



Chapitre N°7.) Le buzzer pour faire du bruit 😊



I. Caractéristiques techniques.....	130
1. Le son.....	130
2. Jouer une note.....	130
3. Comment le brancher ?	131
II. Produire du son : le buzzer.....	132
1. Avec RPi.GPIO.....	132
a. Montage.....	132
b. Manipulations	132
2. Avec gpiozero	132
a. Premières méthodes	132
b. Méthode beep.....	132
III. Jouons des notes maintenant	133
1. La théorie	133
a. La hauteur donne la mélodie	133
b. La durée donne le rythme	133
c. Les effets, timbres et accords.....	134
2. Les listes avec python	134
a. Créer une liste en python.....	134
b. Ajouter une valeur à une liste python.....	134
c. Afficher un item d'une liste	134
d. Compter le nombre d'items d'une liste.....	134
e. Trouver l'index d'une valeur.....	135
3. La pratique avec Au clair de la lune.....	135
a. Fréquences des notes.....	135
b. Création des tableaux.....	135
c. Durée d'un temps.....	136
d. Exemple d'utilisation	136
e. La mélodie par les 2TS SNEC	137
4. HappyBirthday : des croches et un peu d'anglais.....	137
a. Equivalence rythmique.....	137
b. Equivalence anglais/américain	137
c. Happy machin truc	138
d. On y va	138
5. Bien plus simple, finalement	138
a. MarioKart.....	139



I. Caractéristiques techniques

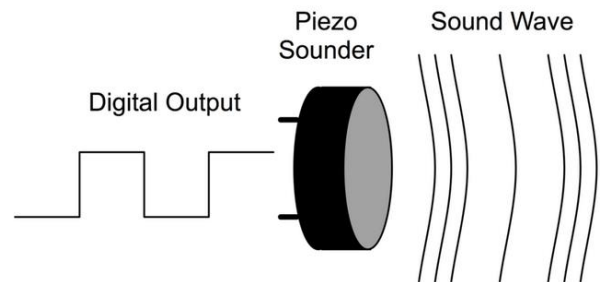
Un matériau piezo électrique est une substance qui produit un courant électrique lorsqu'il est déformé. Et inversement, lorsqu'une tension électrique est placée sur la substance, une déformation a lieu.

Cet effet est causé par des polarisations de molécules. En effet toute molécule est chargée, donc un bout est chargé plus négativement que l'autre. On appelle ceci un dipôle. On peut imaginer alors une orientation des atomes définie par des vecteurs. Dans un monocristal, tous ces vecteurs sont dans le même sens et direction. Au contraire, dans un polycristal, ces vecteurs vont dans tous les sens et directions.

Avec un signal carré en entrée, le une déformation suivie d'un retour à l'état normale va engendrer une oscillation :

1. Le son

D'un point de vue physique, un son est une énergie qui se propage sous forme de vibrations dans un milieu compressible (dans l'eau, dans l'air, dans les matériaux solides, mais pas dans le vide!).

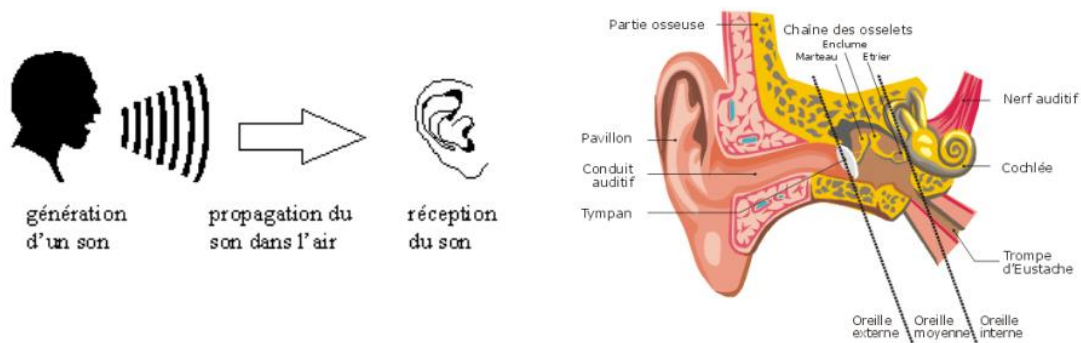


Lorsqu'on jette une pierre dans l'eau, on peut facilement observer le phénomène de propagation des ondes à la surface:



Lors de la diffusion d'un son dans un concert, c'est l'air qui permet sa transmission jusque nos oreilles. De même que l'exemple de l'eau illustré ci-dessus, les molécules d'air transmettent l'énergie et son donc un support pour le son.

