8. Jupyter Notebooks: <https://jupyter-notebook.readthedocs.io/en/stable/>
9. Jupyter Widgets (a.k.a., ipywidgets): <https://ipywidgets.readthedocs.io/en/latest/index.html>
10. Jupyter Extensions: <http://jupyter-contrib-nbextensions.readthedocs.io/en/latest/>
11. Pylint User Manual: <https://pylint.readthedocs.io/en/latest/>
12. Pytest Documentation: <https://docs.pytest.org/en/latest/contents.html>
13. PyCharm Download: <https://www.jetbrains.com/pycharm/download/#section=windows>
14. Pycharm Help: <https://www.jetbrains.com/help/pycharm/meet-pycharm.html>
15. Git Version Control System: <https://git-scm.com/book/en/v2>
16. Github: <https://github.com/>
17. JavaScript Object Notation (JSON) Reference: <http://www.json.org/>
18. Extensible Markup Language (XML) Tutorial: <https://www.w3schools.com/xml/default.asp>
19. SciPy: <https://www.scipy.org/>
20. Pandas Documentation: <http://pandas.pydata.org/pandas-docs/stable/>
21. Regular Expressions Tutorial: <http://www.regular-expressions.info/tutorial.html>
22. Holoviews Documentation (Python graphing package): <http://holoviews.org/>
23. Structured Query Language (SQL) Tutorial: <https://www.w3schools.com/sql/>
24. Postgresql Reference: <https://www.postgresql.org/docs/10/static/index.html>
25. SQLAchemy Documentation: <http://docs.sqlalchemy.org/en/latest/>
26. CherryPy Hypertext Transfer Protocol (HTTP) Server: <https://docs.cherrypy.org/en/latest/index.html>

# Key Concepts

1. Data types
2. Program structure, cohesion, and coupling
3. Data structures and formats
4. Variable Scope and Lifetime
5. Object Oriented Programming
6. Relational Databases, Schemas, and Normalization
7. Descriptive Statistics

# Conditions That Apply to Multiple Training Objectives

1. Online documentation is available
2. Windows operating system
3. Anaconda Python stack is installed
4. Python interpreter
5. Text editor
6. Jupyter Notebook
7. Pycharm IDE
8. Postgresql Database Server Installed

