Edit: Hierbij nog een stuk reflectie bij de update!

Om deze update te maken ben ik eigenlijk weer begonnen bij een savepoint waar ik eerder vastgelopen was. Na wat vragen te stellen in de practicum les lukte het me deze keer wel om een systeem met polymorphism te maken wat werkte zoals ik het wil. Er zit nu ook een mooi melodie opslag functie in en alles is gewoon een stuk beter.

Het programmeren ging erg soepel, ben tegen weinig problemen aangelopen die echt deal breakers waren. De paar lastige problemen waar ik echt niet uit kwam ben ik daarnaast met wat klasgenoten uitgekomen 😊

Originele reflectie:

Hoe ik dit project heb aangepakt is door eerst goed te brainstormen en te testen wat een goede class structuur zou zijn. Hierbij heb ik meerdere configuraties van inheritance en composition uitgeprobeerd. Ik ben uiteindelijk bij een composition based systeem gekomen aangezien inheritance mij niet liet doen wat ik wou, zonder losse synths te hardcoden. Ik heb echter wel erg mijn best gedaan om een manier te vinden om nuttig gebruik te maken van inheritance, omdat ik denk dat Mark en Cieska hier blij van worden, echter kon ik er nergens echt pluspunten voor vinden.

In eerste instantie was de waveform (zoals sine of triangle) namelijk een child van osc, echter kwam ik er snel achter dat als ik daarna de keuze moest maken in de constructor van de synth welke dit waveform gebruikt werdt dit niet echt mogelijk was.

Het is voor mij de bedoeling geweest alle code zo flexibel mogelijk te houden waardoor zo min mogelijk meerdere keren geschreven hoeft te worden. Uiteindelijk ben ik op een synthesizer gekomen die drie random gekozen waveforms combineert tot een sound. Mijn planning is hiervoor geweest om op 1 januari hieraan te beginnen, daarna heb ik hier de rest van het weekend aan gewerkt om daarna van zondag 3 januari ’s ochtends tot maandag 4 januari 12 uur ’s nachts één stuk door te werken om zoveel mogelijk af te krijgen.

Waar ik van baal is dat ik eigenlijk geen tijd heb gehad om tijdens het semester echt aan csd te werken. Dit kwam deels omdat productie een zware last had en ook deels omdat ik zelf al veel projecten en verplichtingen eromheen had gepland. Dit heeft ervoor gezorgd dat ik eigenlijk alles dit weekend zelf heb mogen uitvinden. Er was helaas, naast een paar werkende voorbeelden, niet veel documentatie te vinden over hoe alles werkt. Dit heeft mij een hoop tijd gekost om echt een goed begrip van te krijgen, en dan heb ik het voornamelijk over de jack implementatie. Gelukkig hebben klasgenoten mij wat hints willen geven.

Ik liep wel tegen een probleem aan waar als ik de derde waveform in de osc aanriep, dat hij dan opeens niet meer bij alle drie de voices de phase updatete. Maakte niet uit of het een blank class was of dat er wat in stond. Heb hier nog naar gegoogled en bij iedereen nagevraagd maar niemand wist het. Het compiled namelijk wel gewoon goed. Heb dit later maar opgelost door de formule gewoon binnen osc te doen en de class ff achterwege te laten.

Dingen zoals inheritance en pointers leren ging eigenlijk relatief snel. Wat mij had kunnen helpen om hier wel wekelijks aan te kunnen zitten is misschien niet drie blokken kiezen en zo nodig meerdere projecten en werk ernaast doen. Eigenlijk is gewoon een dagje programmeren per week ideaal namelijk.

Ik ben trots op wat ik neer heb kunnen zetten, alhoewel ik het jammer vind dat ik de laatste bug met de triangle er niet uit heb kunnen krijgen.