Assignment 2

Assignment 2 should be handed in before the **19th** of March. It consists of eight different problems. Each of these deals with a different approach to applying programming concepts in a musical context.

The first four exercises will be completed individually during class and handed in at the end of the session.

The last four exercises must be completed independently outside of class. For these, you must submit both your code and a short written reflection (minimum 300 words) describing your approach, the strategies you used, and what you learned.

The final submission should include:

1. **Code** for exercises 1–4 (done in class).
2. **Code** for exercises 5–8 (done independently).
3. A 300-word **reflection** on your process.

Components

1. Implement a **sound structure** that consists of two (or more) *oscillators* (of your choice) that are combined for a final output. The amplitude of each of the oscillators should be modulated by some kind of UGen, and the overall development should go from *dense* to *sparse to dense* again in terms of spectrum and loudness with a duration of at least 30 seconds.
2. **Multichannel expansion** is an interesting option for sound synthesis in SuperCollider. Implement a *SynthDef* that uses multichannel expansion for implementing **FM Synthesis** with at least 2 different carriers and 2 different modulation frequencies. The synth should be sequenced in time using **routines** where the FM amount starts subtly and progresses to saturation, then ends very quietly again (still slightly different from the original).
3. Create a **sound movement** that uses (at least) three different kinds of noises. All of these should be filtered in different ways (e.g., highpass, lowpass, or bandpass) and appear both individually and together during the implemented sound movement. You can choose either *Routines* or *Tasks* for the sequencing of the noise *SynthDefs*.
4. Create a **repeating sound process** in which rhythmic patterns are formed using a Pbind that combines randomness and sequential elements. Finally, all pitches should evolve either through Markov chains or using the **Pfsm** pattern.
5. Implement a **Pbind** process that uses **tendency masks** for synthesizing sounds in various ways. There should be separate masks for the frequency, amplitude, and *duration of events*. Finally, there should be at least two distinct layers audible, with moments when both are present simultaneously and others when each can be heard individually.
6. **Patterns** in SuperCollider can be useful for generating behaviour from abstract specifications. Implement a layered texture using **Ppar**

where at least **three independent pattern streams** interact. Each layer should have a **distinct character** (for example, sustained tones, rhythmic pulses, and sparse sounds). Use **pattern variables** to define each layer separately, then combine them. The layers should have different densities (notes per second) and occupy different frequency ranges.

7. **Shapes** in music are a powerful way to make things change over time. Implement a function that generates shapes (**envelopes**) that can be played with after being created. The function should contain at least three different methods for generating shapes.
8. Implement **a short musical composition** that sequences SynthDefs for the sound. It should be a sectional composition with at least **three distinct sections** that contrast in character. Each section should use different generative techniques (for example: one section for smooth motion, another with unpredictability, another with directed movement). You can choose between **Patterns** and **Routines**/**Tasks** (or a mix of both).