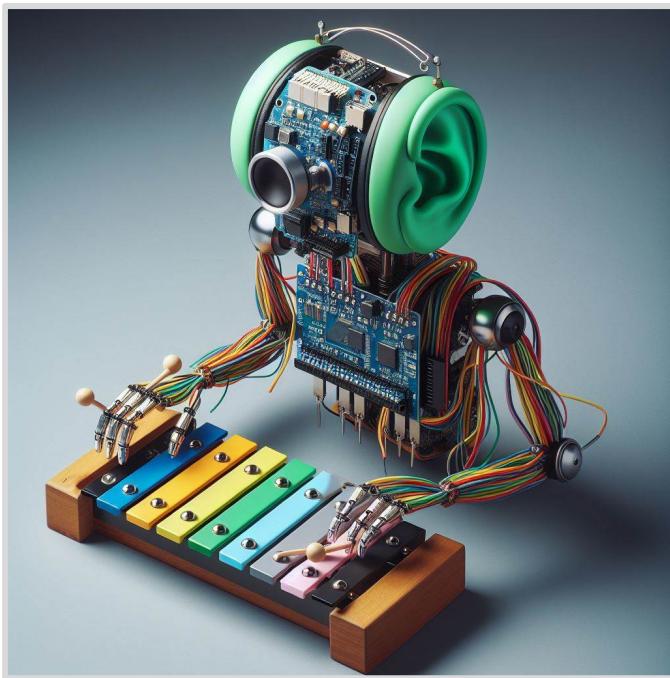


Το 1^ο Δημοτικό Σχολείο Παπάγου παρουσιάζει την:



...ένα ρομπότ που ακούει και παίζει μουσική!

Για τον 6^ο Πανελλήνιο Διαγωνισμό της ΕΛΛΑΚ

Σχ. Έτος: 2023-2024 - Τμήμα: ΣΤ₁

Εκπ/κός: Αναστασία Κωνσταντέλου

Πληροφορίες για τις εικόνες του εξώφυλλου



Το **Λογότυπο** του έργου δημιουργήθηκε αυτόματα με:

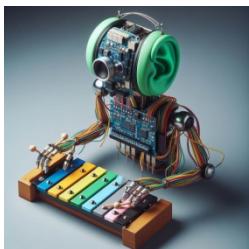
<https://www.design.com/maker/logo>

και επεξεργάστηκε επιπλέον από εμάς με το πρόγραμμα επεξεργασίας εικόνας GIMP.

Χρησιμοποιήσαμε επίσης:

<https://www.fontspace.com/please-write-me-a-sing-font-f18348>

Η παρακάτω εικόνα είναι προϊόν τεχνητής νοημοσύνης:



<https://www.bing.com/>

Keywords: robot with Arduino and one green human ear and 2 hands with four fingers each placed one hand opposite the other, listening to music and playing it to a xylophone

Πώς προέκυψε το όνομα Κυμώ

Η Αρχαία Ελληνική μυθολογία προσφέρει πλούτο μύθων και ονομάτων που μπορεί κανείς να αντλήσει έμπνευση, ακόμα και σήμερα, χιλιάδες χρόνια μετά, στην εποχή της «τεχνητής νοημοσύνης». Επί τούτου λοιπόν, αναζητήσαμε ελληνικό όνομα. Επιπλέον, τονίζουμε ότι το όνομα ΚΥΜΩ περιέχει δύο διακριτά ελληνικά γράμματα, το ΥΨΙΛΟΝ (Y-grec!) και το ΩΜΕΓΑ. Η κατασκευή μας έχει να κάνει με διαχείριση κυμάτων (ηχητικών και ηλεκτρομαγνητικών).

Αναζητήσαμε πληροφορίες για την Νηρήδα Κυμώ στην Wikipedia και στο Chat GPT.

Προέκυψαν, μεταξύ άλλων, οι παρακάτω πληροφορίες:

[Wikipedia](#): Η Κυμώ είναι μία Νηρήδα, η οποία ζει και κυκλοφορεί στα βάθη της θάλασσας. Είχε την ικανότητα να αιχμαλωτίζει ναυτικούς. [Το Ρομπότ Κυμώ έχει την ικανότητα να «αιχμαλωτίζει» ήχους και υπέρυθρη ακτινοβολία (το τηλεχειριστήριο)].

[Chat GPT](#): Η αρχαία ελληνική μυθολογία είναι ένας πλούσιος και πολυεπίπεδος κόσμος με πολλές εκδοχές και εκδοχές των ιστοριών. Ο μύθος της Νηρήδας Κυμώς μπορεί να διαφέρει ανάλογα με την πηγή ή την ερμηνεία. Οι μύθοι είναι συχνά υποκείμενοι σε ποικίλες εκδοχές και διασταυρώσεις, και μερικές φορές η ερμηνεία τους μπορεί να είναι ανοιχτή σε διαφορετικές ερμηνείες ή ερμηνευτικές προσεγγίσεις.

Συντελεστές του έργου:

Μαθητές:

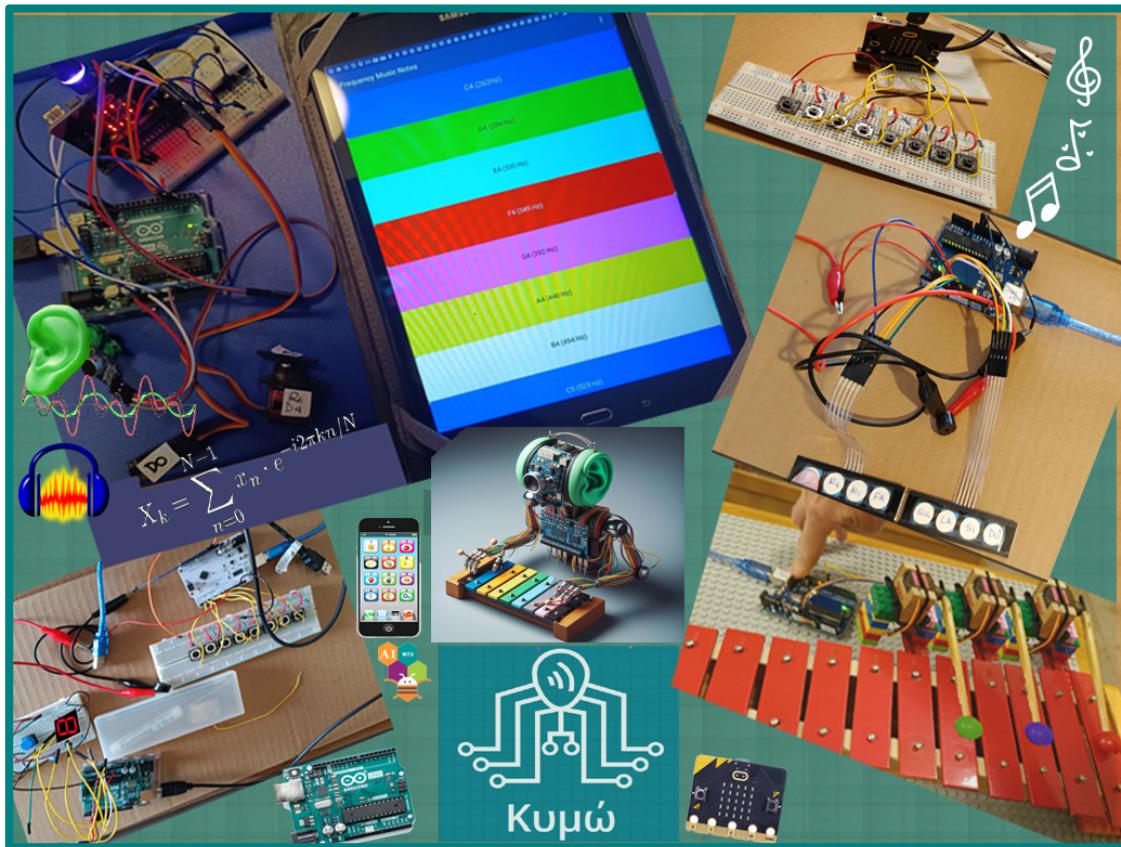
ΑΘΗΤΑΚΗ	ΝΑΤΑΛΙΑ
ΑΛΜΠΑΤΙΧΙ	ΓΙΑΣΕΡ
ΒΟΓΑΤΣΑΣ	ΕΥΑΓΓΕΛΟΣ
ΓΙΑΝΝΑΡΑΣ	ΘΕΟΦΑΝΗΣ
ΓΚΟΥΜΑ	ΜΑΡΙΑ
ΔΙΑΜΑΝΤΟΠΟΥΛΟΣ ΜΗΛΙΩΡΗΣ	ΑΓΓΕΛΟΣ
ΖΑΦΕΙΡΗΣ	ΧΑΡΑΛΑΜΠΟΣ
ΚΑΛΑΪΤΖΙΔΗ	ΗΛΙΑΝΑ-ΕΛΙΣΑΒΕΤ
ΚΙΤΣΟΥ	ΕΛΕΝΗ
ΚΟΝΤΟΓΙΑΝΝΗΣ	ΚΩΝΣΤΑΝΤΙΝΟΣ ΜΑΡΙΟΣ
ΛΑΒΔΑΣ	ΙΩΑΝΝΗΣ
ΛΙΒΑΝΟΥ	ΕΛΙΣΑΒΕΤ ΚΑΛΛΙΡΡΟΗ
ΜΑΚΡΗΣ ΜΑΚΡΙΔΗΣ	ΛΑΖΑΡΟΣ
ΜΑΝΤΖΑΡΑ	ΦΙΛΙΩ
ΠΑΛΑΙΓΕΩΡΓΙΟΥ	ΠΕΤΡΟΣ
ΠΑΝΑΓΩΤΑ	ΔΑΝΑΗ
ΠΑΝΑΓΩΤΑ	ΕΛΛΗ ΙΩΑΝΝΑ
ΠΑΠΑΚΩΝΣΤΑΝΤΙΝΟΠΟΥΛΟΥ	ΑΙΚΑΤΕΡΙΝΗ
ΡΟΒΟΛΗ	ΜΕΛΙΝΑ ΜΑΓΔΑΛΗΝΗ
ΣΑΡΙΑΤ ΠΑΝΑΧΙ	ΠΕΡΣΕΑΣ
ΤΑΤΣΗΣ	ΝΙΚΟΛΑΟΣ
ΤΣΙΟΥΝΗΣ	ΟΔΥΣΣΕΑΣ
ΧΡΥΣΟΣΠΑΘΗ	ΘΕΟΔΩΡΑ

