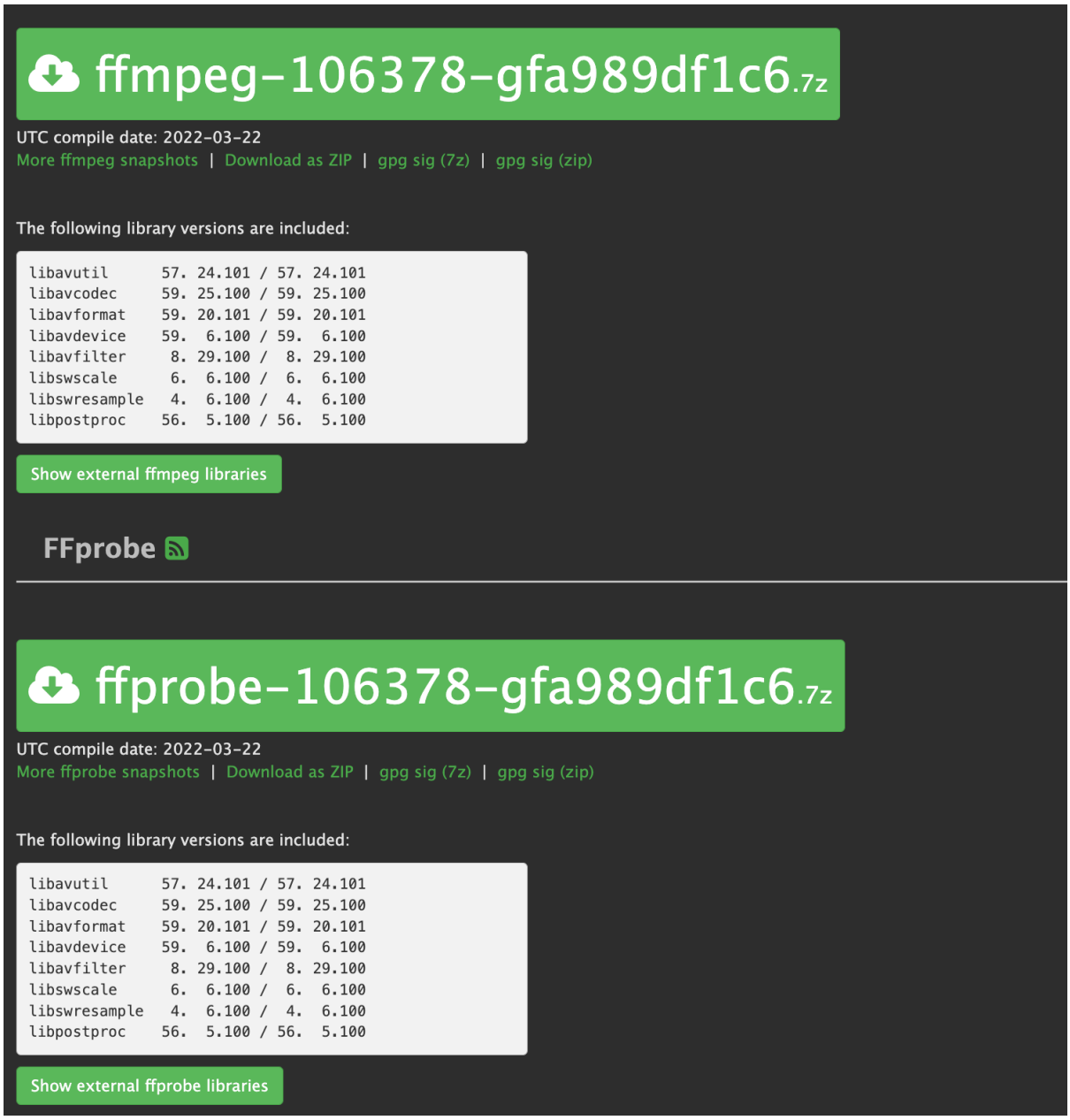
Exercise 3 Report

Coursera Exercise 2 & 3 Shareable Lab Link:
https://hub.labs.coursera.org:443/connect/sharedsmgnmkdp?forceRefresh=false

