Opdracht 3 - Single sample sequencer

In de python repository https://github.com/ciskavriezenga/CSD_22-23 staan voorbeeldscripts in de map “csd2a/theorie/3_assignment”. Deze bevatten oefeningen en opdrachten.

Werk met deze scripts en leer ervan, maar voeg ze niet toe aan Git. In Git zet je zo veel mogelijk alleen je eigen werk, eventueel aangevuld met code van anderen, maar dan altijd met bronvermelding.

Ga dan met de voorbeeldscripts aan de slag: code lezen en begrijpen, denk per script na wat je als output verwacht, daarna pas het script uitproberen en vervolgens de opdrachten uitvoeren die in de comments in de code staan (EXERCISES). Maak jezelf de routine eigen dat je telkens eerst een verwachting hebt wat het script zal doen voordat je het uitvoert. Daarmee houd je zelf de regie over je creatieve proces in plaats van alleen te reageren op situaties.

Als je alle scripts hebt doorgewerkt ben je eraan toe om je eigen script te maken met meer functionaliteit zoals gebruikers-input en (later) meerdere samples.

Voeg de volgende functionaliteit(en) toe:

User input
“Bij het uitvoeren van het script wordt de default BPM aan de gebruiker getoond. Daarna krijgt de gebruiker de mogelijkheid om in de terminal een andere BPM in te voeren die vanaf dat moment gebruikt wordt.“

Denk hierbij ook aan het volgende: wat doet je script wanneer een gebruiker verkeerde input geeft? Wat is verkeerde input en met welke input kun je wel iets?

Time steps
De ritmische sequence is genoteerd in ‘sixteenth note timestamps’. Dit zijn geen echte tijd-waarden maar stappen (steps) ter grootte van 16de noten.
Zie de *code snippet* hieronder uit “06_oneSampleSequenceTime.py”:

```
# create a list with ‘note timestamps' in 16th at which we should play the sample
timestamps16th = [0, 2, 4, 8, 11]
```

In eerdere voorbeelden werd er gewerkt met note duraties, zie de *code snippet* hieronder uit “04_randomNoteDuration.py”

```
# create a list with possible note durations: sixteenth, eighth and a quarter note
noteDurations = [0.25, 0.5, 1]
```

Schrijf een functie genaamd “durationsToTimestamps16th” die als input een lijst van *note durations* verwacht en als output een lijst van *timestamps in zestienden* teruggeeft.
Bijvoorbeeld:

- input: [0.25, 0.5, 0.25, 0.5, 0.5, 1, 1]
- output → [0, 1, 3, 4, 6, 8, 12]

Gebruik deze functie in je code op de plek waar `timestamps16th` wordt aangemaakt.

Time stamps volgens BPM

De timestamps in zestienden worden omgerekend naar timestamps in tijdwaarden. Zie de *code snippet* hieronder.

```
# transform the timestamps16th to a timestamps list with time values
for timestamp in sixteenthTimestamps:
  timestamps.append(timestamp * sixteenthNoteDuration)
```

Schrijf een functie die als input een lijst met time stamps in zestienden én de BPM verwacht en als output een lijst met time stamps als tijdwaarden teruggeeft.
Gebruik deze functie in je code om de samples in een sequence af te spelen.

Optioneel

- Verwerk de volgende functionaliteit.
  Aan het einde van de loop wordt elke keer de ritmische sequence aangepast (bijv. 2 noten worden omgedraaid, achterstevoren, verkleinen / vergroten).
  Gebruik hiervoor een functie.

- Denk na over hoe je een sequence kunt afspelen waarin meerdere samples tegelijkertijd klinken. Hoe ziet de timestamp lijst er dan uit? Hoe refereer je aan welke sample afgespeeld moet worden? Implementeer dit.

- (**uitdagend**)
  De gebruiker kan de bpm ook aanpassen (invoer in de console) **tijdens** het afspelen van de beat. (versnellen / vertragen of nieuwe bpm invoeren)