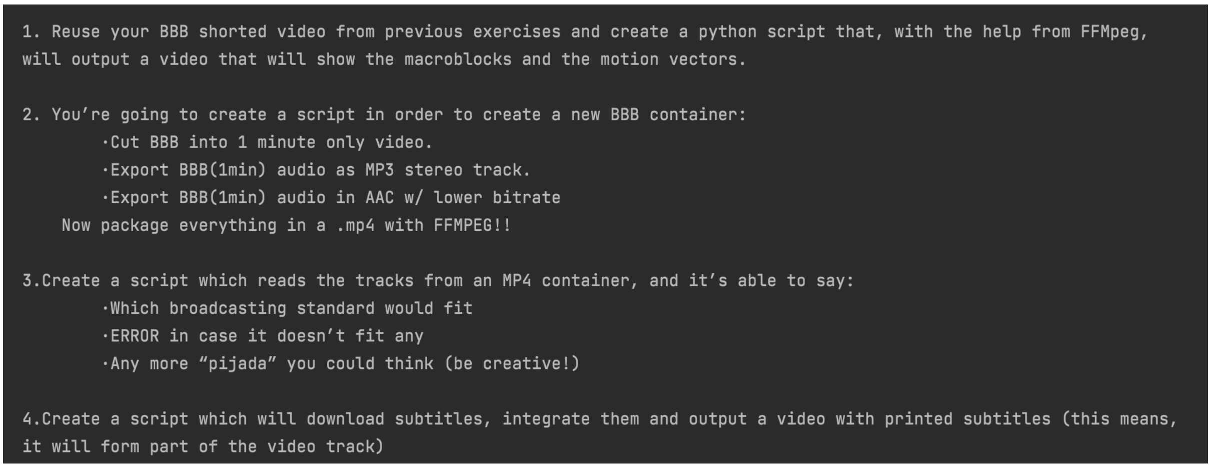
Paula Osés Noguero
NIA: 207310

S2: MORE PYTHON & FFMPEG

In this Lab we had to resolve 4 exercises using python and ffmpeg. For this I have created a Python project using PyCharm where you have a main() where you find the menu. Also a file called semi2.py where there is a class called seminar2 and inside there are the functions for each exercise.
You can see here the image of the menu,

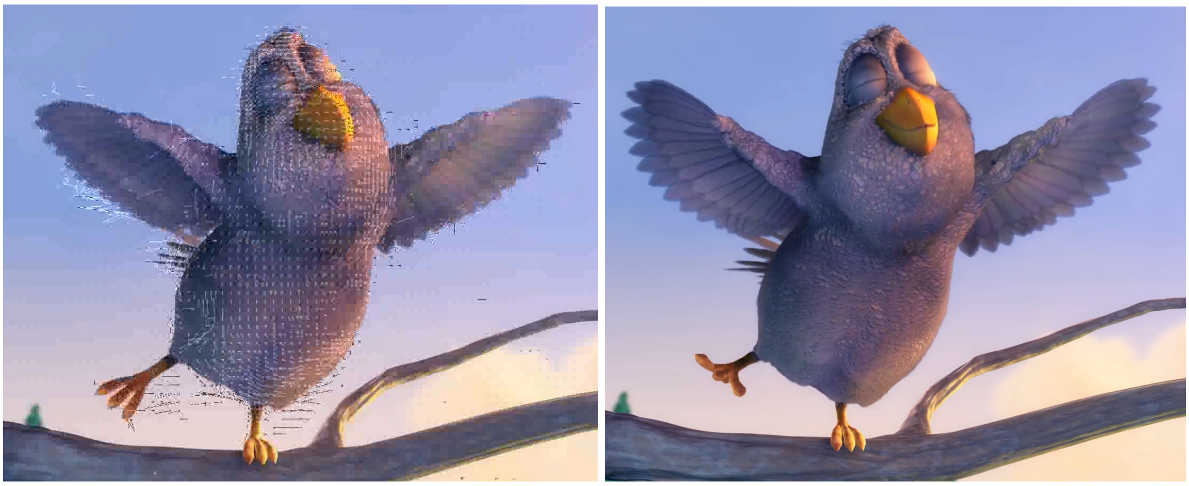


```
1. Reuse your BBB shorted video from previous exercises and create a python script that, with the help from FFMpeg,
will output a video that will show the macroblocks and the motion vectors.

2. You're going to create a script in order to create a new BBB container:
        ·Cut BBB into 1 minute only video.
        ·Export BBB(1min) audio as MP3 stereo track.
        ·Export BBB(1min) audio in AAC w/ lower bitrate
    Now package everything in a .mp4 with FFMPEG!!

3.Create a script which reads the tracks from an MP4 container, and it's able to say:
        ·Which broadcasting standard would fit
        ·ERROR in case it doesn't fit any
        ·Any more "pijada" you could think (be creative!)

4.Create a script which will download subtitles, integrate them and output a video with printed subtitles (this means,
it will form part of the video track)
```