Εκπαιδευτικός: **Κωνσταντέλου Αναστασία, ΠΕ86**

Ευχαριστούμε τη διευθύντρια του Σχολείου, κα Αγγελική Αναστασοπούλου, καθώς και την δασκάλα του τμήματος ΣΤ1, κα Ελένη-Ανδριανή Οικονόμου.

Ευχαριστούμε ιδιαίτερα την κα Λίτσα Ράδου, μουσικό του σχολείου για την τεχνογνωσία και την ηθική υποστήριξη.

Γενική περιγραφή του έργου:



Η **Κυμώ** βασίζεται σε τρία βασικά συστήματα:

	<p>Μία εφαρμογή για κινητό τηλέφωνο, υλοποιημένη με MIT App Inventor, που παράγει τις νότες C4, D4, E4, F4, G4, A4, B4, C5 με συγκεκριμένα χρώματα.</p>
	<p>Στο Arduino, που «ακούει» μέσω μικροφώνου, εφαρμόζει Fast Fourier Transform (FFT) στο σήμα, αποκωδικοποιεί την νότα που ακούει και δίνει εντολή σε ένα από τα 8 Servo motors να «χτυπήσουν» πάνω στο μεταλλόφωνο.</p>
	<p>Στο micro:bit, που συνδέεται με το Arduino μέσω I2C και λειτουργεί ως οπτική διεπαφή της Κυμώς. Δείχνει στην οθόνη 5X5 τη διεθνή ονομασία της νότας με γράμμα και με ένα RGB Led φαίνεται το χρώμα της νότας στην εφαρμογή κινητού τηλεφώνου.</p>

Δομή και Σύνθεση του Έργου

(Σειρά μαθημάτων και σύνθεση των επιμέρους τμημάτων του Robot «Κυμώ»)



- Αναβοσβήνω led (ακροδέκτες- *digital write*)
- Παίζω νότες, απεικονίζω στο matrix 5x5
- Παίζω μελωδίες
- Ανάβω led με αυξανόμενη ένταση φωτός (αριθμητικά)
- Παίζω μελωδίες με αυξομείωση έντασης ήχου (αριθμητικά)
- Ανάβω led με αυξανόμενη ένταση φωτός (με μεταβλητή- *analog write*)
- Παίζω μελωδίες με αυξομείωση έντασης ήχου (με μεταβλητή)
- Παίζω νότες με τη σειρά του «πιάνου»
- Παίζω νότες με μεταβλητή πίνακα (*array – index*)
- RGB led
- Παίζω νότες με τη σειρά (*array notesfreq[]*)
- Απεικονίζω νότα RGB led (χρώμα), matrix 5x5 (γράμμα νότας) (*parallel arrays: notesfreq[], colors[], notesnames []*)
- Push button Piano micro:bit
- Διάβασε το τηλεχειριστήριο IR (Read IR telecommander)
- Juke Box micro:bit



- Δημιουργία εφαρμογής notes frequencies (ηλεκτρονικό μεταλλόφωνο)

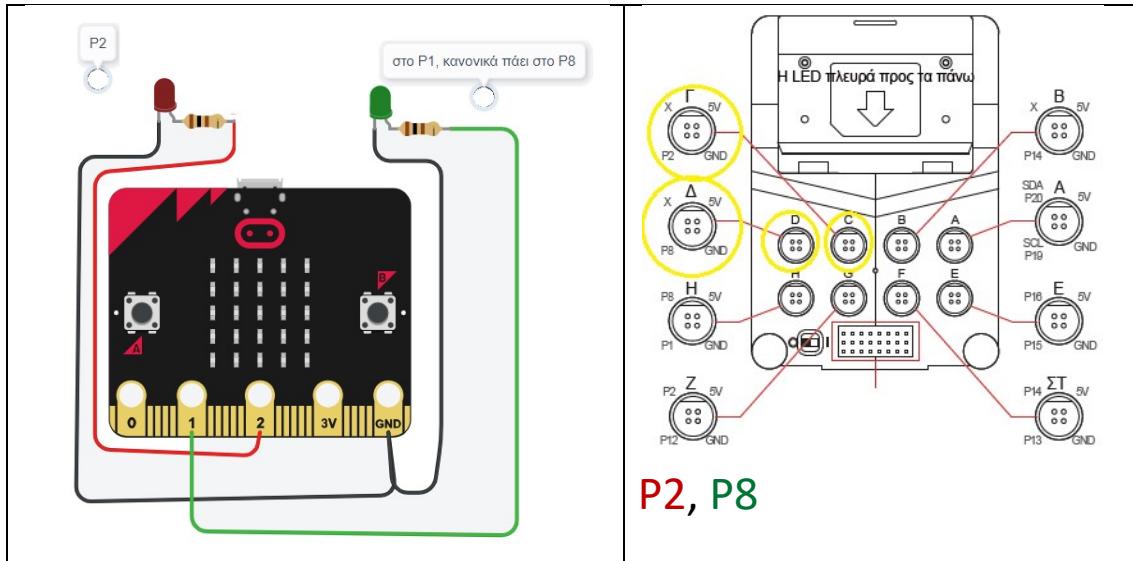


- Membrane Keypad 1x4 Piano Arduino
- Push button Piano Arduino – διαφορά με micro:bit
- Βρες τη συχνότητα του ήχου με το Arduino (FFT algorithm)
 - Απεικόνιση σε Serial monitor
 - Απεικόνιση σε Seven Segment Display
- Arduino Servo Motor
- I2C connection Arduino → micro:bit
 - Απεικονίζω νότα RGB led (χρώμα), matrix 5x5 (γράμμα νότας) στο micro:bit
- Συνδεσμολογία των Servo Motors για να παίξουν Μεταλλόφωνο