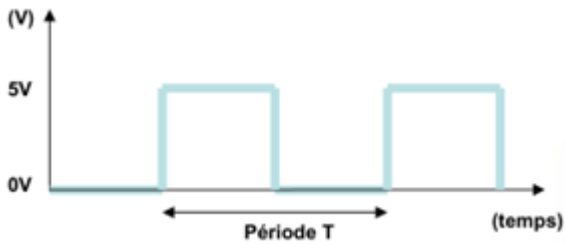
Pour être perçue, il doit y avoir un récepteur sensible. Chez l'homme, l'oreille possède une membrane (le tympan) capable de transmettre les informations de vibration en signaux nerveux jusqu'au cerveau, grâce au nerf auditif. De même, le microphone possède également une membrane permettant de transformer les déplacements de l'air en signaux électriques.



2. Jouer une note

Le buzzer se câble sur une sortie numérique en PWM (ou MLI en français) et le microcontrôleur lui envoie alors un signal périodique dont on fait varier la fréquence en fonction de la note que l'on désire jouer.

$$f = 1/T ; T=1/f ; f : \text{fréquence}; T : \text{période}$$



Exemple : le LA est un signal d'une fréquence f de 440 Hertz soit un signal qui varie 440 fois par seconde. La fréquence du DO est 262 Hz (octave 3) etc .. (<http://jeanjacques.dialo.free.fr/freque.htm>)

Quelle est la période d'un LA ? Combien de temps en microsecondes, le signal est à l'état haut ?

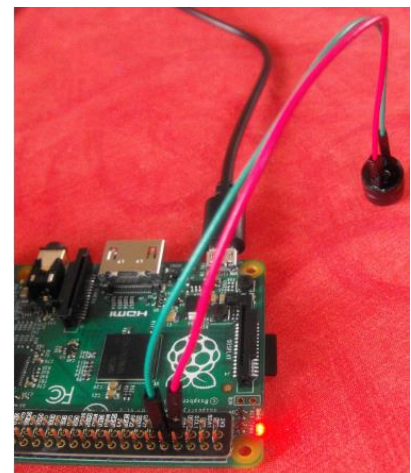
3. Comment le brancher ?

Il faut câbler le + sur la broche et le moins à la masse, pour les repérer c'est simple : c'est écrit dessus



Remarque :

Laisser l'autocollant pour ne faire trop de bruit





I. Produire du son : le buzzer

Le buzzer fait partie de la famille des **actionneurs**.

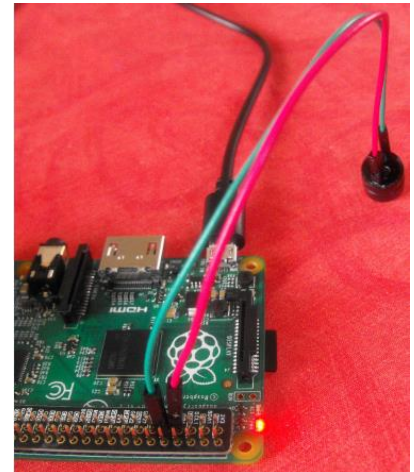
1. Avec RPi.GPIO

a. Montage

Mettez le + sur la broche 21 et la masse sur le –

b. Manipulations

Complétez le programme ci-dessous permettant d'émettre un son (infiniment) à la fréquence de 2 Hz. Surveillez le code avec try et nettoyez avec cleanup et terminez par mettre la broche à l'état bas (GPIO.output(buzzer,0)).



