Concept

Thumb drive, contains music, playlists, program

↓

Raspberry Pi, with desktop icon starting the program

↓

Pi HDMI output to home stereo amplifier

Python program

Nested loop structure

Outer loop – select playlist, randomize etc.

Inner loop – play music and monitor keyboard for pause, skip etc.

Python program

- Outer loop structure
  - Initialization
    - Imports various modules
    - Variables used section – a commented list
    - Functions
    - Identify host, set appropriate working directory
    - Music locations – a commented list
    - Playlist locations – a commented list

IDENTIFY HOST AND SET WORKING DIRECTORY LOCATION APPROPRIATELY

```
myHost = gethostname()
print(f"Found host called {(myHost)}")

if myHost == "Idra":  # LaCie now persistant as M:
    os.chdir("M:/")
elif myHost == "raspberrypi":
    os.chdir("/media/pi/MUSICPLAYER/")
else:
    print("No known host found, program ending.")
    sys.exit()
```

Begin outer loop

```
while True:


# GET THE NAMES OF THE PLAYLISTS IN THE MUSIC DIRECTORY


    playLists = glob.glob("MusicCategories" + "/*")  # returns a list 'playLists' of files in the format MusicCategories\\ZZZ5.txt

    print(f"Number of Playlists found = {len(playLists)}.")  # check how many playlists found, this can be commented out later

    print()
```

# SECTION TO DISPLAY PLAYLIST NAMES AND PICK ONE TO PLAY
chosenPlayListName = playListPick(playLists)

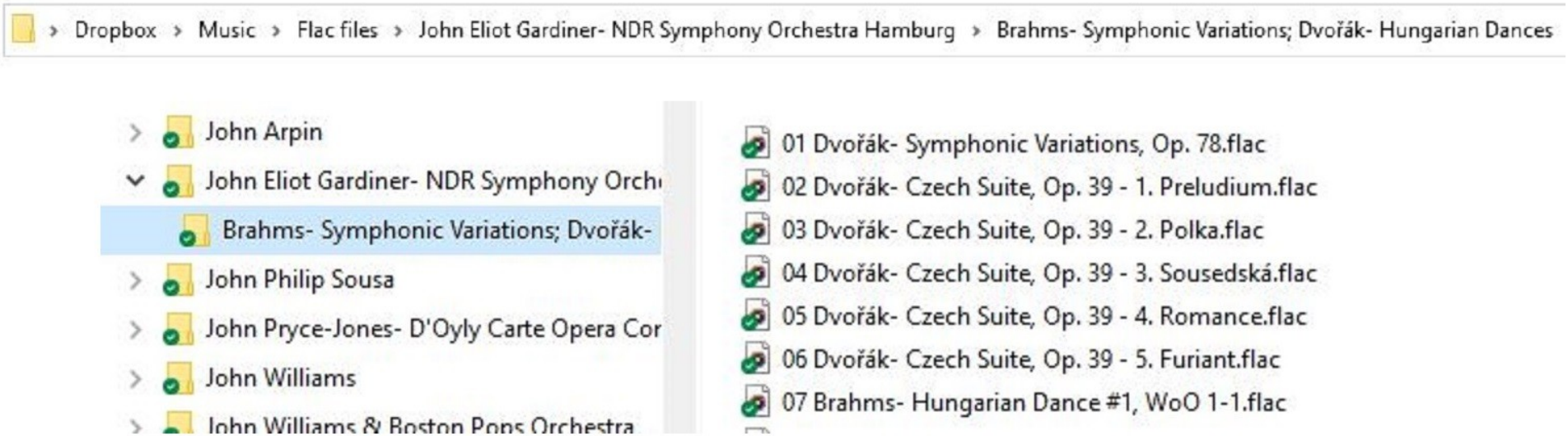
```
playListName = [0] * len(playLists)  # make a list as long as the number of playlists
    place = 0
    while place < len(playLists):
        playListName[place] = os.path.basename(playLists[place])
        #  Remove directory path to clean up playlist names, using'os' to strip off directory headers
        print(f"Playlist name {place} is {(playListName[place])} ")
        place += 1
    print()

    chosenPlayListNumber = int(input("What is the number of the playlist you want to hear?   "))
    chosenPlayListName = playLists[chosenPlayListNumber]
    print()
    print(f"ChosenPlayListName is {playListName[chosenPlayListNumber]}")

    return chosenPlayListName
pass
```