Το Ρομπότ Κυμώ

Αναβοσβήνω led



Χρησιμοποιούμε τις εντολές:

Βασικά Είσοδος Μουσική

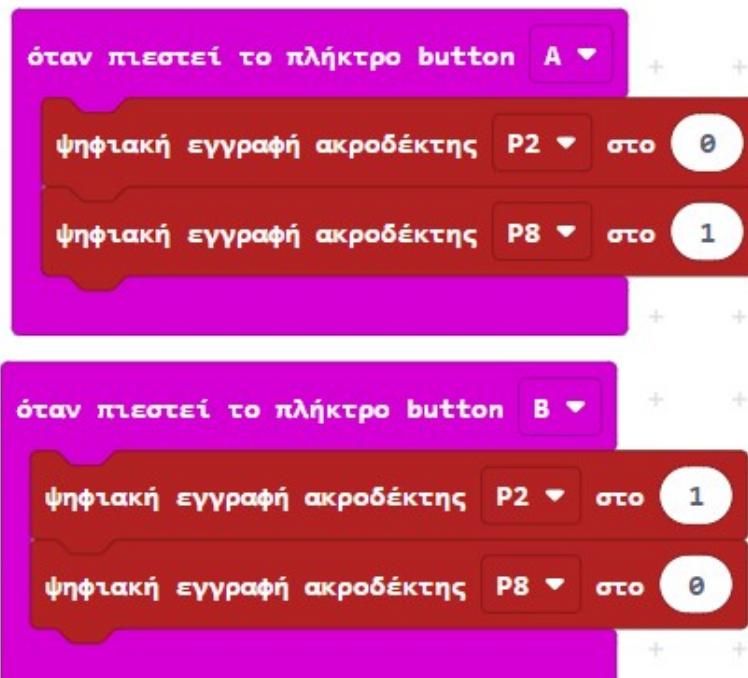
↖ Για Προχωρημένους → Ακροδέκτες

Παραδείγματα:

Παιζει συνέχεια, κάθε μισό δευτερόλεπτο αλλάζει



Πατώντας τα κουμπιά, το κάνουμε εμείς να ανάβει

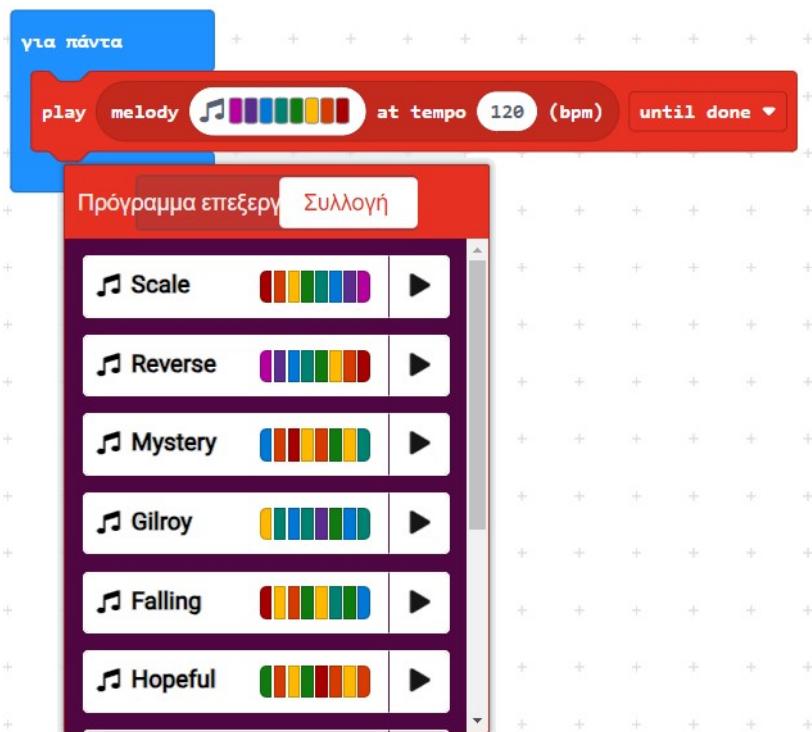


Παιζω νότες

Να παίζει τις νότες τη μία μετά την άλλη



Να παίζει έτοιμες μελωδίες



Η σειρά σας τώρα...

Φτιάξτε πρόγραμμα που:

- Όταν πατάμε το A, να ανάβει το κόκκινο Led και να παίζει Μεσαία Ντο για **μισό** κτύπο.
- Όταν πατάμε το B, να ανάβει το πράσινο Led και να παίζει Μεσαία Ρε για **ένα τέταρτο** κτύπο.
- Όταν πατάμε τα A+B, να ανάβουν το κόκκινο και το πράσινο Led και να παίζει Μεσαία Μι για **τέσσερις** κτύπους.

Φτιάξτε πρόγραμμα που:

- Όταν πατάμε το A, να ανάβει το κόκκινο Led και να παίζει τη μία μετά την άλλη όλες τις μεσαίες νότες για **μισό** κτύπο.
- Όταν πατάμε το B, να ανάβει το πράσινο Led και να παίζει τη μία μετά την άλλη όλες τις υψηλές νότες για δύο κτύπους.
- Όταν πατάμε τα A+B, να ανάβουν το κόκκινο και το πράσινο Led και να παίζει τη μία μετά την άλλη όλες τις χαμηλές νότες για **έναν** κτύπο.

?? Σκεφτείτε:

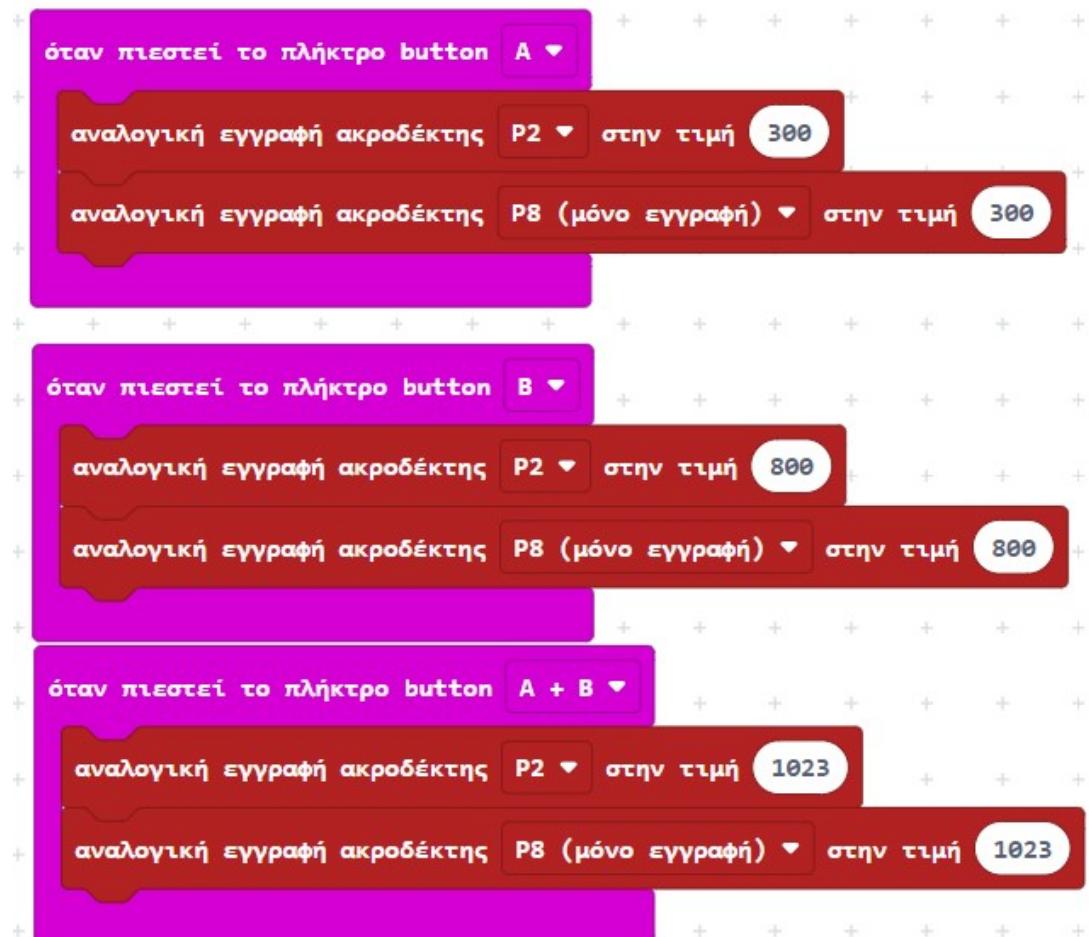
Πώς μπορούμε πατώντας κάθε φορά το A να παίζει με τη σειρά την επόμενη νότα; Ντο, Ρε, Μι, Φα, Σολ, Λα, Σι

