Python Programming Summer 2018

Final Study Guide

July 30, 2018

The test consists of

1. Multiple choice questions - 20 x 2 = 40 points
2. Given code, find the output - 3 x 5 = 15 points
3. Code writing questions - 1 x 20 = 20 points
4. Short answer questions - 8 x 5 = 40 points

- You will have an opportunity to earn 15 extra credit points.
- Please try and attempt all questions. You get points for trying.
- Anything from the slides/homeworks/handouts is fair game.
- Code Writing is based on the examples below and the concepts discussed in class. You will only be asked to write native python for the code writing. Nothing from specific libraries.
- Some of the multiple choice and free response questions will be on the libraries we discussed in class.
- Making me laugh might gain you points (depends on the quality of the joke).

Topics to study

- Python Fundamentals
  - Basic Python Syntax
  - Numeric and Sequence Data types
  - I/O - print statements and the input() function
  - Selection statements (if - else) and loops (while and for loops).
- Modules, functions and lists
  - Python functions, the "def" keyword
  - Different ways of passing arguments - positional arguments, keyword arguments and packing
  - Modules and importing modules
  - List Comprehension
  - Lambda Functions - Filters, Map/Reduce

- Python Built-In Data Structures
  - Lists - Building other data structures using lists
  - Tuples
  - Sets
  - Dictionaries
- File input and output
- Exceptions - raise and except
- Python Strings
- Object Oriented Programming with Python
  - Closures and Decorators
  - Classes - Defining classes, data attributes and instance methods, static attributes and methods, constructors, inheritance
  - Iterables and Iterators
  - Generators
- The Python Standard Library
  - Built-in Constants
  - time
  - system
  - os
  - re
  - copy
  - itertools
- Python Development Tools
  - virtualenv
  - logging
  - testing - writing unit tests
  - Basic documenting tools - just the names of the tools
- Numeric and Scientific Programming
  - The idea behind scientific and numeric programming - large scale number crunching using matrices.
  - numpy - the fact that numpy has several additional data types, arrays and matrix related functions.
  - scipy - the various scientific programming sub modules availabe under scipy
  - matplotlib - pyplot and the use for plotting
- Graphical User Interfaces for Python
  - The idea behind GUI programming - events and responding to events.
  - PyQT - widgets and the various widgets available under PyQt
  - Signals and Slots, the connect function

  - Layouts
  - Painting, brushes and palettes
- Flask
  - Web Design Basics
  - Servers and Clients, Requests and Responses
  - Flask Templating and variables, URL routing
  - Running a SQL query and populating a template
- Machine Learning
  - Supervised and Unsupervised learning
  - K-means clustering
  - pandas and scikit
  - Tensorflow and neural networks
- Parallel Programming - multiprocessing, Process and Pool, threading
- Networks - The concepts of client/server architecture, sockets.
- The multiple choice and the output questions will test your familiarity with the Python language and syntax. The code writing questions will test you knowledge of programming.
- Studying the topics listed above will be enough to pass the test. To get a 100, you would be required to study everything on the notes.
- You don't need to study from outside sources. The test is made entirely from the notes and assignments.

Some Sample Questions

1. Which of the following is not a valid Python Operator?
   (a) :
   (b) .
   (c) –
   (d) in

2. Which of the following QtWidgets can be used to create a drop-down list?
   (a) QComboBox
   (b) QPushButton
   (c) QListBox
   (d) QstatusBar

3. Analyze the following statement:

```
>>> a=(15, 25)
>>> b=(13, 24)
>>> c=a+b
>>> c
```

(a) (28, 49)

(b) (15, 25, 13, 24)

(c) (15, 13, 25, 24)

(d) Error as tuples are immutable

4. Write a Python program to read in a string from the user and print the frequency of each letter. You can assume upper and lowercase characters are different letters.
Sample Run

```
Enter the string: hash slinging slasher
{'h': 3, 's': 3, 'l': 2, 'g': 2, ' ': 2, 'r': 1, 'a': 2, 'i': 2, 'n': 2, 'e': 1}
```

5. In number theory, a perfect number is a positive integer that is equal to the sum of its proper positive divisors, that is, the sum of its positive divisors excluding the number itself. Write a function to print all the perfect numbers between 1 and 10000 (non inclusive).
Sample Run:

```
The perfect numbers are:
6
28
496
8128
```

6. What is the use of the `matplotlib` library?

7. What is the output of the following code?

```
def foo():
    return total + 1
total = 0
print(foo())
```