Remarque :

- Pour sortir, CTRL+C

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BCM)
buzzer = 21
....
```

2. Avec gpiozero

Une classe Buzzer existe dans gpiozero :

```
from gpiozero import Buzzer
```

a. Premières méthodes

Comme pour les leds, .on() et off() allume et éteint le buzzer

Refaire l'exercice précédent y compris l'exception où vous mettez un message de fin et éteignez le buzzer

b. Méthode beep

La syntaxe est :

```
beep(on_time=1, off_time=1, n=None, background=True)
```

Allume et éteint le buzzer

- **on_time (float)** – Nombre de secondes à on, valeur par défaut à 1 seconde
- **off_time (float)** – Nombre de secondes à off, valeur par défaut à 1 seconde
- **n (int)** – Nombre de clignotements; la valeur par défaut, None, signifie sans arrêt
- **background (bool)** – si True (valeur par défaut), démarre un processus (léger) en arrière-plan pour clignoter passe à la ligne suivante. Si c'est False, rend la main seulement lorsque le clignotement est terminé. Attention, dans le cas de la valeur par défaut du nombre de clignotements (**n (int)** =None) implique que cette méthode ne s'arrêtera jamais (sauf par l'appel de la méthode **on()**)



De l'exercice précédent, remplacez les méthodes on() et off() par beep()

II. Jouons des notes maintenant

1. La théorie

a. La hauteur donne la mélodie

Voici les fréquences de notes :

Note/octave	Fréquences des hauteurs (en Hertz)							
	0	1	2	3	4	5	6	7
Do	32,70	65,41	130,81	261,63	523,25	1046,50	2093,00	4186,01
Do#	34,65	69,30	138,59	277,18	554,37	1108,73	2217,46	4434,92
Ré	36,71	73,42	146,83	293,66	587,33	1174,66	2349,32	4698,64
Ré#	38,89	77,78	155,56	311,13	622,25	1244,51	2489,02	4978,03
Mi	41,20	82,41	164,81	329,63	659,26	1318,51	2637,02	5274,04
Fa	43,65	87,31	174,61	349,23	698,46	1396,91	2793,83	5587,65
Fa#	46,25	92,50	185,00	369,99	739,99	1479,98	2959,96	5919,91
Sol	49,00	98,00	196,00	392,00	783,99	1567,98	3135,96	6271,93
Sol#	51,91	103,83	207,65	415,30	830,61	1661,22	3322,44	6644,88
La	55,00	110,00	220,00	440,00	880,00	1760,00	3520,00	7040,00
La#	58,27	116,54	233,08	466,16	932,33	1864,66	3729,31	7458,62
Si	61,74	123,47	246,94	493,88	987,77	1975,53	3951,07	7902,13










Remarques :

- Ne lisez pas hastag mais dièse à côté des notes !
- Le dièse augmente la fréquence tandis que le bémol la diminue. Ainsi un Do# peut s'écrire également Ré bémol (Réb)
- Lorsqu'on dit que la fréquence du La est de 440 Hz c'est vrai mais à l'octave 3.
- Pour passer d'un octave à un autre, pour une même note il suffit de multiplier ou diviser par deux (ou un multiple)

Quelle est la fréquence d'un LA à l'octave 3 ? Dans la suite, on appellera cette note **la3**

b. La durée donne le rythme

On peut donner comme exemple :

- La noire  vaut 1 temps
- La ronde  = 4 noires 
- La blanche  = 2 noires 
- La croche  = 1/2 
- La double croche  = 1/4 



c. Les effets, timbres et accords

Pour terminer la présentation des notes, il faut mentionner :

- **Les effets** : c'est la manière dont laquelle la note est jouée. Vous avez sûrement entendu parlé de *piano*, *fortissimo*... Il y a également l'attaque douce, note piquées ...
- **Le timbre** : chaque instrument se différencie par son timbre qui, pour faire simple, dépend de la fréquence fondamentale et des harmoniques du signal. Le timbre fait qu'une même note sera perçue différemment d'un violon ou d'un piano
- **L'accord** : c'est le fait de jouer plusieurs notes à la fois comme par exemple pour la guitare.

2. Les listes avec python

a. Créer une liste en python

Pour créer une **liste** , rien de plus simple:

```
>>> liste = []
```

b. Ajouter une valeur à une liste python

Vous pouvez ajouter les valeurs que vous voulez lors de la création de la **liste python** :

```
>>> liste = [1,2,3]
>>> liste
[1, 2, 3]
```

Ou les ajouter après la création de la liste avec la méthode `append` (qui signifie "ajouter" en anglais):

