CS 365: Digital Forensics
Spring 2020
Assignment #1

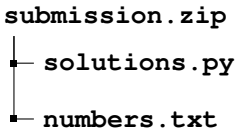
Arun Dunna
adunna@cs.umass.edu

Revision 1.0 - January 23, 2020

Due: January 29, 11:55pm

Submission Instructions

1. Your programming solutions for this assignment are to be written in **Python 3.6**.
2. Your programming solutions are to be submitted to the corresponding Gradescope programming assignment by the deadline **January 29, 11:55pm**. These should be submitted in a **ZIP file** containing your Python code files, in the following format:

   This will match the format of the assignment code distributable.

Prerequisites

1. Ensure that you are enrolled in the course on Gradescope. The assignment submission will open a few days before the deadline. If you have not been automatically enrolled, you can use the following enrollment code: **9DZ53K**
2. Setup and test your own environment for executing **Python 3.6** code. For example, this can be in an IDE such as PyCharm, or in a terminal with the **`python3.6`** command.
3. Obtain the assignment distributable from the course website: **`asgn01distrib.zip`**

1 Programming

The assignment distributable contains skeleton code in the **`solutions.py`** file. Edit the file to complete the following functions, further described in the docstrings of each function in the skeleton code. If types of inputs are not specified, and you receive a type that is impossible to deal with given the function specifications, raise a **TypeError** exception. If the input is a container of objects, such as a list, and the types of those objects are not given, assume that they can be any type (or a mix of types).

1.1 Python Basics [60 points]

(2 points) `sum(x, y)` : return the sum of integers x and y

(4 points) `divide(x, y, trunc=True)` : (assume integers) return x/y, and truncate if **trunc** is true

(2 points) `equiv_val(x, y)` : return true ***iff*** x and y are equal in value

(2 points) `equiv_mem(x, y)` : return true ***iff*** x and y point to the same object in memory

(5 points) `concat(a, b)` : return strings a and b joined together by a space, *i.e.* **"a b"**

(5 points) `listinfo(x)` : return a tuple of the first, last, and middle elements in x; $n(x) \geq 3$ and $n(x)$ is odd

(5 points) `mlast(x, m)` : return the last m elements of x; assume $n(x) \geq m$

(5 points) `summation(x)` : return the sum of elements in x; note sum refers to summing numbers, not strings

(5 points) `oddlist(x)` : return true ***iff*** all elements in x are odd

(5 points) `lookup(d, k)` : return the key k's associated value ***iff*** k is in dictionary d; return false otherwise

(20 points) `factorialdict(n)` : return a dictionary d containing the first n factorials from 0, *s.t.* $d[n] = n!$

1.2 Python I/O [40 points]

(5 points) `read_numbers()` : return the contents of **`numbers.txt`** as a string

(5 points) `list_numbers()` : return the contents of **`numbers.txt`** as a list; newlines separate elements

(30 points) `convert_bases()` : return the contents of **`numbers.txt`** as a list of tuples; in bases (2, 10, 16)