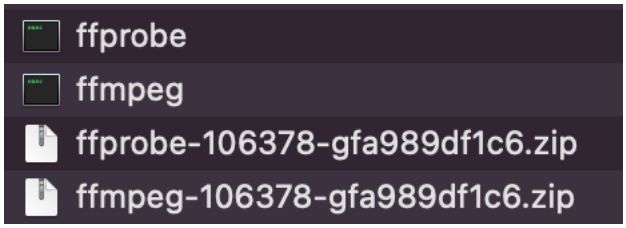
Installation of ffprobe and ffmpeg

To run ffprobe and ffmpeg, I need to install it first. We can head over to https://ffmpeg.org/download.html and download the packages. I am currently running on MacOS with M1 Apple Silicon. Therefore, I will download the one for Mac.



I will download as ZIP for both ffprove and ffmpeg.

Go to the downloads folder, unzip it, and it will show you two files named ‘ffprobe’ and ‘ffmpeg’



Next, we need to open the terminal.
Firstly, we need to go to the Downloads folder:
```
cd ~/Downloads
```

Then, we will display all the files in the Downloads folder:
```
ls
```

This is to check in the usr/local/bin exists:
```
ls /usr/local/bin
```

if it does not exist, create /usr/local/bin:
```
sudo mkdir -p /usr/local/bin
```

Moving on, we want to copy the binaries into that directory:
```
sudo cp ff* /usr/local/bin
```

As the Mac system does not allow us to run this as we downloaded it off from the internet, so we need to remove them from quarantine. To remove binaries from quarantine:
```
sudo xattr -dr com.apple.quarantine /usr/local/bin/ff*
```

Next, we need to check PATH:
```
echo $PATH
```

Now we are ready to run ffmpeg and ffprobe:
```
ffmpeg
ffprobe
```

Below are the screenshots when running ffmpeg and ffprobe on the terminal:

ffmpeg

```
rizfebriansyah@Febriansyahs-MacBook-Pro ~ % ffmpeg
ffmpeg version N-106378-gfa989df1c6-tessus  https://evermeet.cx/ffmpeg/  Copyright (c) 2000-2022 the
 FFmpeg developers
  built with Apple clang version 11.0.0 (clang-1100.0.33.17)
  configuration: --cc=/usr/bin/clang --prefix=/opt/ffmpeg --extra-version=tessus --enable-avisynth --
-enable-fontconfig --enable-gpl --enable-libaom --enable-libass --enable-libbluray --enable-libdav1d
 --enable-libfreetype --enable-libgsm --enable-libmodplug --enable-libmp3lame --enable-libmysofa --e
nable-libopencore-amrnb --enable-libopencore-amrwb --enable-libopenh264 --enable-libopenjpeg --enabl
e-libopus --enable-librubberband --enable-libshine --enable-libsnappy --enable-libsoxr --enable-libs
peex --enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvmaf --enable-libvo-amrw
benc --enable-libvorbis --enable-libvpx --enable-libwebp --enable-libx264 --enable-libx265 --enable-
libxavs --enable-libxvid --enable-libzimg --enable-libzmq --enable-libzvbi --enable-version3 --pkg-c
onfig-flags=--static --disable-ffplay
  libavutil      57. 24.101 / 57. 24.101
  libavcodec     59. 25.100 / 59. 25.100
  libavformat    59. 20.101 / 59. 20.101
  libavdevice    59.  6.100 / 59.  6.100
  libavfilter     8. 29.100 /  8. 29.100
  libswscale      6.  6.100 /  6.  6.100
  libswresample   4.  6.100 /  4.  6.100
  libpostproc    56.  5.100 / 56.  5.100
Hyper fast Audio and Video encoder
usage: ffmpeg [options] [[infile options] -i infile]... {[outfile options] outfile}...

Use -h to get full help or, even better, run 'man ffmpeg'
```

