1. **Commitment:**
    a. Deserialization
        i. Reads a frequency from keyboard input (one key pressed at a time.)
    b. Physics / Wave Logic
        i. Produces a smooth sine waveform of the proper frequency by reading from a ROM.
        ii. Physics module outputs player positions for smooth transit between frequencies.
    c. Display
        i. Capable of rendering static player and collectable sprites.
        ii. Background is separated into two regions by the waveform profile from the physics module.
        iii. Title screen display available.
        iv. Renders quickly (responds in real-time to inputs).
    d. Audio
        i. Produces a frequency tone matching the input frequency.
    e. Game Logic
        i. The scenery moves forward steadily.
        ii. The player oscillates at the correct frequency.
        iii. Collectables are generated and displayed.

2. **Goal**
    a. Deserialization
        i. Reads frequencies from keyboard input; deals with multiple key presses.
    b. Physics / Wave Logic
        i. Produces smooth sine waveforms; capable of handling multiple frequencies at a time to produce a superimposed waveform.
        ii. Physics module handles smooth transitions between waveform types, for both the player's path and background display.
    c. Display
        i. Renders animated sprites.
        ii. Background contains an image (either rendered from memory or generated). (Waveform separates this background from the blue foreground).
        iii. Score and title screens available.
        iv. Renders quickly.
    d. Audio
        i. Canned music plays from memory.
    e. Game Logic
        i. Scenery moves forward at an increasing rate.
        ii. Player oscillates with the correct player path.
        iii. Collectables are generated and collected; score increases with collection.
        iv. Enemies are generated and kill the player.

3. **Stretch Goal**
    a. Deserialization
        i. Reads frequencies from keyboard input; parses all possible inputs into either a single frequency or a pair of frequencies.
    b. Physics / Wave Logic
        i. Depending on gameplay testing, can produce a variety of superimposed waveform types (e.g. triangle waves).
        ii. Flexible module; can be customized by game logic to produce any combination of wave types.
    c. Display
        i. Background moves slowly and/or has filters applied for game effects (e.g. directional blur).
        ii. Background and/or foreground effect generated using appropriate noise.
    d. Audio
        i. Simple music generation FSM plays music according to the input frequency.
    e. Game Logic
        i. Scenery moves forward at an increasing rate.
        ii. Player oscillates with the correct player path.
        iii. Collectables are generated and collected; score increases with collection.
        iv. Enemies are generated; player is killed by enemies.
        v. Generation of enemies and collectables is not hard-coded or fully deterministic, but reasonable spacing and patterns are enforced by the game logic.
        vi. Collectables and enemies move relative to the background.
        vii. A high score is recorded across plays and displayed on the ‘game over’ screen.