```
>>> liste = []
>>> liste
[]
>>> liste.append(1)
>>> liste
[1]
>>> liste.append("ok")
>>> liste
[1, 'ok']
```

On voit qu'il est possible de mélanger dans une même liste des variables de type différent. On peut d'ailleurs mettre une liste dans une liste.

c. Afficher un item d'une liste

Pour lire une liste, on peut demander à voir l'index de la valeur qui nous intéresse:

```
>>> liste = ["a","d","m"]
>>> liste[0]
'a'
>>> liste[2]
'm'
```

Il est d'ailleurs possible de modifier une valeur avec son index

```
>>> liste = ["a","d","m"]
>>> liste[0]
'a'
>>> liste[2]
'm'
>>> liste[2] = "z"
>>> liste
['a', 'd', 'z']
```

d. Compter le nombre d'items d'une liste

Il est possible de compter le nombre d'items d'une liste avec la fonction `len` .

```
>>> liste = [1,2,3,5,10]
>>> len(liste)
5
```



e. Trouver l'index d'une valeur

La méthode `index` vous permet de connaître la position de l'item cherché.

```
>>> liste = ["a", "a", "a", "b", "c", "c"]
>>> liste.index("b")
```

3. La pratique avec Au clair de la lune



a. Fréquences des notes

En python, il est possible d'initialiser plusieurs variables sur une même ligne, comme par exemple :

```
# au lieu de
a = 1
b = 2
#il est possible de regrouper sur une même ligne
a,b = 1,2
```

Soit les notes suivantes :

do, re, mi, fa, sol, la, si = 32.7, 36.71, 41.20, 43.65, 49, 55, 61.74

En se basant sur le tableau ci-dessous, déclarez les variables do4, re4 ...si4

Fréquences des hauteurs (en Hertz)								
Noteloctave	0	1	2	3	4	5	6	7
Do	32,70	65,41	130,81	261,63	523,25	1046,50	2093,00	4186,01
Do#	34,65	69,30	138,59	277,18	554,37	1108,73	2217,46	4434,92
Ré	36,71	73,42	146,83	293,66	587,33	1174,66	2349,32	4698,64
Ré#	38,89	77,78	155,56	311,13	622,25	1244,51	2489,02	4978,03
Mi	41,20	82,41	164,81	329,63	659,26	1318,51	2637,02	5274,04
Fa	43,65	87,31	174,61	349,23	698,46	1396,91	2793,83	5587,65
Fa#	46,25	92,50	185,00	369,99	739,99	1479,98	2959,96	5919,91
Sol	49,00	98,00	196,00	392,00	783,99	1567,98	3135,96	6271,93
Sol#	51,91	103,83	207,65	415,30	830,61	1661,22	3322,44	6644,88
La	55,00	110,00	220,00	440,00	880,00	1760,00	3520,00	7040,00
La#	58,27	116,54	233,08	466,16	932,33	1864,66	3729,31	7458,62
Si	61,74	123,47	246,94	493,88	987,77	1975,53	3951,07	7902,13



b. Création des tableaux

Créez une liste appelée `AuClairDeLaLuneNotes` contenant toutes les notes de cette mélodie. On



utilisera do4, re4 ...

Créez une autre liste donnant durée de chaque note, les temps, le tempo. Une noire vaut 1 temps etc... Ce tableau s'appellera *AuClairDeLaLuneTempo*

c. Durée d'un temps

Italien	Français	Nb de pulsations par minute
Largo	Large	40 - 60
Lento	Lent	52 - 68
Adagio	À l'aise	60 - 80
Andante	Allant	76 - 100
Moderato	Modéré	88 - 112
Allegretto	légèrement allègre	100 - 128
Allegro	allègre (joyeux)	112 - 160
Vivace	Vif	120 - 140
Presto	Pressé, rapide	140 - 200
Prestissimo	Très rapide	140 - 200

Le tempo est la vitesse à laquelle les notes vont être jouées. Dans la musique électronique on parle de BPM (Battements par minute)

- Si une noire vaut 60, quelle est la durée en seconde de la première note d'au clair de la lune ?
Même question pour noire=180
- Déclarez une variable *noire* avec le tempo de 180 au début du programme en python. Déclarez une variable *TempsNote* en fonction de *noire* qui donne le temps d'une note.

d. Exemple d'utilisation