ffprobe

```
rizfebriansyah@Febriansyahs-MacBook-Pro ~ % ffprobe
ffprobe version N-106378-gfa989df1c6-tessus  https://evermeet.cx/ffmpeg/  Copyright (c) 2007
-2022 the FFmpeg developers
  built with Apple clang version 11.0.0 (clang-1100.0.33.17)
  configuration: --cc=/usr/bin/clang --prefix=/opt/ffmpeg --extra-version=tessus --enable-av
isynth --enable-fontconfig --enable-gpl --enable-libaom --enable-libass --enable-libbluray --
-enable-libdav1d --enable-libfreetype --enable-libgsm --enable-libmodplug --enable-libmp3lam
e --enable-libmysofa --enable-libopencore-amrnb --enable-libopencore-amrwb --enable-libopenh
264 --enable-libopenjpeg --enable-libopus --enable-librubberband --enable-libshine --enable-
libsnappy --enable-libsoxr --enable-libspeex --enable-libtheora --enable-libtwolame --enable
-libvidstab --enable-libvmaf --enable-libvo-amrwbenc --enable-libvorbis --enable-libvpx --en
able-libwebp --enable-libx264 --enable-libx265 --enable-libxavs --enable-libxvid --enable-li
bzimg --enable-libzmq --enable-libzvbi --enable-version3 --pkg-config-flags=--static --disab
le-ffplay
  libavutil      57. 24.101 / 57. 24.101
  libavcodec     59. 25.100 / 59. 25.100
  libavformat    59. 20.101 / 59. 20.101
  libavdevice    59.  6.100 / 59.  6.100
  libavfilter     8. 29.100 /  8. 29.100
  libswscale      6.  6.100 /  6.  6.100
  libswresample   4.  6.100 /  4.  6.100
  libpostproc    56.  5.100 / 56.  5.100
Simple multimedia streams analyzer
usage: ffprobe [OPTIONS] [INPUT_FILE]

You have to specify one input file.
Use -h to get full help or, even better, run 'man ffprobe'.
```

Definition of terms

Video Format (Container): A container, sometimes known as a wrapper, is a file that contains audio and video data as well as extra metadata. It's also known as a file extension since it's frequently encountered as a file name, such as .mov, .avi or .mp4.

Video Codec: A video codec is a piece of hardware or software that compresses and decompresses digital video in order to reduce file sizes and simplify storage and dissemination. Codecs are also employed to improve the playing quality of files.

Audio Codec: An audio codec is a hardware or software device that can encode or decode a digital audio stream. An audio codec is a computer programme that implements a method for compressing and decompressing digital audio data using a specific audio file coding standard such as WMA, MP3, DTS.

Frame Rate: Frame rate (FPS) is the rate at which individual still pictures, known as frames, are collected by a recording device and/or shown onto a screen.

Aspect Ratio: An aspect ratio is a proportionate connection between the width and height of a picture. It is usually written as two numerals separated by a colon, for example, 16:9, sixteen-to-nine.

Resolution: Resolution is a term used to describe the sharpness and clarity of an image. Picture resolution is usually expressed in pixels per inch (PPI), which refers to how many pixels are displayed per inch of an image.

Video Bit Rate: The amount of bits transferred or processed in a given unit of time is referred to as the bitrate. In other words, video bitrate refers to the amount of video data that is transferred at any given moment.

Audio Bit Rate: It specifies the speed at which audio bits are transported from one place to another. In simple terms, it measures and calculates the quantity of data delivered in a particular length of time.

Audio Channels: It is the pathway or communication channel by which a sound signal is conveyed from the player source to the speaker.