Exercise 1:

For the first exercise we were asked to create a script which can cut N seconds from the BBB video, and then use this video to output it showing the macroblocks and motion vectors.
To do so I have used the following command,

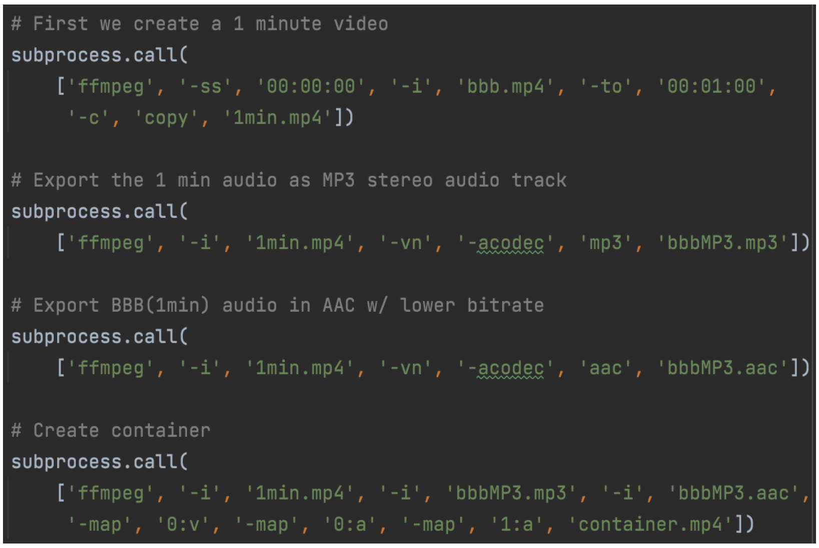
```
ffmpeg -flags2 +export_mvs -i 1min.mp4 -vf codecview=mv=pf+bf+bb  output.mp4
```



We can see that there are the macro blocks and the motions vector (image 1)

Exercise 2:

In this exercise we were asked to create a script which will, first, cut the video, then take the audio and export it into MP3 as well as AAC with lower bit rate and finally save all of this in a container. To do so I have used the following commands,

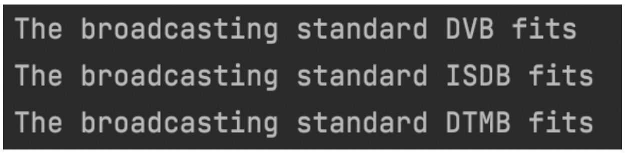


```
# First we create a 1 minute video
subprocess.call(
    ['ffmpeg', '-ss', '00:00:00', '-i', 'bbb.mp4', '-to', '00:01:00',
     '-c', 'copy', '1min.mp4'])

# Export the 1 min audio as MP3 stereo audio track
subprocess.call(
    ['ffmpeg', '-i', '1min.mp4', '-vn', '-acodec', 'mp3', 'bbbMP3.mp3'])

# Export BBB(1min) audio in AAC w/ lower bitrate
subprocess.call(
    ['ffmpeg', '-i', '1min.mp4', '-vn', '-acodec', 'aac', 'bbbMP3.aac'])

# Create container
subprocess.call(
    ['ffmpeg', '-i', '1min.mp4', '-i', 'bbbMP3.mp3', '-i', 'bbbMP3.aac',
     '-map', '0:v', '-map', '0:a', '-map', '1:a', 'container.mp4'])
```

The final output is a container (*conatiner.mp4*) that has the video and the two audios.

Exercise 3:

For this exercise we were asked to create a script that reads the container and tells which broadcasting standard would fit. For this I have created a dictionary for each broadcasting standard and then I compare it looking which codecs they accept. The result I get is that the container will fit the following broadcastings standards, DVB, ISDB and DTMB.



```
The broadcasting standard DVB fits
The broadcasting standard ISDB fits
The broadcasting standard DTMB fits
```

(In this exercise when you execute it is possible that it seems that it doesn't output anything since once it is executed it also prints the menu to do another operation).

Exercise 4:

This exercise consists of inserting the subtitles to the bbb video. For this first we download them from my github using this command,

```
requests.get("https://raw.githubusercontent.com/Paula022/Subtitles/main/big_buck_bunny.eng.srt")
```

And then we add them to the video using,

```
ffmpeg -i 1min.mp4 -vf subtitles=big_buck_bunny.eng.srt mysubtitledmovie.mp4
```

After this is executed you get the video with subtitles called *mysubtitledmovie.mp4*