Φτιάξτε πρόγραμμα που: (*matrices – leave for last*),

- Πατώντας A να παίζει τις χαμηλές νότες
- Πατώντας B να παίζει τις μεσαίες νότες
- Πατώντας A+B να παίζει τις υψηλές νότες

Ανάβω led με αυξανόμενή ένταση φωτός (αριθμητικά)

Δείτε και αυτές τις εντολές:



Τι κάνουν; Ποία η διαφορά με τα πρώτα παραδείγματα; (*analog - variables*)

Φτιάξτε πρόγραμμα που:

Πατώντας το A να αυξάνει την ένταση του φωτός

Πατώντας το B να μειώνει την ένταση του φωτός

Πατώντας τα A+B να σβήνει το φως.

Φυλλάδιο για φωτοτυπία

Η σειρά σας τώρα...

Χρησιμοποιούμε τις εντολές:

 Βασικά  Είσοδος  Μουσική

 Για Προχωρημένους   Ακροδέκτες

Φτιάξτε πρόγραμμα που:

- Όταν πατάμε το Α, να ανάβει το κόκκινο Led και να παίζει Μεσαία Ντο για μισό κτύπο.
- Όταν πατάμε το Β, να ανάβει το πράσινο Led και να παίζει Μεσαία Ρε για ένα τέταρτο κτύπο.
- Όταν πατάμε τα Α+Β, να ανάβουν το κόκκινο και το πράσινο Led και να παίζει Μεσαία Μι για τέσσερις κτύπους.

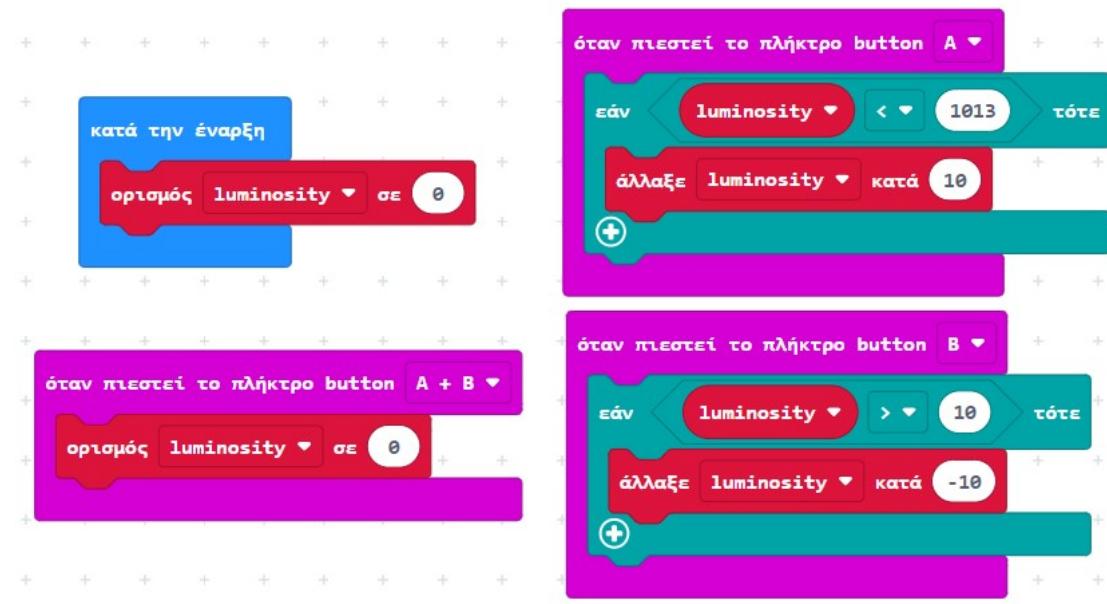
Φτιάξτε πρόγραμμα που:

- Όταν πατάμε το Α, να ανάβει το κόκκινο Led και να παίζει τη μία μετά την άλλη όλες τις μεσαίες νότες για μισό κτύπο.
- Όταν πατάμε το Β, να ανάβει το πράσινο Led και να παίζει τη μία μετά την άλλη όλες τις υψηλές νότες για δύο κτύπους.
- Όταν πατάμε τα Α+Β, να ανάβουν το κόκκινο και το πράσινο Led και να παίζει τη μία μετά την άλλη όλες τις χαμηλές νότες για έναν κτύπο.

Τι κάνουν; Ποία η διαφορά με τα πρώτα παραδείγματα;

Φτιάξτε πρόγραμμα που:

- Πατώντας το A να αυξάνει την ένταση του φωτός
 - Πατώντας το B να μειώνει την ένταση του φωτός
 - Πατώντας τα A+B να σβήνει το φως.
-

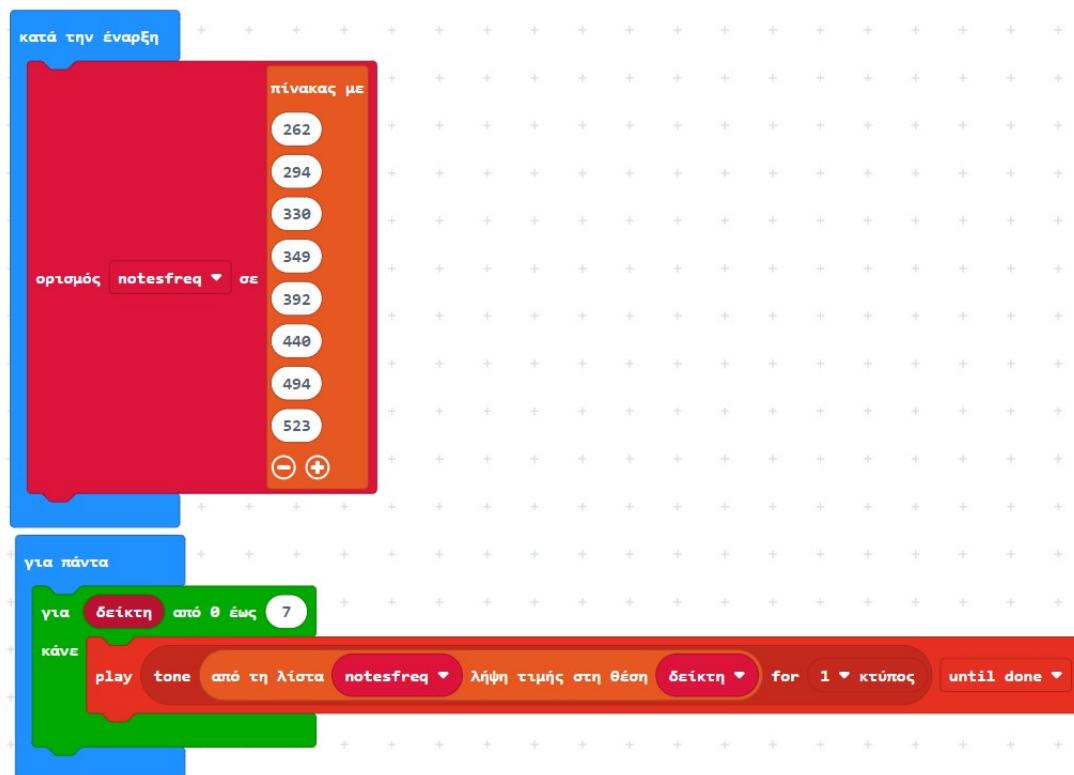


Παιζω νότες αποθηκευμένες σε πίνακα

Παιζω νότες αποθηκευμένες σε πίνακα (array)

Πώς μπορούμε πατώντας κάθε φορά το A να παίζει με τη σειρά την επόμενη νότα;

Ντο, Ρε, Μι, Φα, Σολ, Λα, Σι



Φτιάξτε πρόγραμμα που:

- Πατώντας A να παίζει τις χαμηλές νότες **C3..B3**
- Πατώντας B να παίζει τις μεσαίες νότες **C4..B4**
- Πατώντας A+B να παίζει τις υψηλές νότες **C5..B5**
(κάνετε στρογγυλοποίηση στη συχνότητα)
(Μπορείτε να κάνετε 3 πίνακες ή έναν ενιαίο πίνακα, θέλει προσοχή στον δείκτη).