```

from gpiozero import TonalBuzzer #Buzzer pour jouer des notes
from time import sleep
from gpiozero.tones import Tone
noire = 300
TempsNote = 60/noire

buzzer = TonalBuzzer (21)

buzzer.play (Tone (frequency=440)) # utilisation de la fréquence La3
sleep (1*TempsNote) # temps d'une note noire

buzzer.stop () # arrêt et attente pour avoir l'impression d'une attaque
sleep (0.25*1*TempsNote) # 25% d'attente de la note précédente

buzzer.play (Tone (frequency=220)) # La2 une croche
sleep (0.5*TempsNote)

buzzer.stop ()
sleep (0.25*0.5*TempsNote) # 25% d'attente de la note précédente

buzzer.play (Tone (frequency=261.63)) # Do3 une noire
sleep (TempsNote)

buzzer.stop ()

```



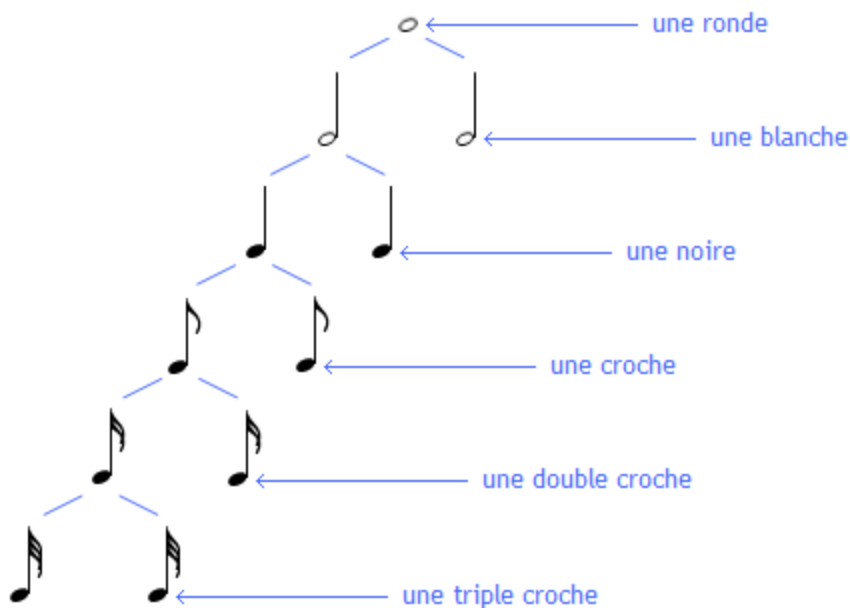
Testez ce code après avoir mis le tempo (=noire) à 200 bpm

4. La mélodie par les 2TS SNEC

Maintenant il faut jouer toute la mélodie. Pour cela utiliser une boucle for avec les deux tableaux précédemment créés *AuClairDeLaLuneNotes* et *AuClairDeLaLuneTempo*. A noter que leur taille est `len(AuClairDeLaLuneNotes)`

5. HappyBirthday : des croches et un peu d'anglais

a. Equivalence rythmique



b. Equivalence anglais/américain

Il est facile de trouver sur internet des partitions toutes faites mais elles sont toujours en anglais.

La notation anglaise utilise les lettres en commençant par le « la » : A= la, B= si, C= do, D= ré, E= mi, F= fa et G= sol

En ce qui concerne la notation des octaves, il faut ajouter un : la3 s'écrit A4.

Soit C4 est un do à l'octave 3 :





c. Happy machin truc Happy Birthday to You

Patty Hill Mildred J. Hill



Quel est, en anglais la première note ? Quel est le tempo ?

d. On y va ...

Voici le début du programme :

```
from gpiozero import TonalBuzzer
from gpiozero.tones import Tone
from gpiozero import TonalBuzzer #Buzzer pour jouer des notes
from time import sleep
from gpiozero.tones import Tone
buzzer = TonalBuzzer(21,octaves=4) #4 octaves au dessous ou en dessous

do, re, mi, fa, sol, la, si = 32.7, 36.71, 41.20, 43.65, 49.55, 61.74
do4, re4, mi4, fa4, sol4, la4, si4 = 4*do, 4*re, 4*mi, 4*fa, 4*sol, 4*la, 4*si
A3, B3, C3, D3, E3, F3, G3 = 4*la, 4*si, 4*do, 4*re, 4*mi, 4*fa, 4*sol
A4, B4, C4, D4, E4, F4, G4 = 8*la, 8*si, 8*do, 8*re, 8*mi, 8*fa, 8*sol

