CS 5010 – Exam 1 – Study Guide

Topics that may be covered on Exam 1 *(note, **not** an exhaustive list)*:

1. Data conceptualization
   a. What is data?
      i. Data is raw, unprocessed, and uninterpreted information
      ii. It is an abstract concept
   b. Different types of data
      i. Qualitative
         - Descriptive information (relates to the quality of something)
      ii. Quantitative data
         - Discrete data
            a. Numerical information
            b. Can only take certain values (whole numbers, "counted")
         - Continuous data
            a. Numerical information
            b. Continuous range (often "measured")
      iii. Categorical
         - A limited, and usually fixed, number of possible values to describe an item (giving an item a "label" or placing into a "category")
   c. Properties of data
      i. Source – where did the data come from? *Trustworthy? Unbiased? Reliable? Sufficient?*
      ii. Quality – data maintained and data is clean
      iii. Scale – data is representative?
      iv. Variety – different types of data
   d. Value of data is extracted through context and presentation
      i. Case studies (case studies are used to provide context, while no direct questions on these will be asked on the exam, you can use them to provide examples to support your solutions)
         - Edward Tufte's redesign of the Space Shuttle Challenger data
         - Data presentation on the consumption of sugary soda
      ii. Context and presentation increases understanding and can change behavior/attitude. *Caution*: careful not to mislead/misinform!
   e. Data → Information → Knowledge
      i. What each of them are; difference between each of them
   f. Data products
      i. Case studies (case studies are used to provide context, while no direct questions on these will be asked on the exam, you can use them to provide examples to support your solutions)
         - CD Database
         - Google Flu Tracking

2. Python / Coding (General)

a. Be able to read code
b. Be able to show the output of a piece of code
c. Be able to compare two pieces of code
d. Describe the (many!) benefits of applying object-oriented principles
e. Describe what the major data structures are and their properties (and scenarios in which they would be best suited – e.g. lists, dictionaries, and tuples)
f. Decide on an appropriate data structure(s) to use given the scenario
g. Correctly implement the appropriate data structure(s) for the given scenario and manipulate data within this structure correctly and appropriately
h. Describe the major functions used and their properties (and scenarios in which they would be best suited)
i. Explain some of the elements that results in "good quality" code

3. Python

a. Python basics, including…
    i. Python is a general-purpose, high-level, dynamically typed programming language
    ii. When coding, adhere to good technique and style, including
        - Consistent spacing and indentation
        - Commenting
        - Appropriate naming conventions for variables (identifiers) and functions (related: knowing not to name a variable the same as a Python reserved word)
    iii. Basic Types
        - `int`, `long`, `bool`, `float`, `str`, `list`, `tuple`, `dict`, `file`
    iv. Getting input from the user and appropriately/correctly using this input
    v. Flow control
        - if, if-else, and if-else-if statements
        - loops; continue and break; loops with data structures
b. Functions
    i. Benefits of functions (including ideas behind code reuse)
    ii. How to write functions, and how to call functions, function arguments (parameter list)
    iii. Higher-Order Functions – Map, Filter, and Reduce
c. Modules
    i. Use of existing vs. user-defined modules
    ii. Importing modules
d. Object-Oriented Programming (OOP) – Classes
    i. Define a class in Python; creating instances of the class
    ii. Class fields/attributes
    iii. Class constructor (`__init__`)
    iv. Concept of "self"
    v. Printing class objects (`__str__`)

e. Object-Oriented Programming (OOP) – Inheritance
    i. Benefits of using inheritance
    ii. Base class (“super class”)
    iii. Derived class (“sub-class”)
    iv. “IS-A” relationships
    v. The derived class inherits from the base class
    vi. Public vs. Private (and Private vs. Protected) – e.g. for fields/attributes
    vii. “DRY” concept (“Don’t Repeat Yourself!”)
    viii. Base class constructor
        - Unique – be aware of how it’s written
        - Parameters to the base class constructor
        - Call to super-class constructor (important!)
        - Handling local fields/attributes
    ix. Can have multiple layers of inheritance (not just one base class and one derived class)
    x. Inherited methods from base class
    xi. Runtime polymorphism – compiler evaluates the “type” of the object that is calling the method and looks in that object’s class for that method
    xii. Overridden methods

f. Exception handling
    i. What are exceptions
    ii. Be aware of the exception hierarchy
    iii. Runtime errors vs compilation errors
        - “Runtime” errors are encountered during the running of the program
        - Whereas “compilation” errors (“syntax” errors) happen at, and are caught at, compilation time (before your program even gets the chance to run).
    iv. Try-Except and Try-Except-Finally statements to handle exceptions
    v. Merits of using Try-Except / Try-Except-Finally
    vi. Order of except statements is very important!
        - Want to catch a more specific exception before a more general exception
        - Exception statements are evaluated one by one from top to bottom
        - You cannot visit more than on except statement
        - The first except block that matches gets execute (match means the same exception or a descendant of the exception)
        - Avoid creating unreachable code!

Some Suggested Resources/References:

1. [Module 1] Data conceptualization and Python
   - 01-Data Conceptualization.pdf
   - 01-The Challenger-An Information Disaster.pdf
   - Python 3-X Basics.pdf
   - Module exercises
   - Python Scripts 1-13
   - Homework 1: Data Quiz
2. [Module 2] Python Part II
   - 02-Python – Map Filter Reduce.pdf
   - Python Scripts 14-21
   - Python Scripts sphere, spheretest, newton, newtontest, newtontest2
   - Homework 2: Python
3. [Module 3] Python Part III
   - 1-Private, Protected, and Public in Python.pdf
   - 2-Delete in Python.pdf
   - 3-Functions and Parameter List in Python.pdf
   - Python Scripts 22-26
   - Python exception hierarchy: https://docs.python.org/3/library/exceptions.html#exception-hierarchy

Notes:

- It might be clear but worth stating, knowledge on *how to use or install* Anaconda or Spyder ***will not*** be tested on this or any future exam
- Advanced printing techniques (including "format string") ***will not*** be explicitly tested on the exam, however you're welcome to use it as part of your solution if you wish
- Redefining/redirecting standard output (**stdout**) to print from the screen (standard output) to a file (see `pyScript06.py` and other related scripts) ***will not*** be explicitly tested on the exam, however you're welcome to use it as part of your solution if you wish

--*--

Specifics of the exam will be posted on Collab and discussed in the live session of the next module. Bring your questions! Of course you're also welcome to email me directly.

Good Luck! ☺

--*--