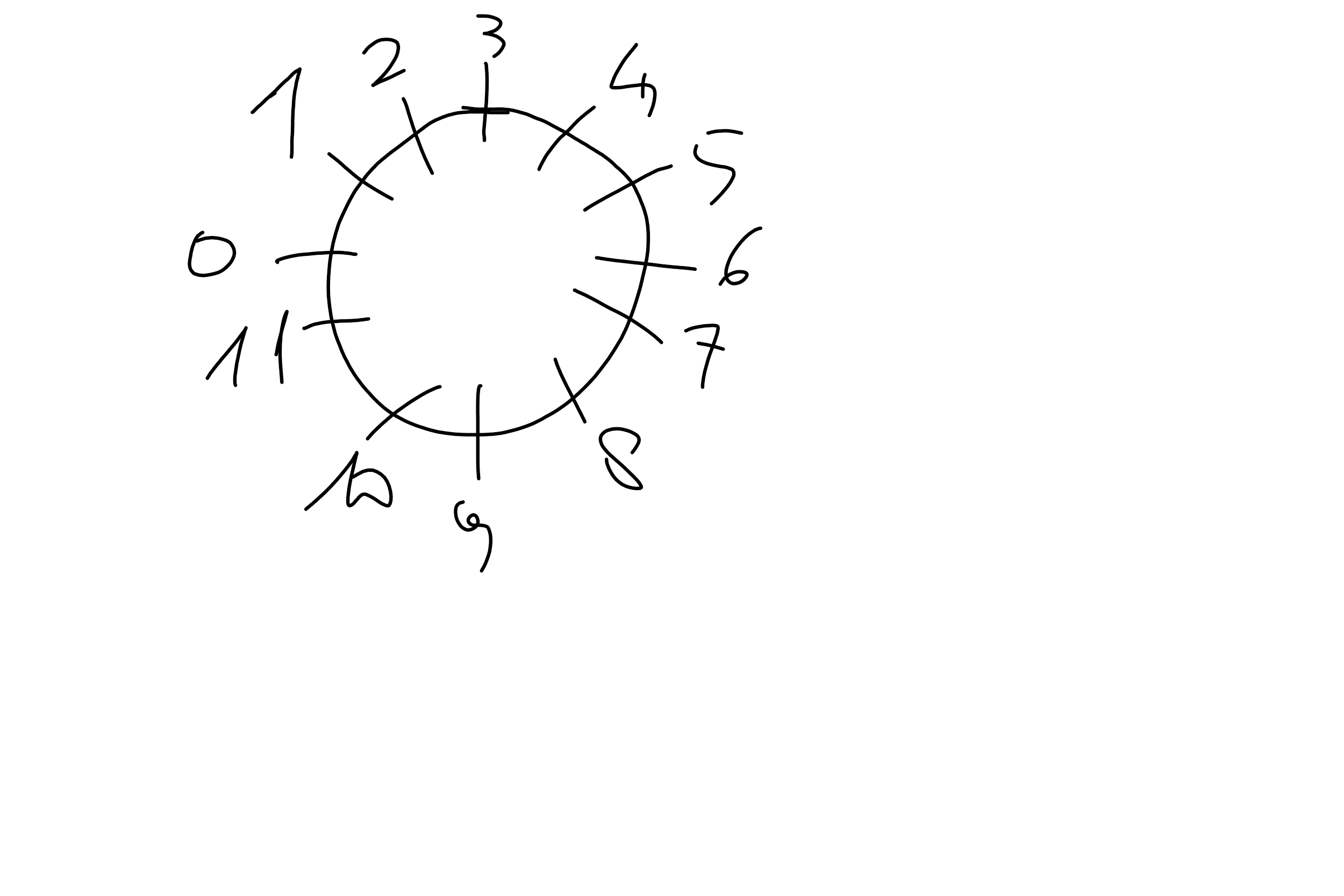
**Regole Luci**

Abbiamo 12 Luci disposte su una circonferenza equidistanziate come le ore su un orologio.

Numeri delle luci e corrispondenti note midi

----------------------------------------------------------------------------------------------------------

Tutte le luci sono direzionate verso il centro del cerchio, dove vi sono 3 damigiane le quali vengono illuminate.