NOTE FREQUENCY CHART HEROIC AUDIO											
	Octave 0	Octave 1	Octave 2	Octave 3	Octave 4	Octave 5	Octave 6	Octave 7	Octave 8	Octave 9	Octave 10
C	16.35	32.70	65.41	130.81	261.63	523.25	1046.50	2093.00	4186.01	8372.02	16744.04
C#	17.32	34.65	69.30	138.59	277.18	554.37	1108.73	2217.46	4434.92	8869.84	17739.69
D	18.35	36.71	73.42	146.83	293.66	587.33	1174.66	2349.32	4698.64	9397.27	18794.55
D#	19.45	38.89	77.78	155.56	311.13	622.25	1244.51	2489.02	4978.03	9956.06	19912.13
E	20.60	41.20	82.41	164.81	329.63	659.26	1318.51	2637.02	5274.04	10548.08	
F	21.83	43.65	87.31	174.61	349.23	698.46	1396.91	2793.83	5587.65	11175.30	
F#	23.12	46.25	92.50	185.00	369.99	739.99	1479.98	2959.96	5919.91	11839.82	
G	24.50	49.00	98.00	196.00	392.00	783.99	1567.98	3135.96	6271.93	12543.86	
G#	25.96	51.91	103.83	207.65	415.30	830.61	1661.22	3322.44	6644.88	13289.75	
A	27.50	55.00	110.00	220.00	440.00	880.00	1760.00	3520.00	7040.00	14080.00	
A#	29.14	58.27	116.54	233.08	466.16	932.33	1864.66	3729.31	7458.62	14917.24	
B	30.87	61.74	123.47	246.94	493.88	987.77	1975.53	3951.07	7902.13	15804.26	

Digital RGB micro:bit

	R	G	B
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Ορίζουμε 3 πίνακες με μήκος 8: R, G, B (διαβάζουμε τον πίνακα κάθετα).

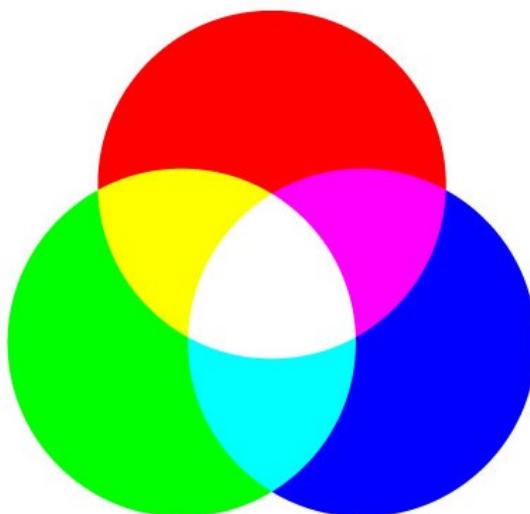
Παρατηρήστε πως αλλάζουν τα 0 και 1 και αν υπάρχει κάποιο μοτίβο στην αλλαγή.

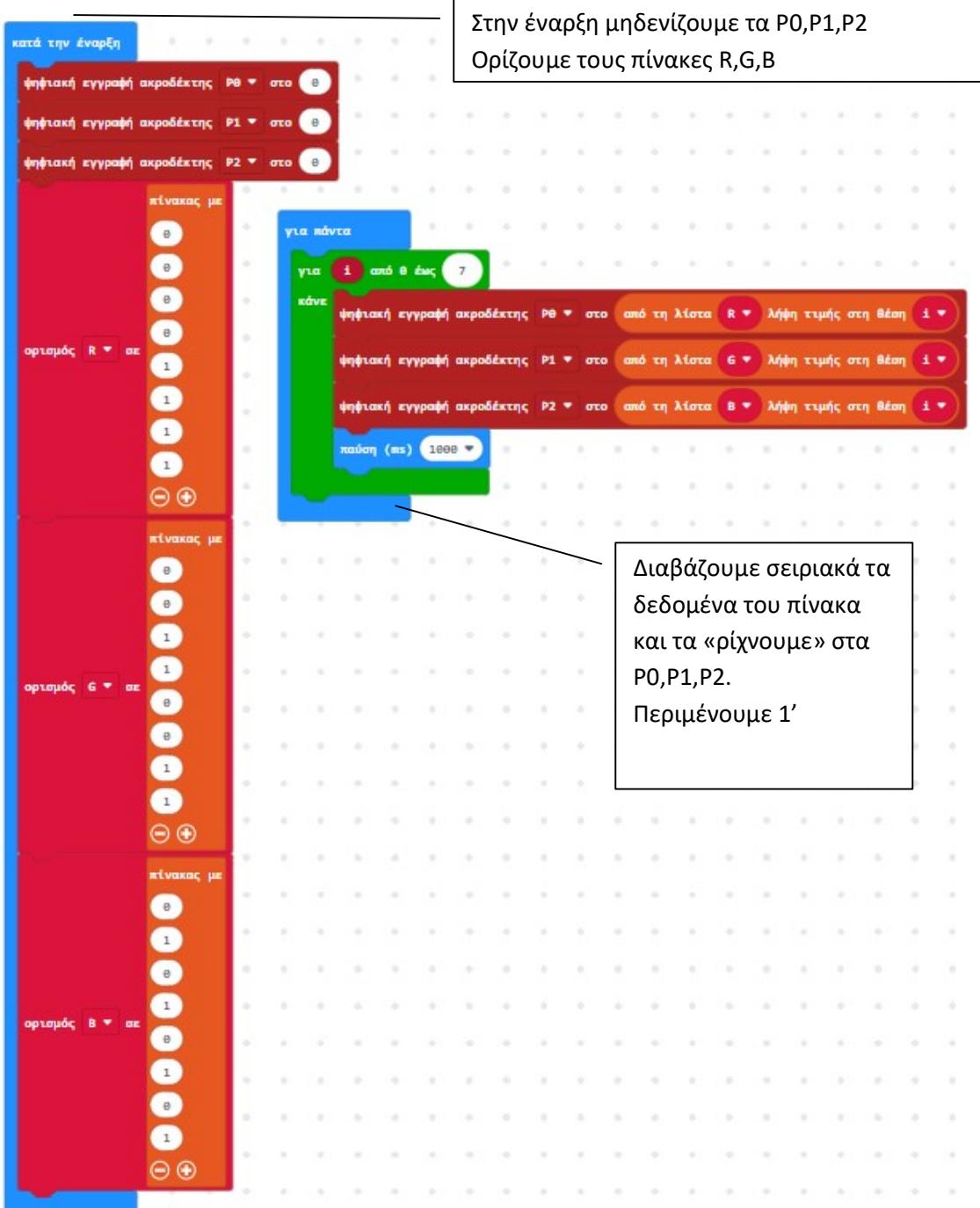
R = [0, 0, 0, 0, 1, 1, 1, 1]

G = [0, 0, 1, 1, 0, 0, 1, 1]

