CS 365: Digital Forensics
Spring 2020
Assignment #3

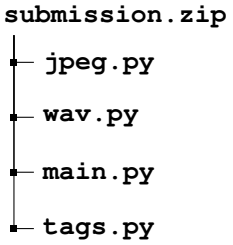
Arun Dunna
adunna@cs.umass.edu

Revision 1.0 - February 10, 2020

Due: February 19, 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 **February 19, 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.

   Alternatively, you can upload each Python file individually in Gradescope instead of using a ZIP file.

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: **`asgn03distrib.zip`**

1 Programming

The assignment distributable contains skeleton code in two files: **`wav.py`** and **`jpeg.py`**. It also contains **`main.py`** to test the program, and **`tags.py`** to use in your parsing of EXIF tags - **DO NOT EDIT THESE FILES**. Edit the WAV and JPEG Python files to complete following functions, further described in the docstrings of each function in the skeleton code. You may assume that the files you are reading (passed into your functions as strings) exist.

1.1 WAV Manipulation [50 points]

(15 points) `carve_wav(inputFile)`: extract any/all valid WAV files from the input file

(35 points) `parse_wav_header(inputBytes)`: extract fields/values from a WAV file byte sequence

1.2 JPEG Manipulation [50 points]

(10 points) `carve_jpeg(inputFile)`: extract any/all valid JPEG files from the input file

(40 points) `parse_exif(inputBytes)`: extract tags/values from any/all EXIF data segments

1.3 Extra Credit [30 points]

(30 points) `pack_wav(audioBytes, fields)`: pack the given audio data and WAV header field values to a WAV file, then return it; use the following specifications:

(a) *audioBytes* is a bytes object containing the raw audio/sound data. Assume it is always nonempty.

(b) *fields* is a tuple containing (nChannels, samplesPerSecond, averageBytesPerSecond, blockAlignment, bitsPerSample), which will be given in the format (int, int, int, int, int). These will always be set to valid values. All other header fields will have to be calculated yourself or through default/standard values.

(c) The *AudioFormat* field value is by default 1, corresponding to PCM.

(d) Return the full packed bytes as a bytes object. You can test your implementation by writing your output bytes object to a file.

(e) The `pack_wav()` function should be written in the **`wav.py`** file.