Firstly, we need to download the latest FFmpeg static build.

```
#this will download latest ffmpeg static build
exist = !which ffmpeg
if not exist:
  !curl https://johnvansickle.com/ffmpeg/releases/ffmpeg-release-amd64-static.tar.xz -o ffmpeg.tar.xz \
     && tar -xf ffmpeg.tar.xz && rm ffmpeg.tar.xz
  ffmdir = !find . -iname ffmpeg-*-static
  path = %env PATH
  path = path + ':' + ffmdir[0]
  %env PATH $path

!which ffmpeg
```

```
/usr/local/bin/ffmpeg
```

Need to install tabulate library by entering the command: pip install tabulate

This will be all the required imports needed:

```
from tqdm import tqdm
import subprocess as sp
import os
import json
import shlex

import tabulate
import pandas as pd
```

Afterwards, we will collect the media stream information for the videos. In this example below, it will be the Voyage of The Planet of Prehistoric Women:

Collecting Media Stream Information for Voyage to the Planet of Prehistoric Women (using ffprobe)

```
In [4]: !ffprobe -hide_banner ./Files/Voyage_to_the_Planet_of_Prehistoric_Women.mp4
```

```
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from './Files/Voyage_to_the_Planet_of_Prehistoric_Women.mp4':
  Metadata:
    major_brand     : mp42
    minor_version   : 0
    compatible_brands: mp42mp41
    creation_time   : 2021-08-02T19:26:10.000000Z
  Duration: 00:00:20.09, start: 0.000000, bitrate: 8337 kb/s
  Stream #0:0[0x1](eng): Video: hevc (Main) (hvc1 / 0x31637668), yuv420p(tv, bt709), 640x360 [SAR 1:1 DAR 16:9], 8038
kb/s, 29.97 fps, 29.97 tbr, 30k tbn (default)
    Metadata:
      creation_time   : 2021-08-02T19:26:10.000000Z
      handler_name    : ?Mainconcept Video Media Handler
      vendor_id       : [0][0][0][0]
      encoder         : HEVC Coding
  Stream #0:1[0x2](eng): Audio: mp3 (mp4a / 0x6134706D), 48000 Hz, stereo, fltp, 320 kb/s (default)
    Metadata:
      creation_time   : 2021-08-02T19:26:10.000000Z
      handler_name    : #Mainconcept MP4 Sound Media Handler
      vendor_id       : [0][0][0][0]
```

We will now set the variables for the required configurations for the reformatted videos.

This will be the variables which is required during the reformatting process:

In [9]:
```
labels = {
    'video_codec': 'h264',
    'resolution': '640x360',
    'aspect_ratio': '16:9',
    'frame_rate': 25,
    'video_bit_rate': 3700, #this is in kilobytes

    'audio_codec': 'aac',
    'audio_bit_rate': 256,
    'audio_channels': 'stereo'
}
```

In [10]:
```
video_format = 'mp4'

#this will be the video settings
video_codec = 'libx264'
resolution = '640x360'
aspect_ratio = '16:9'
frame_rate = 25
video_bit_rate = 3.7 #this is in megabytes

#this will be the audio settings
audio_codec = 'aac'
audio_bit_rate = 256
audio_channels = 2 #stereo
```

We now need to attain the metadata of the videos:

Now we we need to retrieve the metadata of the videos:

In [11]:
```
#we will now be storing the videos' required metadata in a nested dictionary
#we will assign the key variable in a format: data[vfile][video_metadata_label]

#["streams"][0] = video
#["streams"][1] = audio

def storing_metadata(vfile, json_video_metadata, data):

    try:
        #this will be the video aspect
        data[vfile] = {"video_codec": json_video_metadata["streams"][0]["codec_name"]}
        data[vfile]["resolution"] = str(json_video_metadata["streams"][0]["width"]) + 'x' + str(json_video_metadata["st
        data[vfile]["aspect_ratio"] = json_video_metadata["streams"][0]["display_aspect_ratio"]
        frames, seconds = json_video_metadata["streams"][0]["r_frame_rate"].split('/')
        data[vfile]["frame_rate"] = int(frames)/int(seconds)
        data[vfile]["video_bit_rate"] = int(json_video_metadata["streams"][0]["bit_rate"])/1000

        #this will be the audio aspect
        data[vfile]["audio_codec"] = json_video_metadata["streams"][1]["codec_name"]
        data[vfile]["audio_bit_rate"] = int(json_video_metadata["streams"][1]["bit_rate"])/1000
        data[vfile]["audio_channels"] = json_video_metadata["streams"][1]["channel_layout"]
    except:
        pass
```

It will be stored in a nested dictionary.
The format would be data[vfile][video_metadata_label]
Meaning that for example,
data['Voyage_to_the_Planet_of_Prehistoric_Women.mp4']['frame_rate']
will return the framerate of the file.