L’intero lavoro sulle luci si basa sul far muovere le ombre o traslucenza delle damigiane che vengono proiettate sui muri della stanza, simulando un movimento di quest’ultima.

Inizialmemte avevo usato sc che leggendo il file MIDI di questa trascrizione mi dava la possibilità di applicare a determinate sezioni un relativo “arrangiamento per le luci”

<https://github.com/RandColors/project-guidance/blob/master/Drums/Konnakol/Konnakol_etude_01.scd>

Essendo estremamente laborioso, è diventato più agevole farlo a mano.

Qui il video dell’arrangiamento a mano delle luci,[midi da ableton a processing per DMX](https://youtu.be/ToO88orqzj0)



viene usato **Processing tramite il MIDI** per accedere al **DMX** tramite la sched**a Enntec DMX USB** <https://www.enttec.com/product/lighting-communication-protocols/dmx512/dmx-usb-interface/>

che controlla **2 dimmer pack da 6 canali**  <https://www.arenaluci.it/media/files/manuali/CO670L%20EN-R0.pdf>

Ho utilizzato dei suoni di campana tibetana campionati per sonificare le sequenze.

**Risultato estetico** -> <https://www.instagram.com/p/CUSfnLAAM2x/>

Nome del progetto : ENTIRE MIDI Lights Konnakol ACOUSTIC analisi.als

Regole per L’arrangiamento delle luci:

In processing semplicemente nota 0 = dmx.channel[0] etc..

**1) Luci singole**

es: corrispondenza tra nota MIDI (0..11) e luce

Scegli su quale nota deve essere applicato il ritmo.

Mettere un offset settabile sulla singola KonaWord o su una frase ad esempio

KonaWord(4,4,1, argSynth: \SingleSelector,  **offset: 0**)

// definisci il KonaTani

a = KonaTani.new(60, #["I4", "O", "O"], 5, argSynth: \ SingleSelector);

MIDIClient.init;

a.midiOut = MIDIOut.newByName("loopMIDI Port", "loopMIDI Port");

// crea la frase

(

~ph1 = KonaTime.newFrom([

~chatusra["[ 1, 3 ]"],

~chatusra["[ 1, 1, 1, 1 ]"],

~chatusra["[ 4 ]"],

~chatusra["[ 1, 1, 1, 1 ]"],

~chatusra["[ 1, 2, 1 ]"]

],a);

)

// aggiungi la frase

a.add(~ph1,**offset:0**)

Entrambi i problemi qui sotto sono legati al rendere cambiale realtime dei valori: o dei Synth o dell’0ffset.

**Problema 1**

Probabilmente per cambiare dinamicamente il valore serve utilizzare un KonaGenerator, e slegare il valore del **argSynth: \SingleSelector dal KonaTani,** questo per fari si che alla frase ~ph1 si possano abbinare altri comportamenti di luci definiti nello switch della KonaWord dei synths;

**Problema 2**

Es: mettiamo di voler prendere la frase ~ph1 e leggerla con il metodo .rout all’interno di una routine, per ottenere una lettura .inf e successivamente agire andando a cambiare il parametro Offset.

// make a variation --------------------------------------------------

m = a.gen.**changeOffset**(~ph1, **offset: 11.rand**);

(

d = Routine {

inf.do {

m.rout.embedInStream; // read ~ph1 indefinitely with random offset setted once or change it live

};

};

)

d.play(**a.clock**); // Importante mantieni il clock di a ovvero il KonaTani

**Problema 3 (lasciare gli Onset invariati e modificare la durata della nota)**

Andando a vedere KonaWord siccome dobbiamo gestire la durata della nota data dalla differenza tra noteON e noteOFF , andiamo ad agire sulla variabile:

var <**speed**; //The duration wait between syllables

Probabilmente devo anche qui come prima usare un KonaGenerator, per avere un metodo che modifichi tutto ciò che ci serve.

*Codice abbozzato:*

//Automated mapping for single light channel with note dur control

{\SingleSelector}{

rout = Routine {

word.size.do { |i|

if(word[i]!='-') {

if(i==0) // vel fissa a 127

{note = 0; vel = 127} // Luce indirizzo 0 etc..