| **#** | **Action** | **Conditions** | **Criteria** |
| --- | --- | --- | --- |
|  | Navigate through files and directories via the Windows command prompt. Ref: a | B | * Navigates to specific folder location and display folder contents * Creates and remove files and directories. |
|  | Install Python via Anaconda. Ref: c | A, B | * Displays Anaconda version at command prompt. |
|  | Run Python interpreter from command line. Ref: a, d | B, C | * Displays help topics. * Closes interpreter. |
|  | Use Python expressions to conduct basic mathematical calculations. Ref: d | B, C, D | * Obtains correct numerical result from story-problem style math problem. |
|  | Assign values to variables. Ref: d | B, C, D | * Can write statements that assign primitive types to variables. |
|  | Use variables in an expression. Ref: d | B, C, D | * Can use variables to break complex expressions into multiple parts. |
|  | Call built-in functions. Ref: d, e | B, C, D | * Passes arguments to function and assigns return value to variable. |
|  | Determine the type of a variable. Concepts: 1. Ref: d | B, C, D | * Can write expressions that return variable types or tests that a variable is a given type. |
|  | Convert values to different data types. Concepts: 1. Ref: d | B, C, D | * Int to float, float to string, etc. |
|  | Manipulate strings. Ref: d | A, B, C, D | * Concatenate string literals and variables. * Convert strings to upper or lower case. * Use indices and slices to extract portions of a string. |
|  | Write a function. Concepts: 2. Ref: d, f | B, C, E | * Function passes arguments and returns a value. * Has function doc string. * Complies with Google Python Style Guide, including comments. |
|  | Add comments and doc-strings to code. Ref: d, f | B, C | * Comments enhance understanding of code and are not redundant. * Complies with Google Python Style Guide. |
|  | Print output to console. Ref: d | B, C, E | * Output displayed on console. |
|  | Create and edit environment variables. Ref: b |  | * Can determine if PATH and PYTHONPATH variables are set correctly |
|  | Run Python script from command line. Ref: a, d | B, C, D | * Script prints data to console. |
|  | Obtain input arguments from command line. Ref: a, d | A, B, C, D | * Script accepts parameters. * Script prints values to command line, including comments. |
|  | Create Python module in text editor. Concepts: 2. Ref: d, g | B, C, D, E | * Can create a module that can be run from command line and prints output to screen. * Complies with Google Python Style Guide, including comments. |
|  | Draft and execute Python code in Jupyter notebook. Ref: d, h | B, C, F | * Start notebook server. * Create new notebook. * Create and edit notebook cells. * Execute cells. * Restart Python kernel and clear output. |
|  | Manipulate lists. Concepts: 3. Ref: d | B, C, F | * Create list. * Select list items using index slicing. * Determine length of list. * Modify list (append or modify list item using slicing. * Create and select items from nested list. |
|  | Manipulate dictionaries. Concepts: 3. Ref: d | B, C, F | * Create and modify dictionary. * Select items using dictionary keys. * Determine dictionary length. * Create and select items from a nested dictionary. |
|  | Use an If statement. Ref: d | B, C, F | * If statement uses Else or Elif elements. |
|  | Iterate over multiple items using for loop. Ref: d | B, C, F | * Iterate over all items in a list. * Iterate over all items (keys, values, or both) in a dictionary. * Iterate over a range object (i.e., standard for loop). * Use break or continue statements within for loop. * Complies with Google Python Style Guide. |
|  | Repeat statements using while loop. Ref: d | A, B, C, F | * Use break or continue statements within while loop. * Complies with Google Python Style Guide. |
|  | Use a version control system to create backups and code branches. Ref: o | A, B, C  Git version control system available | * Uses commits and branching. |
|  | Share code with others. Ref: o, p | A, B, C  Git and Github are available | * Clones and pulls from repositories. * Pushes to repositories. |
|  | Import and use a Python Standard Library package. Ref: d, e | A, B, C, F | * Import statement complies with Google Python Style Guide. |
|  | Extract data from text formatted as Javascript Object Notation (JSON). Concepts: 3. Ref: d, e, q | A, B, C, F  JSON contains both objects and arrays, with multiple levels of nesting | * Atomic data is available via primitive Python objects. * Uses conditional statements to select specific data. |
|  | Convert data to JSON. . Ref: d, e, q | A, B, C, F | * Output conforms to JSON specification. |
|  | Create a project in an integrated development environment (IDE). Ref: m, n | A, B, C, G | * Project contains at least one Python module. |
|  | Create a Python module. Concepts: 2. Ref: d | A, B, C, G | * Module includes functions and module level variables. * Module can be imported. |
|  | Evaluate Python module with Pylint: Concepts: 2. Ref: f, k | A, B, C | * Modifies code to eliminate Pylint warnings. |
|  | List, install, update, and remove third-party Python packages. Ref: a, c | A, B | * Uses conda at command line. |
|  | Use global and nonlocal variables inside a function. Concepts: 4. Ref: d | A, B, C | * Changes to variable in function are visible in global or nonlocal scope. |
|  | Generate list using list comprehension. Ref: d | A, B, C | * Accurate results. |
|  | Generate dictionary using dictionary comprehension. Ref: d | A, B, C | * Accurate results. |
|  | Recognize and interpret information formatted as extensible markup language (XML). Ref: r | Manual (will not manipulate XML via Python code) | * Identifies XML formatted text. * Identifies tags, attributes, and content, and nesting relationships. |
|  | Identify and correct XML syntax errors. Concepts: 3. Ref: r | A  Manual (will not manipulate XML via Python code) | * Corrects missing closing tags, missing root element, unquoted values, incorrect nesting. |
|  | Set a breakpoint in IDE. Ref: n | B, C, G | * Sets unconditional and conditional breakpoints. * Clears breakpoints when no longer needed. |