#une croche = 0.5 une double croche 0.25 noire =1 blanche =2
noire = 150 #BPM
TempsNote = 60/noire

HappyBirthdayNotes =
[_, _, _, G3, C4, B3, G3, G3, A3, G3, D4, C4, G3, G3, G4, E4, C4, B3, A3, F4, F4, E4, C4, D4, B3]
HappyBirthdayTempo =
[_, _, _, 1, 1, 2, 0.5, 0.25, 1, 1, 1, 2, 0.5, 0.25, 1, 1, 1, 1, 1, 0.5, 0.25, 1, 1, 1, 2]
```

Complétez les deux premiers éléments de la liste HappyBirthdayNotes et de HappyBirthdayTempo

Testez cette nouvelle mélodie

6. Bien plus simple, finalement

Au lieu de devoir calculer la fréquence, le module gpiozero le fait tout seul !!!

Exemple :

```
>>> from gpiozero import TonalBuzzer
>>> from gpiozero.tones import Tone
>>> b = TonalBuzzer(17)
>>> b.play(Tone(frequency=220.0)) # Hz
>>> b.play("A4")
```

Le code précédent devient donc (notez les chaînes de caractères) :

```
from gpiozero import TonalBuzzer #Buzzer pour jouer des notes
from time import sleep
buzzer = TonalBuzzer(21,octaves=4)
```



```
noire = 150 #BPM
TempsNote = 60/noire
```

```
HappyBirthdayNotes =
["G3", "G3", "A3", "G3", "C4", "B3", "G3", "G3", "A3", "G3", "D4", "C4", "G3", "G3", "G4", "E4",
"C4", "B3", "A3", "F4", "F4", "E4", "C4", "D4", "B3"]
HappyBirthdayTempo =
[0.5, 0.25, 1, 1, 1, 2, 0.5, 0.25, 1, 1, 1, 2, 0.5, 0.25, 1, 1, 1, 1, 1, 0.5, 0.25, 1, 1, 1, 2]

for i in range(len(HappyBirthdayNotes)):
    print(HappyBirthdayNotes[i])
    buzzer.play(HappyBirthdayNotes[i])
    sleep(HappyBirthdayTempo[i]*TempsNote)
    buzzer.stop()
    sleep(0.3*HappyBirthdayTempo[i]*TempsNote)
```

Testez ce code

7. MarioKart

Voici le code :

```
MarioNotes=["E7", "E7", "0", "E7", "0", "C7", "E7", "0", "G7", "0", "0", "0",
"G6", "0", "0", "0", "C7", "0", "0", "G6", "0", "0", "E6", "0", "0", "A6", "0", "B6", "0", "A6#", "
A6", "0", "G6", "E7", "G7", "A7", "0", "F7", "G7", "0", "E7", "0", "C7", "D7", "B6", "0", "0",
"C7", "0", "0", "G6", "0", "0", "E6", "0",
"0", "A6", "0", "B6", "0", "A6#", "A6", "0", "G6", "E7", "G7", "A7", "0", "F7", "G7", "0", "E7", "
0", "C7", "D7", "B6", "0", "0" ]

MarioTempo=[0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75,
0.75,
0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75,
0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 1, 1, 1, 0.75, 0.75, 0.75, 0.75,
0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75,
0.75, 0.75, 0.75,
0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 1, 1, 1,
0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75]

from gpiozero import TonalBuzzer #Buzzer pour jouer des notes
from time import sleep
buzzer = TonalBuzzer(21, octaves=4)

noire = 600 #BPM CA VA VITEEEEEEEEEEEEEEE
TempsNote = 60/noire

for i in range(len(MarioNotes)):
    _____
    _____
    _____
    _____
    _____
```

Complétez la fin du programme afin de gérer les temps morts sans note repérées par des "0"



BTS SN – EC

Séquence : Fabrication électronique avec Raspberry et Python

07-Les capteurs-actionneur : capteur ultrason