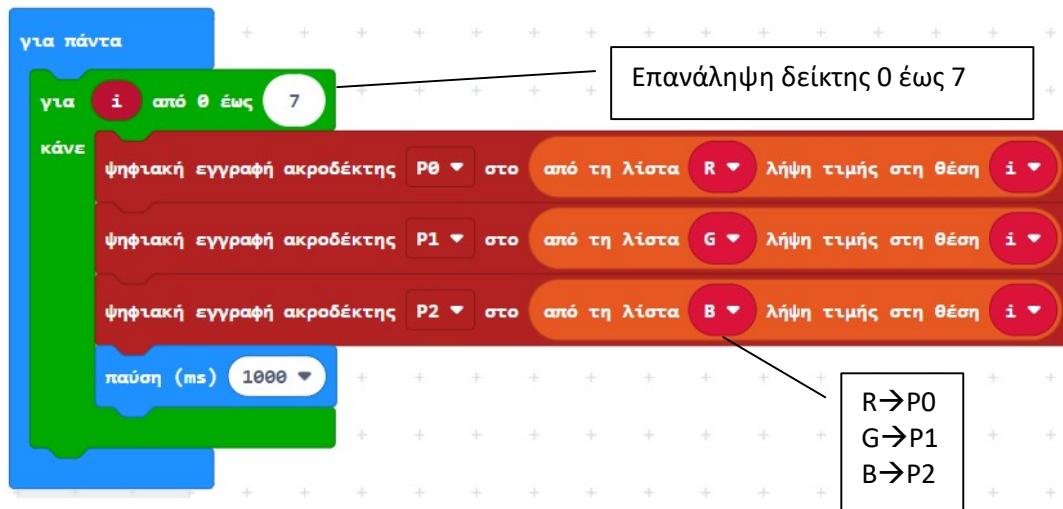
B = [0, 1, 0, 1, 0, 1, 0, 1]

Έτσι, μπορούμε να έχουμε μέχρι 7 χρώματα: Κόκκινο, Πράσινο, Μπλε, Μοβ,
Γαλάζιο, Κίτρινο και Λευκό





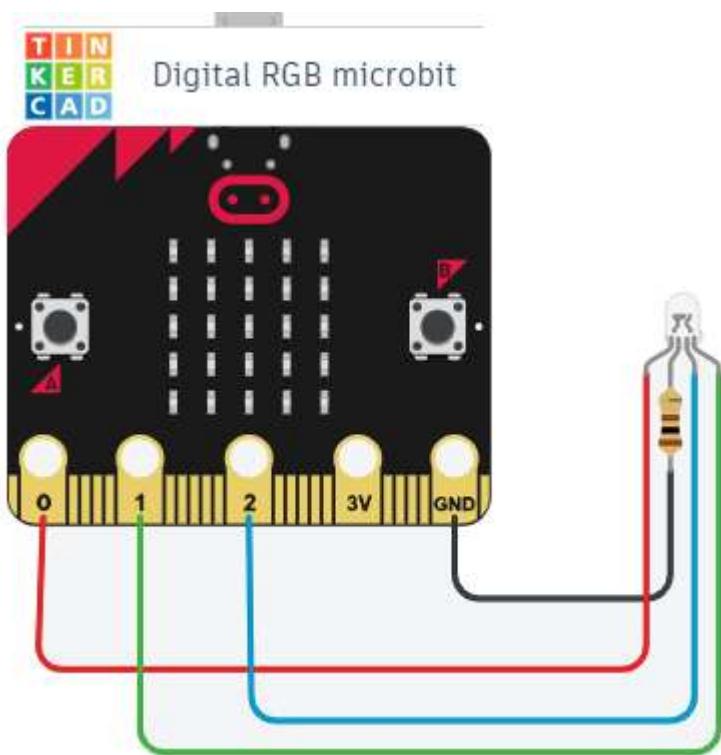
RGB Digital microbit for tinkercad



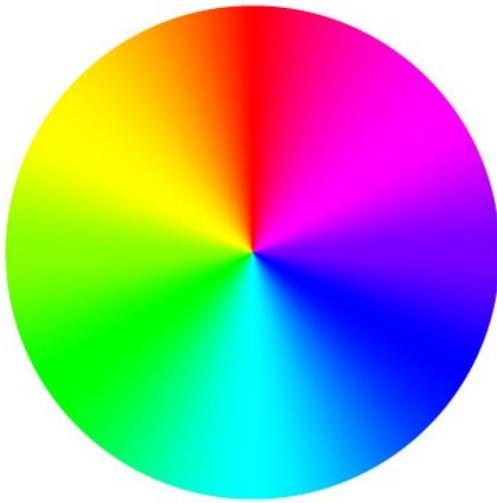
Ο κώδικας σε  Python: (Copy – Paste στο Tinkercad)

```
pins.digital_write_pin(DigitalPin.P0, 0)
pins.digital_write_pin(DigitalPin.P1, 0)
pins.digital_write_pin(DigitalPin.P2, 0)
R = [0, 0, 0, 0, 1, 1, 1, 1]
G = [0, 0, 1, 1, 0, 0, 1, 1]
B = [0, 1, 0, 1, 0, 1, 0, 1]

def on_forever():
    for i in range(8):
        pins.digital_write_pin(DigitalPin.P0, R[i])
        pins.digital_write_pin(DigitalPin.P1, G[i])
        pins.digital_write_pin(DigitalPin.P2, B[i])
        basic.pause(1000)
basic.forever(on_forever)
```



Επέκταση: Αναλογικό RGB



Αν θέλουμε να έχουμε χρώματα σε όλο το φάσμα, θα πρέπει να αλλάζουμε αναλογικά τα RGB. Αντί για 0 και 1 έχουμε δυνατότητα να δίνουμε τιμές από **0** έως **1023**. Βλέπουμε ότι έχουμε στην πραγματικότητα δεν έχουμε άπειρες επιλογές, αλλά συγκεκριμένες.

Οι πίνακες μας θα μπορούσαν να έχουν για παράδειγμα τις παρακάτω τιμές:

R = [1023, 1023, 1023, 1023, 1023, 0, 0, 0]

G = [0, 0, 818, 818, 1023, 1023, 1023, 0]

B = [0, 1023, 1023, 409, 0, 0, 1023, 1023]

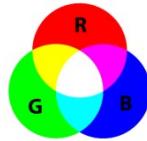
που μας δημιουργεί τα παρακάτω χρώματα:

κόκκινο	μοβ	ροζ	πορτοκαλί	κίτρινο	πράσινο	γαλάζιο	Μπλε
red	magenta	pink	orange	yellow	green	cyan	blue

Βέβαια, η απόχρωση εξαρτάται από το συγκεκριμένο RGB led που έχουμε και θα πρέπει να κάνουμε πειραματισμούς ώστε να βρούμε τις ακριβείς τιμές.

(Βλέπε και αρχείο: διάβασμα τιμών με ποτενσιόμετρα.docx)

Read RGB with potentiometers (Analog RGB)



Ποτενσιόμετρο

Κύκλωμα

Κώδικας

Θέλουμε να διαβάσουμε τις αναλογικές τιμές που χρειάζεται να εισάγουμε για να κάνουμε το RGB Led να ανάψει με χρώματα κίτρινο, πορτοκαλί και ροζ.

Πατώντας **A** διαβάζουμε το **R**

Πατώντας **B** διαβάζουμε το **G**

Πατώντας **A+B** διαβάζουμε το **B**

Καταγράφουμε τις τιμές στον παρακάτω πίνακα:

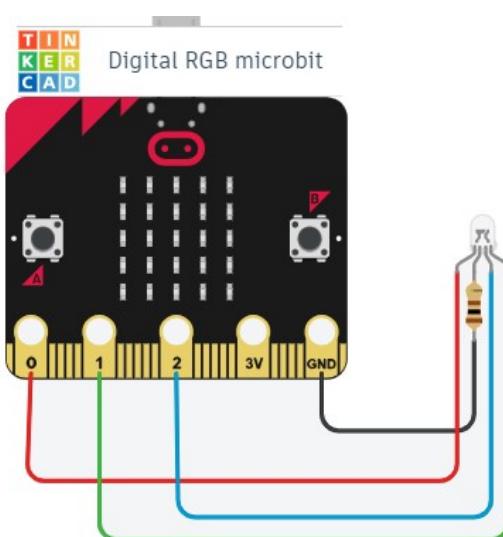
Χρώμα	A (R)	B (G)	A+B (B)
κίτρινο			
πορτοκαλί			
Ροζ			

Αυτές τις τιμές θα τις χρησιμοποιήσουμε για να ανάψουμε σωστά τα χρώματα στο Led και να ταιριάξουν με αυτά της εφαρμογής.