|  | Step through code in IDE debugging mode. Ref: n | B, C, G | * Determines value of variables at current and higher-level scopes. * Steps into, over, or out of code sections as intended. |
|  | Uses assert statements to validate assumptions. Ref: d | B, C | * Assert statement is logically correct. |
|  | Uses automated unit tests. Ref: l | A, B, C  Pytest installed | * Unit test per Pytest documentation. * Unit test provides accurate results. |
|  | Create and instantiate a class. Concepts: 5. Ref: d, f | A, B, C | * Includes instance fields and methods. * Includes constructor. * Per Google Python Style Guide. |
|  | Create and use a class with static fields and methods. Concepts: 5. Ref: d, f | A, B, C | * Per Google Python Style Guide. |
|  | Create and use an inherited class. Concepts: 5. Ref: d, f | A, B, C | * Overrides methods in the parent class, including constructor. * Per Google Python Style Guide. |
|  | Create and use an iterator class. Concepts: 5. Ref: d, f | A, B, C | * Items from object can be selected with and index, similar to accessing item in a list. * Per Google Python Style Guide. |
|  | Create and use a class with property (getter and setter) methods. Concepts: 5. Ref: d, f | A, B, C | * Per Google Python Style Guide. |
|  | Create and use tuples and sets. Ref: d | A, B, C | * Accurate results. |
|  | Throw a user-defined exception. Ref: d | A, B, C | * Exception documented in module or function docstring per Google Python Style Guide. |
|  | Catch and handle errors and exceptions. Ref: d | A, B, C | * Uses try, except, finally, and else sections. |
|  | Create and use a generator function. Ref: d | A, B, C | * Accurate results. |
|  | Convert between Python list objects and numpy arrays. Ref: s | A, B, C | * Accurate results. |
|  | Create a Pandas dataframe. Ref: t | A, B, C | * Displays dataframe. |
|  | Extract data from Pandas dataframe. Ref: t | A, B, C | * Accurate results. |
|  | Group and sort Pandas dataframe. Ref: t | A, B, C | * Accurate results. |
|  | Calculate summary statistics from Pandas dataframe. Concepts: 7 Ref: t | A, B, C | * Calculates minimums, maximums, averages, medians, and standard deviations. |
|  | Add and delete columns from Pandas dataframe. Ref: t | A, B, C | * Accurate results. |
|  | Rename columns on Pandas dataframe. Ref: t | A, B, C | * Accurate results. |
|  | Modify data in Pandas dataframe. Ref: t | A, B, C | * Accurate results. |
|  | Set and use multi-level indices on Pandas dataframes. Ref: t | A, B, C | * Accurate results. |
|  | Stack and unstack Pandas dataframes. Ref: t | A, B, C | * Accurate results. |
|  | Join Pandas Dataframes. Ref: t | A, B, C | * Accurate results. |
|  | Convert between JSON and Pandas dataframes. Ref: q, t | A, B, C | * Accurate results. |
|  | Insert non-String data into a String and add additional text and formatting. Ref: d, e | A, B, C | * Accurate results. * Use Python str.format() method and format strings. |
|  | Extract data from or modify a string using a regular expression. Ref: e, u | A, B, C | * Accurate results. * Uses anchors, character classes, back-references, grouping, and repetition. |
|  | Create an interactive user interface within a Jupyter notebook. Ref: h, i, j | A, B, C, F | * Interface includes multiple ipywidgets, including text boxes, buttons, sliders, Boolean widgets (checkboxes, toggle buttons, etc), selection widgets (dropdown boxes, selection sliders, etc.), and layout widgets. * Interfaces uses Jupyter extensions for initialization cells and hiding cells. * Interface uses formatted Markdown cells to guide user. |
|  | Create a scatter plot and calculate linear correlation. Concepts: 7. Ref: v | A, B, C | * Two numeric variables plotted against each other on X and Y axes. * Determines whether correlation is weak, moderate, or strong. |
|  | Create a line chart. Concepts: 7. Ref: v | A, B, C | * One or more numeric values plotted vs. time. |
|  | Create a histogram and evaluate distribution of data. Concepts: 7. Ref: v | A, B, C | * Identifies likely range of measurements for variable. * Identifies whether distribution appears to be unimodal or multi-modal, symmetric or skewed. |
|  | Create a box-plot chart and evaluate differences between subgroups. Concepts: 7. Ref: v | A, B, C | * Graphically displays variance for two or more subgroups of a single variable. * Evaluates likelihood that underlying probability distributions of various subgroups are different. |
|  | Create a stacked bar chart. Concepts: 7. Ref: v | A, B, C | * Ranks subgroups according to various variable and attribute categories |
|  | Display data in tabular format. Ref: t | A, B, C | * Accurate results. |
|  | Run a relational database server. Ref: x | A, B, C | * Can execute structured query language (SQL) commands on Postgresql service running on local machine. |
|  | Select data from multiple tables in relational database. Ref: w, x, y | A, B, C, F (via SQLAlchemy package), G (via DB Navigator Plugin or SQLAlchemy package) | * Uses SQL SELECT statement with joins and WHERE clause. |
|  | Update data in relational database. Ref: w, x, y | A, B, C, F, G | * Uses SQL UPDATE statement. |
|  | Insert data into relational database. Ref: w, x, y | A, B, C, F, G | * Uses SQL INSERT statement. |
|  | Draft schema for relational database. Concepts: 6 Ref: w, x, y | A, B, C, F, G | * Multiple tables with primary keys. * Defined one-to-one, many-to-one, and many-to-many relationships. * Schema in third normal form or higher. |
|  | Create Pandas dataframe from SQL statement. Ref: t, w, y | A, B, C, F, G | * Uses SQL features included in Pandas package. |
|  | Use Hypertext Transfer Protocol (HTTP) to receive data. Ref: e | A, B, C | * Uses urllib packages from Python Standard Library. * Receives FRC performance data from FIRST API HTTP server. * Receives FRC performance data from Blue Alliance HTTP server. |
|  | Respond to HTTP request. Ref: z | A, B, C | * Uses CherryPy server. |
|  | Simplify an algebraic expression | A, B, C | * Uses Sympy symbolic computation package |
|  | Create a Python package with modules that can be imported into other Python modules. Ref: d | B, C, G | * Does not rely on relative imports * Uses PYTHONPATH environment variable to locate package contents |