{note = 1; vel = 127};

if(tani.midiOut != nil) { tani.midiOut.noteOn(0, note, vel); };

if(i==0) {

word[i].post;

} {

word[i].asString.toLower.post;

};

" ".post;

**(speed\*(1/4)).post; " ".post;**

**(speed/4).wait;**

if(tani.midiOut != nil) { tani.midiOut.noteOff(0, note, vel); };

**(speed\*(3/4)).wait;**

} {

word[i].post; " ".post;

speed.post; " ".post;

};

};

yieldAndReset(nil);

};

};

Problema 4

Bisogna settare un .wait relativo, se vogliamo dur\*1/4 la durata della nota (speed \*(1/4)).wait,

Poi bisogna aggiungere dopo il noteOff il tempo tolto sopra , altrimenti velocizza la lettura della frase, noi invece vogliamo solo accorciare la durata delle note.

Rendere questa una variabile un argomento settabile almeno per ogni frase.

**2) Regola Mapped on Syllables**

Ta = midi note 0

Ki = midi note 1

Di = midi note 2

Mi = midi note 3

words[9] = ['Da', 'Di', 'Gi', 'Na', 'Dom', 'Ta', 'Ka', 'Di', 'Mi'];

'Da', = midi note 0

'Di', = midi note 1

'Gi', = midi note 2

'Na', = midi note 3

'Dom', = midi note 4

'Ta', = midi note 5

'Ka', = midi note 6

'Di', = midi note 7

'Mi' = midi note 8

Questa è la parola più lunga disponibile.

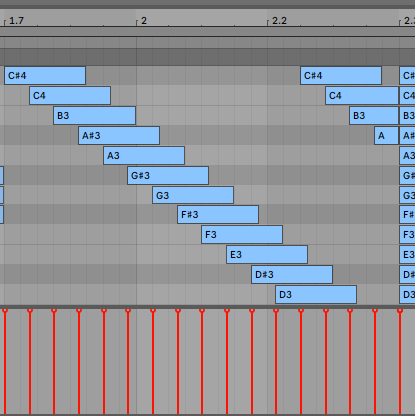
Altrimenti per concatenazione con accumulo di indici ed un % per iterare sempre tra 0 - 11

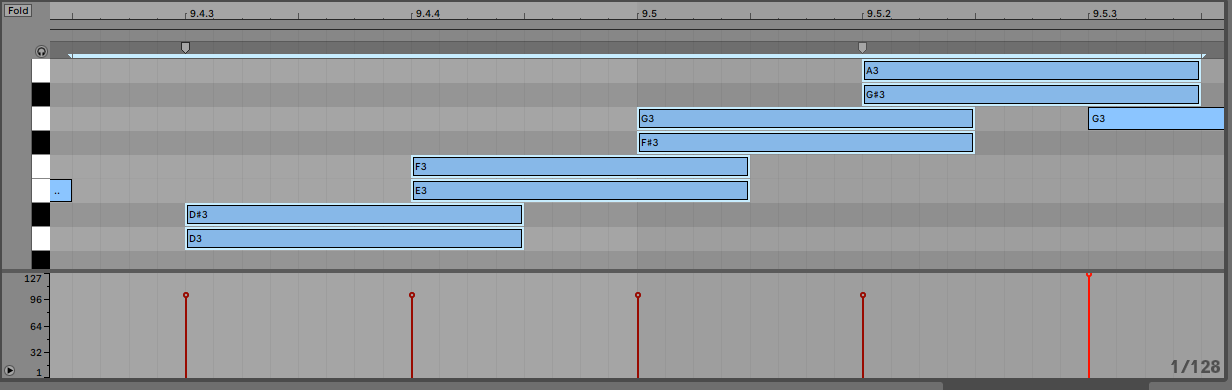
Parola 9 + parola 3 = parola 12 (9+3) -> dall’indice 0 all’indice 11

