Assignment 2

Assignment 2 should be handed in before the **21st** of March. It consists of eight different problems. Each of these deals with a different approach of using concepts from programming in a musical context.

The **code should be handed in** as well as a small text with **a minimum of 300 words** explaining the approaches and the value these could have in a musical setting.

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 *sparse* to *dense* 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 4 different carrier and 4 different modular frequencies. The synth should be sequenced in time using routines where the FM amount starts with being subtle but progresses towards saturation and finally ends by being very quiet 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 (multiple filters) and appear both individually and in group during the implemented sound movement. You can choose either *Routines* or *Patterns* for the sequencing the noise *SynthDefs*.
4. Create a **repeating sound process** where *rhythmic patterns* are formed using a Pbind that binds together randomness and sequential

elements. Finally, all pitches should evolve according to either **Markov chains** or by using the **Pfsm** pattern.

5. Implement a **Pbind** process that uses **tendency masks** for synthesizing sounds in various ways. There should be different masks for *frequency*, *amplitude* and *duration of events*. Finally, there should be at least two different layers audible with moments where both exists at the same time and others where each can be heard individually.
6. **Patterns** in SuperCollider can useful to generate behavior on abstract specifications. Combining patterns is a useful way to create higher-level structures that can form a piece of music. Create a *pattern factory* that implements at least three different methods of generating patterns. The factory should be implemented with functions and/or events Additionally, write code that uses the factory to create different combination of music it is capable of generating.
7. **Shapes** in music are a powerful way of making things change in time. Implement a function that generates shapes (**envelopes**) that can be used both for synthesis and patterns and their movements in time. The function should contain at least three different methods for generating shapes as well as utility methods for testing them.
8. Implement **a short musical composition** that contains (at least) three sections with (at least) two layers running at the same time. The layers should be created in such a way (stored somewhere) that they can be put in a different order in time and their order should be different (random) each time the composition is played. Finally, there should be different kinds of layers with some being very **periodic** while others are very **noisy**.