Digital RGB micro:bit

Πατώντας το button A του micro:bit, θα ανάβει ένα RGB Led με το χρώμα της νότας στην εφαρμογή του κινητού. Στο matrix 5X5 θα εμφανίζεται η διεθνής ονομασία της νότας . Επίσης, θα ακούγεται η νότα από το ενσωματωμένο piezo buzzer του microbit. Αυτό θα γίνεται με τη σειρά, γι' αυτό και χρησιμοποιούμε «παράλληλους» πίνακες. Ο δείκτης **i** μας επιτρέπει να προσπελάσουμε μία-μία τις νότες.



→ **i**

i	Note		R	G	B	Frequency (Hz)
			P	P	P	
0	C4	C	0	0	1	262
1	D4	D	0	1	0	294
2	E4	E	0	1	1	330
3	F4	F	1	0	0	349
4	G4	G	1	0	1	392
5	A4	A	1	1	0	440
6	B4	B	1	1	1	494
7	C5	c	0	0	1	523

Frequency Music Notes :

- C4 (262Hz)
- D4 (294 Hz)
- E4 (330 Hz)
- F4 (349 Hz)
- G4 (392 Hz)
- A4 (440 Hz)
- B4 (494 Hz)
- C5 (523 Hz)

```

testRGBwith buttonAserially

def on_button_pressed_a():
    global i
    music.play(music.tonePlayable(freqNotes[i], music.beat(BeatFraction.WHOLE)),
               music.PlaybackMode.UNTIL_DONE)
    pins.digital_write_pin(DigitalPin.P10, R[i])
    pins.digital_write_pin(DigitalPin.P11, G[i])
    pins.digital_write_pin(DigitalPin.P12, B[i])
    basic.show_string("" + (notes[i]))
    if i < 7:
        i += 1
    else:
        i = 0
input.on_button_pressed(Button.A, on_button_pressed_a)
i = 0
B: List[number] = []
G: List[number] = []
R: List[number] = []
freqNotes: List[number] = []
notes: List[str] = []

```

```

colors = ["BLUE",
          "GREEN",
          "CYAN",
          "RED",
          "MAGENTA",
          "YELLOW",
          "WHITE",
          "BLUE"]

notes = ["C", "D", "E", "F", "G", "A", "B", "c"]
freqNotes = [262, 294, 330, 349, 392, 440, 494, 523]
R = [0, 0, 0, 1, 1, 1, 1, 0]
G = [0, 1, 1, 0, 0, 1, 1, 0]
B = [1, 0, 1, 0, 1, 0, 1, 1]
basic.clear_screen()
pins.digital_write_pin(DigitalPin.P10, 0)
pins.digital_write_pin(DigitalPin.P11, 0)
pins.digital_write_pin(DigitalPin.P12, 0)
i = 0

```

testRGBwith buttonAserially

The Scratch script consists of the following blocks:

- Control: `when green flag clicked`
- Control: `repeat []`
- Control: `end`
- Control: `if then` (when `button A pressed`)
 - Sound: `play tone [freqNotes v] [pitch v] until done` (freqNotes: R, pitch: i; freqNotes: G, pitch: i; freqNotes: B, pitch: i)
 - Sound: `play tone [notes v] [pitch v] until done` (notes: R, pitch: i)
 - Control: `repeat []`
 - Control: `end`
- Control: `repeat []`
- Control: `end`

A callout box on the right side contains the text: "Παίζει την αντίστοιχη νότα".

A separate box contains the text: "Επιλέξτε το Pin, σύμφωνα με:" followed by a pinout diagram of a Microbit board. The pins shown are P1, P8, P12, P2, P13, P14, P15, and P16. The 5V and GND pins are also labeled.

The image displays three screenshots of the micro:bit Play Music app interface, illustrating different ways to select notes:

- Screenshot 1 (Left):** Shows a vertical list of musical notes (C, D, E, F, G, A, B, C) under the heading "πίνακας με". Below this, a list of frequencies (262, 294, 330, 349, 392, 440, 494, 523) is shown under the heading "πίνακας με". The mode is set to "notes".
- Screenshot 2 (Middle):** Shows a vertical list of binary values (0, 0, 0, 1, 1, 1, 0) under the heading "πίνακας με". Below this, a list of frequencies (262, 294, 330, 349, 392, 440, 494, 523) is shown under the heading "πίνακας με". The mode is set to "freqNotes".
- Screenshot 3 (Right):** Shows a vertical list of binary values (0, 0, 0, 1, 1, 1, 0) under the heading "πίνακας με". Below this, a list of frequencies (262, 294, 330, 349, 392, 440, 494, 523) is shown under the heading "πίνακας με". The mode is set to "notes".

Όλα μπαίνουν μέσα στο **κατά την έναρξη**.

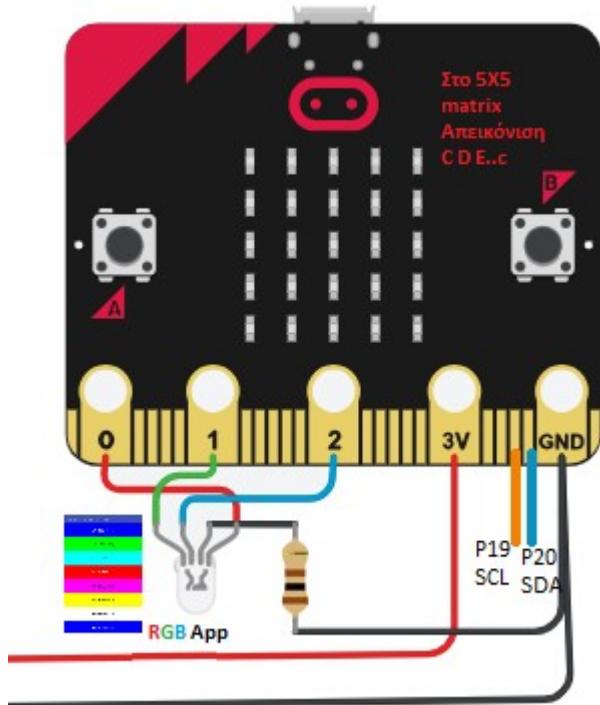
Στην έναρξη ορίζουμε τους πίνακες, καθαρίζουμε το matrix 5X5, μηδενίζουμε τα pins (RGB σβηστό) και ορίζουμε αρχική τιμή

για τον δείκτη πίνακα →0

Απεικόνιση RGB led, matrix 5x5, play tone

Παίζουμε τις νότες C4..C5 τη μία μετά την άλλη, απεικονίζεται το χρώμα τους στο συνδεδεμένο RGB led , φαίνεται το όνομά τους στο 5x5 matrix και ακούγεται ο ήχος από buzzer του micro:bit.

