TASK I

Why the audio feature chosen is appropriate to reflect the variation in the time series ?

Using the **frequency** (or pitch) of the sound to represent the variation in the stock price time series is an appropriate choice for several reasons:

1. **Direct Mapping of Magnitude:** Frequency is a fundamental property of sound that we perceive as pitch. By mapping higher stock prices to higher frequencies and lower prices to lower frequencies, we create a direct and intuitive correspondence between the magnitude of the time series and the perceived pitch of the sound. This allows the listener to easily "hear" the rises and falls in stock prices.

2. **Temporal Fidelity:** The approach generates a sound segment for each data point in the time series. This ensures that the temporal sequence of the stock prices is preserved in the audio. The pitch of the sound changes in lockstep with the stock price over time, providing an auditory representation of the time series' dynamic evolution.

3. **Accessibility:** Frequency is a readily manipulated parameter in synthetic sound generation. Sine waves, as used in the code, are simple to generate and control, making it straightforward to implement this mapping. The resulting sound is also generally easy for listeners to perceive and interpret changes in pitch.

Challenges faced in synchronizing audio and visuals.

Synchronizing the generated audio with the animated visualization presented some challenges, primarily related to timing and ensuring a one-to-one correspondence between the audio segments and the animation frames:

1. **Frame Rate and Audio Duration:** The animation was created with a specific frame rate (determined by the *interval* parameter in *FuncAnimation*). The audio was generated by assigning a fixed *duration_per_point* to each data point in the time series. To achieve synchronization, the total duration of the animation needed to match the total duration of the audio. This required careful coordination between the *interval* in *FuncAnimation* and the *duration_per_point* used in audio generation.

2. **Mapping Data Points to Frames and Audio Segments:** Each data point in the time series corresponds to both a frame in the animation (showing the time series up to that point) and a segment of audio (with a frequency determined by that data point). Ensuring that the *n*th data point's frequency is heard precisely when the animation displays the *n*th frame (or the transition to it) is crucial for effective synchronization. Any mismatch in the timing of these events would lead to a desynchronized output.

3. **Audio and Video Codecs and Containers:** Combining the generated audio (WAV format) and the animation (MP4) into a single synchronized video file required using a video editing library (*moviepy*). This step involves re-encoding the audio and video streams and combining them within a compatible container format (MP4). Compatibility issues with codecs or container formats could potentially cause problems during this merging process.

4. **Computational Resources:** Generating both the animation frames and the audio data, and then combining them, can be computationally intensive, especially for longer time series. This could lead to longer processing times and potential memory issues on systems with limited resources.