Now, we will display the video files in a table form.

Displaying the video files in a table

In [12]:
```
#we will now search through the video files and use the ffprobe command to retrive the videos' metadata
data = {}
path_to_files = "Files/"

for vfile in os.listdir(path_to_files):

    #this is exclusion of the reformatted videos under the newformat folder
    if 'newformat' not in vfile:
        video_path = "./" + path_to_files + vfile

        #this is where you run the terminal command to erase the temporary file (DS_Store). Afterwards, we will retriev
        os.system("find . -name '.DS_Store' -delete")
        video_metadata = sp.run(shlex.split(f'ffprobe -hide_banner -show_streams -loglevel error -of json {video_path}'
        json_video_metadata = json.loads(video_metadata)

        #using the helper function, store and save the data in a nested dictionary
        storing_metadata(vfile, json_video_metadata, data)

pd.DataFrame.from_dict(data)
```

We will first search through the video files and use the ffprobe command to retrive the videos' metadata. The acquired data was recovered as bytes using subproccesses and shlex. Moving on, we convert data to dictionary format with json.loads for simple retrieval. The acquired data gathered is stored and saved in a list.

We will now create a report [all_videos_issues.txt'] to look for issues with the video.

Now, we need to identify the issues with the video, and generate a report in .txt format

In [13]:
```
from tabulate import tabulate

with open('all_videos_issues.txt', 'w') as issues:

    issues.write("VIDEOS WITH INCORRECT SETTINGS"+ '\n' + '\n')

    for vfile in data.keys():

        #this variable is used to tabulate video settings data for printing and saving to a.txt file
        table = []

        #this is to check for video_format (container) type (.mp4)
        if vfile[-3:] != video_format:
            table.append(['video_format', vfile[-3:], video_format])


        issues.write("Video Name: " + vfile + '\n')
        issues.write('**********************************************************************\n')

        #search through and analyse all retrieved metadata settings
        for video_metadata_label in data[vfile].keys():

            #this is what happens if the existing video configurations do not correspond to the required ones
            if data[vfile][video_metadata_label] != labels[video_metadata_label]:

                current_setting = str(data[vfile][video_metadata_label])
                desired_setting = str(labels[video_metadata_label])
                table.append([video_metadata_label, current_setting, desired_setting])

        issues.write(tabulate(table, headers=['Metadata', 'Current setting', 'Desired setting']) + '\n\n\n')
```

Lastly, we need to reformat the videos to the required settings.

Now, we need to reformat the videos to the specified requirements

A new video file will be generated with '_formatOK' appended at the end of the video file name.
These new video files will be stored in a new directory too:
/Files/newformat/{newvideofilename}

1) The Hill Gang Rides Again
We need to change these metadata:

- video bit rate
- audio bit rate

```
In [14]: !ffmpeg -i ./Files/The_Hill_Gang_Rides_Again.mp4 -loglevel error -b:v {video_bit_rate}M -b:a {audio_bit_rate}k ./Files/
```

2) Cosmos War of The Planets
We need to change these metadata:

- frame rate
- aspect ratio
- resolution
- video bit rate
- audio bit rate

```
In [15]: ={resolution} -b:v {video_bit_rate}M -b:a {audio_bit_rate}k ./Files/newformat/Cosmos_War_of_the_Planets_formatOK.mp4 -y
```

3) Last Man On Earth 1964
We need to change these metadata:

- video format (container)
- video codec
- audio codec
- frame rate
- video bit rate
- audio bit rate