**Midterm Exam – Your Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Due: Sunday (see Syllabus)**

**Points: 160**

**EXAM RULES**

**DURATION: 3 HOURS**

**MATERIALS ALLOWED: - *Introduction to Python for Computer Science and Data  
 Science,* by Deitel & Deitel**

**- CT-206 Course Material (e.g., PowerPoint slides)**

**- Your own CT-206 notes and past assignments**

**HARDWARE ALLOWED: Personal Computer**

**SOFTWARE ALLOWED: Python Interpreter and IDE, Microsoft Word**

**INTERNET ACCESS: NOT ALLOWED**

**PHONE CALLS AND TEXTING: NOT ALLOWED**

**Instructions**

**You must take this exam in electronic form. Once you start the exam in Canvas, a timer starts. You *must* submit the modified exam document in Canvas before the end of the 3-hour period or you will earn a zero.**

**Place your written answers on this document, just below each question and above the horizontal separator. Add additional space, as needed. For the programming questions, start a new page and then paste your Python source code into this document. Follow your code with a screen capture of just the runtime output produced by your program during execution.**

**Submit your completed exam in Canvas.**

**EXAM QUESTIONS (5 QUESTIONS, 160 POINTS)**

**1. (20 points )** Define each of the following fundamental concepts in your own words.

1. Interpreted vs. Compiled Languages
2. Interpretive Overhead
3. Scripting (Procedural) vs. Object Orientation vs. Event Driven Programming Paradigms
4. Command Line (Prompt)
5. Functions
6. Control Flow
7. Data Structure
8. Control Structure
9. List Comprehension
10. Python’s Best Features

**2. (20 points, 2 points each)** True or False; circle your answers:

1. T F: A Python dictionary is a random-access data structure.
2. T F: A Python list is a sequential access data structure.
3. T F: Using functions to isolate code is a good programming practice.
4. T F: Lexicographical ordering starts by comparing the ASCII value of the first letters of   
    two strings to determine the order.
5. T F: Standard Input in Perl and Python is the keyboard.
6. T F: Standard Output in Perl and Python is the terminal.
7. T F: Regular expressions are used to process text.
8. T F: Python sets can be used to remove duplicate values in other data iterable data  
    structures.
9. T F: A Python list is a mutable, random access data structure of mixed type.
10. T F: Dictionaries are useful to store pairs of data items which are related and of the same  
     length.

**3. (40 points)** Write a Python function in that computes the sample mean (average) of a list of numbers. The return value should be the mean and the input value should be the list of numbers. Do not use a built-in function for the mean. Write a main to drive your function. Show an example of your function in use.

**4. (40 points)** Write a Python function that takes in a piece of sample text via a long string (you pick the string) and returns a dictionary with each word as a key and it’s frequency (the number of times it occurred in the original string) as the value. Write a main to drive your function. Show an example of your function in use. Be sure to strip out all punctuation

**5. (40 points)** Write a Python function which removes a collection of common words from a long piece of text. Be sure to strip out all punctuation. The common words to be removed are:

a, an, as, at, by, for, in, is, it, of, that, this, to, was, will, the

These are typical words which are considered to have low semantic value.

Process each paragraph provided below individually (build a function that handles a paragraph). Your end result for each paragraph should be a string or list containing the processed paragraph with the common words and punctuation removed. It is not required to put the text sample into a file (you can simply copy and paste into a string variable inside your Python script). Your main should pass paragraphs to your function and it should print what is returned as a collection of paragraphs.

Use the following text (taken from the official Python tutorial):

If you do much work on computers, eventually you find that there’s some task you’d like to automate. For example, you may wish to perform a search-and-replace over a large number of text files or rename and rearrange a bunch of photo files in a complicated way. Perhaps you’d like to write a small custom database, or a specialized GUI application, or a simple game.

If you’re a professional software developer, you may have to work with several C/C++/Java libraries but find the usual write/compile/test/re-compile cycle is too slow. Perhaps you’re writing a test suite for such a library and find writing the testing code a tedious task. Or maybe you’ve written a program that could use an extension language, and you don’t want to design and implement a whole new language for your application.

You could write a Unix shell script or Windows batch files for some of these tasks, but shell scripts are best at moving around files and changing text data, not well-suited for GUI applications or games. You could write a C/C++/Java program, but it can take a lot of development time to get even a first-draft program. Python is simpler to use, available on Windows, Mac OS X, and Unix operating systems, and will help you get the job done more quickly.

Python is simple to use, but it is a real programming language, offering much more structure and support for large programs than shell scripts or batch files can offer. On the other hand, Python also offers much more error checking than C, and, being a very-high-level language, it has high-level data types built in, such as flexible arrays and dictionaries. Because of its more general data types, Python is applicable to a much larger problem domain than Awk or even Perl, yet many things are at least as easy in Python as in those languages.

Python allows you to split your program into modules that can be reused in other Python programs. It comes with a large collection of standard modules that you can use as the basis of your programs — or as examples to start learning to program in Python. Some of these modules provide things like file I/O, system calls, sockets, and even interfaces to graphical user interface toolkits like Tk.

Python is an interpreted language, which can save you considerable time during program development because no compilation and linking is necessary. The interpreter can be used interactively, which makes it easy to experiment with features of the language, to write throw-away programs, or to test functions during bottom-up program development. It is also a handy desk calculator.

***Paste your Python scripts and screen captures of their executions here, starting each script-capture on a new page. Clearly identify each program with the question number for which it is an answer.***