Altra opzione è semplicemente se uno mette una frase quella rimane tra % 0 – 11

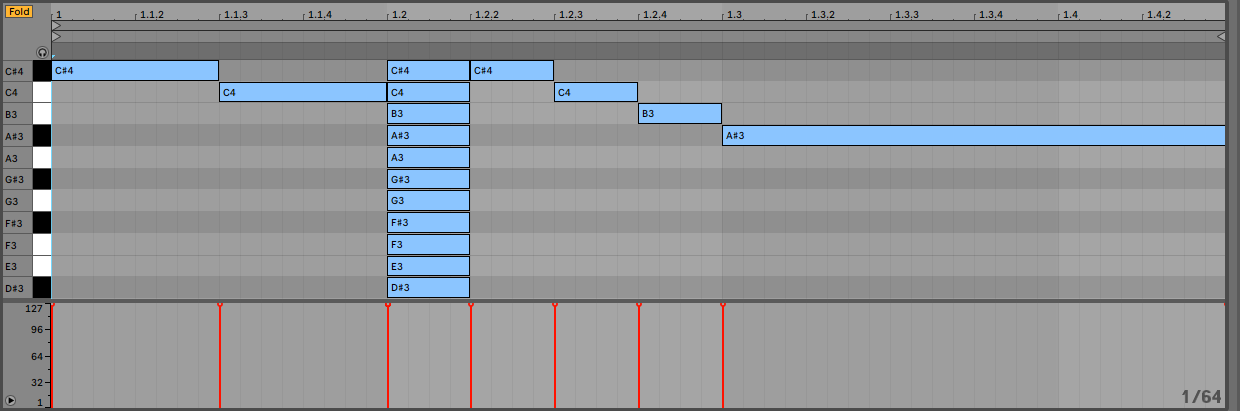
Come prima possibilità di ridurre la durata, in questo caso anche di allungarla

**Sovrapponi e cicla** su tutti i valori (0..11), troncando sull’ultima battuta.



**3 ) Più note contemporaneamente sovrapponi la dur e sposta l’offset**  

**4) Accento a Cluster anche nel mezzo di una frase**

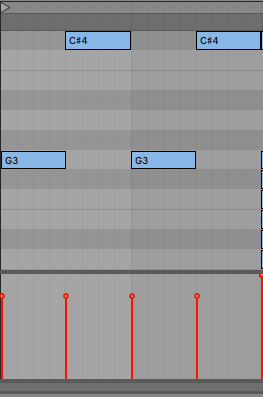


Questa opzione vince su tutte, ovvero deve essere possibile metterla su qualsiasi nota di una sequenza.

**5) alterna 2 o più note.**

È un arpeggiatore essenzialmente, scegli su che note vuoi arpeggiare e fallo in maniera % ciclica,

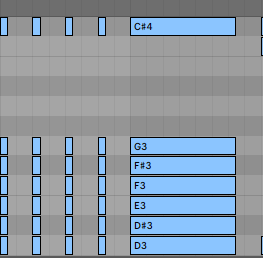
Decide anche qui se le note avranno durata < o > rispetto a dur

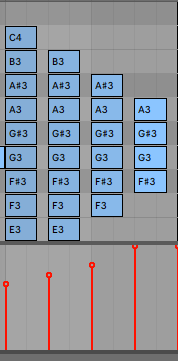


**6) alterna 2 o più note ma come gruppi**



**7) gruppi durate controllate**



**8) decresci il numero di note** di un’unità superiore ed una inferiore, andando verso una nota. 

**9) fai tante note** tipo 4 battute da 4/4 in sedicesimi e metti gli accenti in determinati punti senza interrompere la **direzione delle note**

In pratica la luce gira in cerchio ed ogni “tot” c’è un accento.

Da Aggiungere:

10) avere la possibilità di fare dei crescendo e diminuendo sulla velocity di intere frasi, decidendo solo il valore iniziale e quello finale, in maniera lineare ed exp potrebbe essere molto utile, sia per enfatizzare dei finali in “dissolvenza”, sia per fare delle frasi più gesture e vive.

Vi è inoltre anche un'altra luce che è un singolo faretto teatrale, da controllare per accenti, aggiungere quindi un canale dedicato, con un fattore di scaling diverso probabilmente 127 sarebbe una legnata in una stanza(verifica)