Picking a playlist screen

```
Number of Playlists found = 11.

Playlist name 0 is ZZZ5.txt
Playlist name 1 is ClassicMix.txt
Playlist name 2 is EasyListening.txt
Playlist name 3 is PopMusic.txt
Playlist name 4 is VocalClassic.txt
Playlist name 5 is VocalLight.txt
Playlist name 6 is WorldMusic.txt
Playlist name 7 is InstrumentalLight.txt
Playlist name 8 is Opera.txt
Playlist name 9 is InstrumentalClassic.txt
Playlist name 10 is ZZZ7.txt

What is the number of the playlist you want to hear?
```

```
# CONVERT CHOSEN PLAYLIST FROM A TEXT FILE TO A LIST OBJECT, STRIPPED OF LINE FEEDS

    songList = makeSongList(chosenPlayListName)


  songListTemp = []
    with open(f"{chosenPlayListName}", "r") as f:
        songListTemp = f.readlines()
# This list object includes a line feed at the end of each line which causes a failure to open the music file

    songList = []
    for item in songListTemp:
        item = item.replace("\n", "")
        songList.append(item)
# This list object is stripped of the line feed

    totalSongs = len(songList)
    print(f"There are {totalSongs} tunes in this playlist")
    print()
    return songList
pass
```

Why Python doesn’t like Dvořák



Display first few songs

```
def displayFirstFewSongs(songList):

    totalSongs = len(songList)
    x = min(5,totalSongs)
    print(f"The first {x} songs are;")
    count = 0
    while count < x:
        songName=os.path.basename(songList[count])
        print(songName)
        count = count+1
    print()
pass
```

Make a play order list

def randomizer(songList):

```
totalSongs = len(songList)
  playOrderList = list(range(0, totalSongs))  # range start number to stop number, stop number is not included in the list
  randFlag = False
  randSelect = input("Do you want to shuffle the songs?  Y or N ")
  if randSelect == "Y" or randSelect == "y":
      randFlag = True
      print()
      print("Music will play in random order")
      print()
  else:
      print()
      print("Music will not be randomized")
      print()
  if randFlag == True:
      random.shuffle(playOrderList)

  return playOrderList
```

Show first songs and make play order list

```
ChosenPlayListName is ZZZ5.txt
There are 7 tunes in this playlist

The first 5 songs are;
Chopin_S1.mp3
Pachelbel_Canon_Clip.mp3
Beet5.mp3
LonelyBull.mp3
Biber.mp3

Do you want to shuffle the songs?  Y or N
```

Outer loop completed

- List of locations of the songs to play
- List of the order to play the songs

Inner loop

- Must run two tasks simultaneously

- Play music in chosen order
    - stop when all music played
- Monitor for keyboard input
    - make appropriate response to input

Inner loop 1 – increment song

```
while True:
    if pygame.mixer.music.get_busy() == False:  # Music not playing
        if songNext == len(songList):  # No more songs in list
            reasonForBreak = "AllSongsPlayed"
            break
        else: # Play the next song and increment the counter
            playNext = playOrderList[songNext]
            nextSong = os.path.basename(songList[playNext])
            print(f"Next song is {nextSong}")
            print(f"s = skip, q = quit, p = pause, n = new playlist")
            print()
            pygame.mixer.music.load(songList[playNext])
            pygame.mixer.music.play()
            songNext += 1
```

Inner loop 2 – keyboard monitor

```
while True:
    if pygame.mixer.music.get_busy() == False:  # Music not playing
    |
    | some code
    |
    else:  # Music is playing
        time.sleep(0.1)
        if keyboard.is_pressed("n"):  # Change to another Playlist
            pygame.mixer.music.stop()
            reasonForBreak = "PickAnotherPlaylist"
            break
        if keyboard.is_pressed("q"):  # Quit
            pygame.mixer.music.stop()
            reasonForBreak = "Quit"
            break
```

Inner Loop – skip a track

```
else:  # Music is playing
        time.sleep(0.1)

        if keyboard.is_pressed("s"):  # Skip to next track
            print("Skipping this song")
            print()
            pygame.mixer.music.stop()
            if songNext == len(songList):  # No more songs in list
                print(f"That was the final song")
                reasonForBreak = "NoMoreSongs"
                break
            playNext = playOrderList[songNext]
            nextSong = os.path.basename(songList[playNext])
            time.sleep(1.0)
            print(f"Next song is {nextSong}")
            print(f"s = skip, q = quit, p = pause, n = new playlist")
            print()
            pygame.mixer.music.load(songList[playNext])
            pygame.mixer.music.play()
            songNext += 1
```

Inner loop has exited on “break”

```
if reasonForBreak == "Quit":
    print("Stopping on keypress 'q'")
    time.sleep(0.1)
    print("Program terminated")
    break

if reasonForBreak == "PickAnotherPlaylist":
    print("Setting up to choose another playlist")
    #tcflush(sys.stdin, TCIFLUSH) # clean out anything in the input queue - not available for windows
    time.sleep(2.0)
    pass
```