**ECE 1111: Engineering Computation I**

**Homework No. 13: Python Application Programming**

**Deposit your work in:**

**/data/courses/ece\_1111/current/homework/hw\_13/<lastname\_firstname>**

**Goal:** Introduce you to system level programming in Python.

**Description:** Create a collection of mp3 files. You can use this music:

*https://www.isip.piconepress.com/courses/temple/ece\_1111/resources/data/music/songs.tar.gz*

if you want to tolerate my selection of songs. Note the username and password are both “temple”. You can alternately user your own music. This tar file contains 30 mp3 files. Your overall task is to write a program that randomly shuffles the songs and “plays” them. The requirements of this shuffle program are:

1. Your code should find all files with a .mp3 extension in a directory tree (multiple levels) and build them into a list. Keep track of how many times a song has been played using this list. Test your code using a subset of these files to make sure it works before you attempt the entire list.
2. You must guarantee that all 30 songs will be played exactly once before you start playing a song a second time. Similarly, all songs must be played twice before you continue to the third pass.
3. Your program automatically progresses from one song to the next (use a loop). It must play forever.
4. Before a song is played, you must log it as played, so if you kill the script during playing of this song, it will restart from the next song that must be played (not a song that has been previously played).
5. The program must keep track of its status so that you can restart the program. If I kill your script in the middle of playing a song, it should resume from where it left off – meaning it should play the next most available song that it was originally planning to play next anyway. You don’t have to play the current song from the point where it was killed.
6. When all songs have been played once and you start the next pass, the order must be randomized (you can’t use the same order for every pass).
7. It must be coded in Python (v3).
8. It must run from a terminal window as a Linux command (e.g., my\_player.py). A GUI is not required. You can substitute “echo filename; sleep 1” for an actual mp3 player, since ece-000 doesn’t give you access to an audio system. However, you should be able to run it from your laptop using a standard mp3 play command.
9. You must demonstrate you can launch the command in the background and then kill the job with a kill command (as demonstrated in class). Describe what happens when you kill the shellscript – be very specific in how you explain what is happening.

Finally, try to minimize memory (don’t load all songs into memory) and run-time (you want this running in the background with a low priority).

The program should be called myplay.py (e.g., “*myplay.py $HOME/songs*”) and take a directory path as its argument. The first thing it should do is search that directory and its subdirectories for files that end in “.mp3” and add any new files to the history. If the history doesn’t exist, it should create it.

The output that I want to see is this (taken from my personal version of this script):

**58 (52, 1): playing file tosca\_-\_rolf\_royce\_feat\_stephen\_graf\_hadik\_wildner.mp3**

**59 (157, 1): playing file zero\_7\_-\_i\_have\_seen.mp3**

**60 (51, 1): playing file air\_-\_la\_femme\_d\_argent.mp3**

**61 (16, 1): playing file ms\_mr\_-\_hurricane.mp3**

**62 (60, 1): playing file variety\_labs\_-\_london\_in\_the\_rain.mp3**

**63 (49, 1): playing file koop\_-\_waltz\_for\_koop.mp3**

**64 (77, 1): playing file emancipator\_-\_when\_i\_go.mp3**

**65 (130, 1): playing file saint\_germaine\_-\_land\_of.mp3**

The first number is the index of the song being played. The second number is the position of the file in the original list of files (this is list is randomly shuffled after all the songs have been played once). The third number is the number of times the song has been played. All songs will be played once before it plays a song for the second time.

Note that when I kill the script, and run it again, it will not play any of the songs that have been previously played once. When it plays songs, it never plays the same song twice until all songs have been played the same number of times.