To create a 30-minute binaural beats production that transitions from beta frequency to deep theta frequency and then back to beta frequency, you'll need to design a script that generates and modulates these frequencies over the specified time frame. Additionally, you'll need to handle the merging of the audio with an MP4 video file.

Given the provided files, I'll outline a plan and provide an updated version of the script to achieve your goals:

1. \*\*Generate Binaural Beats\*\*: Generate binaural beats that transition as specified.

2. \*\*Merge Audio with Video\*\*: Merge the generated audio with an MP4 video file.

Let's break down the tasks:

### Task 1: Generate Binaural Beats

We'll use the `BinauralBeatGenerator.py` and `BinauralBeatMixer.py` scripts to create and modulate the binaural beats. Here's a script that modifies your existing code to create the desired transitions.

#### Updated `BinauralBeatGenerator.py`:

```python

import numpy as np

from scipy.io.wavfile import write

def generate\_tone(frequency, duration, sample\_rate=44100):

t = np.linspace(0, duration, int(sample\_rate \* duration), endpoint=False)

return np.sin(2 \* np.pi \* frequency \* t)

def generate\_binaural\_beat(base\_freq, beat\_freq, duration, sample\_rate=44100):

left\_tone = generate\_tone(base\_freq, duration, sample\_rate)

right\_tone = generate\_tone(base\_freq + beat\_freq, duration, sample\_rate)

return np.vstack((left\_tone, right\_tone)).T

def create\_binaural\_transition(start\_freq, end\_freq, duration, sample\_rate=44100):

transition\_duration = duration // 2

frequencies = np.linspace(start\_freq, end\_freq, transition\_duration \* sample\_rate)

left\_channel = np.sin(2 \* np.pi \* frequencies \* np.arange(transition\_duration \* sample\_rate) / sample\_rate)

right\_channel = np.sin(2 \* np.pi \* (frequencies + 0.1) \* np.arange(transition\_duration \* sample\_rate) / sample\_rate)

return np.vstack((left\_channel, right\_channel)).T

def generate\_binaural\_beats\_production(output\_file, sample\_rate=44100):

beta\_freq = 20 # Beta frequency around 20 Hz

theta\_freq = 4 # Theta frequency around 4 Hz

total\_duration = 30 \* 60 # 30 minutes

transition\_duration = total\_duration // 6 # 5 minutes for each transition

# Generate the start in beta frequency

start\_beats = generate\_binaural\_beat(200, beta\_freq, transition\_duration, sample\_rate)

# Transition from beta to theta

transition\_down = create\_binaural\_transition(beta\_freq, theta\_freq, transition\_duration, sample\_rate)

# Main part in deep theta frequency

theta\_beats = generate\_binaural\_beat(200, theta\_freq, total\_duration - 2 \* transition\_duration, sample\_rate)

# Transition from theta to beta

transition\_up = create\_binaural\_transition(theta\_freq, beta\_freq, transition\_duration, sample\_rate)

# Combine all parts

full\_beat = np.concatenate((start\_beats, transition\_down, theta\_beats, transition\_up))

# Write to file

write(output\_file, sample\_rate, (full\_beat \* 32767).astype(np.int16))

if \_\_name\_\_ == "\_\_main\_\_":

output\_file = "binaural\_beats.wav"

generate\_binaural\_beats\_production(output\_file)

```

### Task 2: Merge Audio with Video

Next, we need to merge the generated binaural beats audio with an MP4 video file. We can use `ffmpeg` for this task. We'll modify the `main.py` to handle this step.

#### Updated `main.py`:

```python

import subprocess

import os

def merge\_audio\_video(audio\_file, video\_file, output\_file):

command = [

'ffmpeg',

'-i', video\_file,

'-i', audio\_file,

'-c:v', 'copy',

'-c:a', 'aac',

'-strict', 'experimental',

output\_file

]

subprocess.run(command, check=True)

if \_\_name\_\_ == "\_\_main\_\_":

audio\_file = "binaural\_beats.wav"

video\_file = "input\_video.mp4" # Replace with your video file path

output\_file = "final\_output.mp4"

# Ensure audio file is generated first

if not os.path.exists(audio\_file):

print(f"Error: Audio file {audio\_file} does not exist.")

elif not os.path.exists(video\_file):

print(f"Error: Video file {video\_file} does not exist.")

else:

merge\_audio\_video(audio\_file, video\_file, output\_file)

print(f"Successfully merged audio and video into {output\_file}.")

```

### How to Run

1. \*\*Generate Binaural Beats\*\*: Run `BinauralBeatGenerator.py` to generate the `binaural\_beats.wav` file.

```sh

python BinauralBeatGenerator.py

```

2. \*\*Merge Audio with Video\*\*: Run `main.py` to merge the generated audio with your video file.

```sh

python main.py

```

Ensure you have `ffmpeg` installed on your system to handle the audio-video merging. You can install it via most package managers or download it from the [official site](https://ffmpeg.org/download.html).

Feel free to ask if you need any more help or modifications!