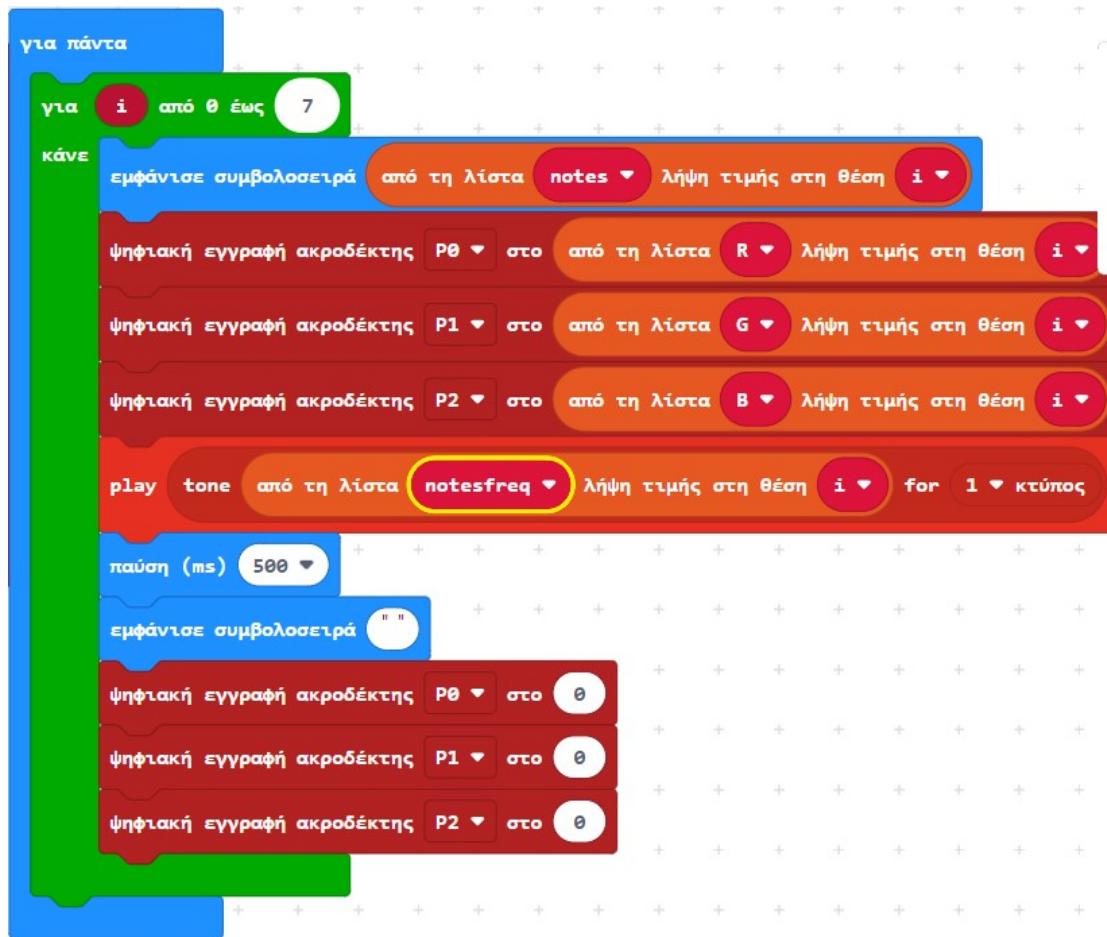
Εικόνα από 



Στο κατά την έναρξη δηλώνουμε τους πίνακες

```
notesfreq = [262, 294, 330, 349, 392, 440, 494, 523]
colors = ["BLUE", "GREEN", "CYAN", "RED", "MAGENTA", "YELLOW", "WHITE", "BLUE"]
notes = ["C", "D", "E", "F", "G", "A", "B", "c"]
R = [0, 0, 0, 1, 1, 1, 1, 0]
G = [0, 1, 1, 0, 0, 1, 1, 0]
B = [1, 0, 1, 0, 1, 0, 1, 1]
```

Επίσης, καθαρίζουμε την οθόνη.



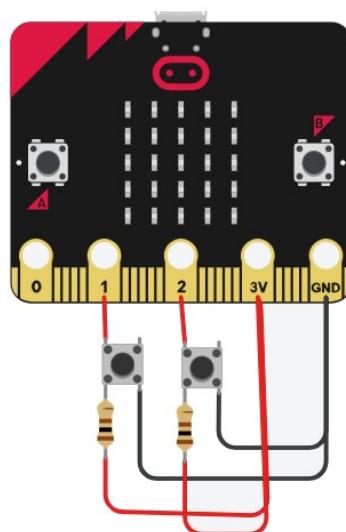
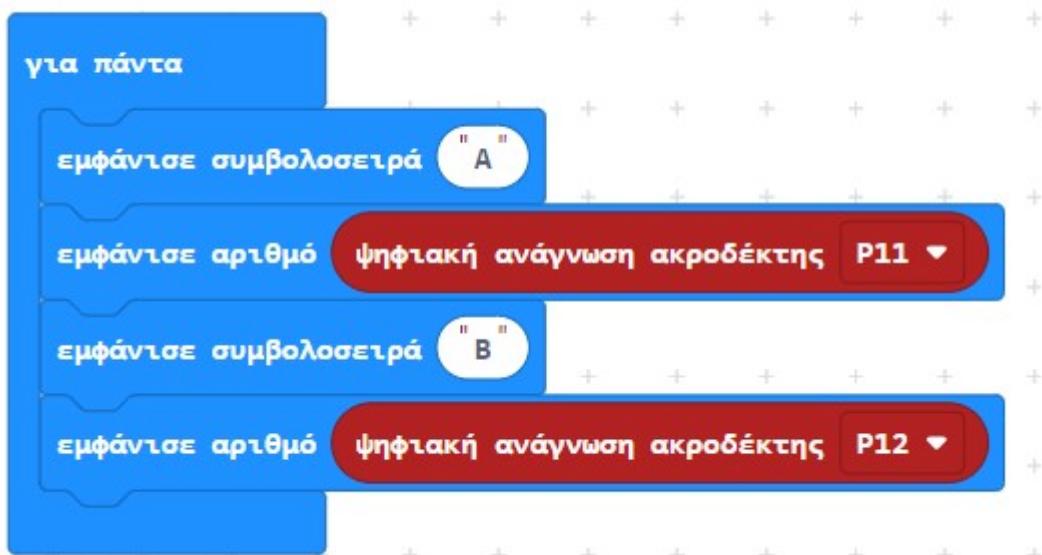
push button piano micro:bit

Αρχική δοκιμή σε microbit για πιάνο με push buttons

Test P11 P12 microbit piano

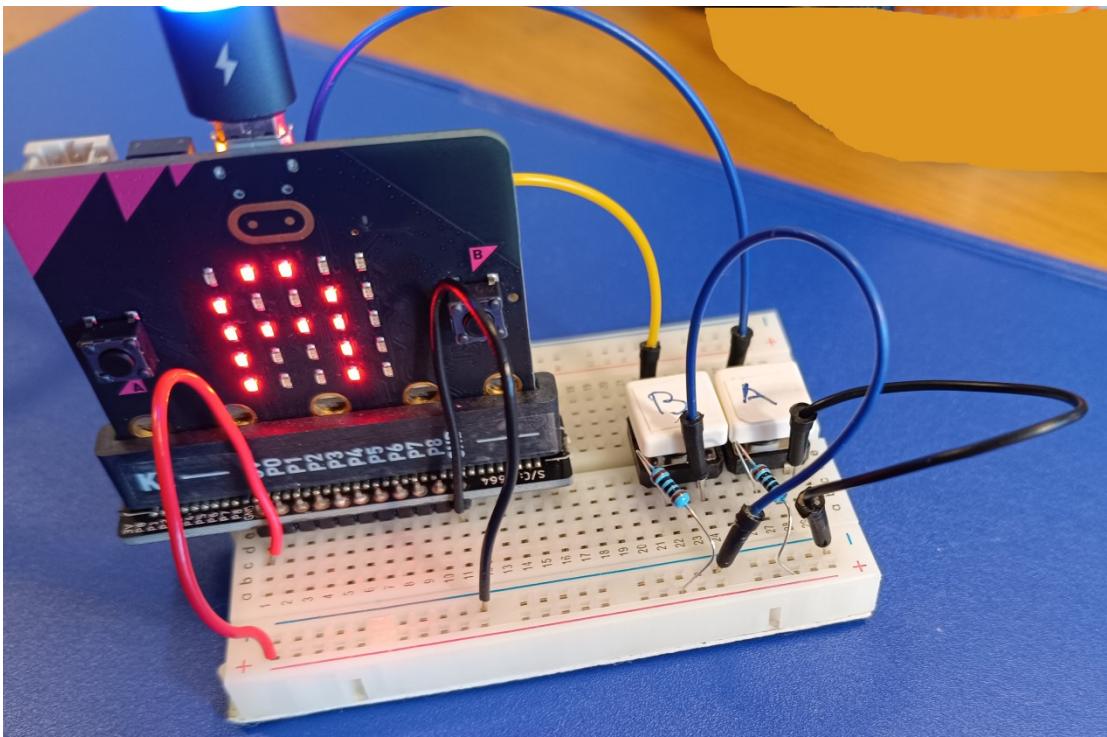
Έλεγχος Push Button σε micro:bit?? Τι διαβάζω όταν πατάω το push button;

test p11 p12 piano



test p11 p12 microbit piano

To Tinkercad αντί για P11, P12 βάζουμε P1, P2



Δοκιμή με μουσική

piano P11 P12

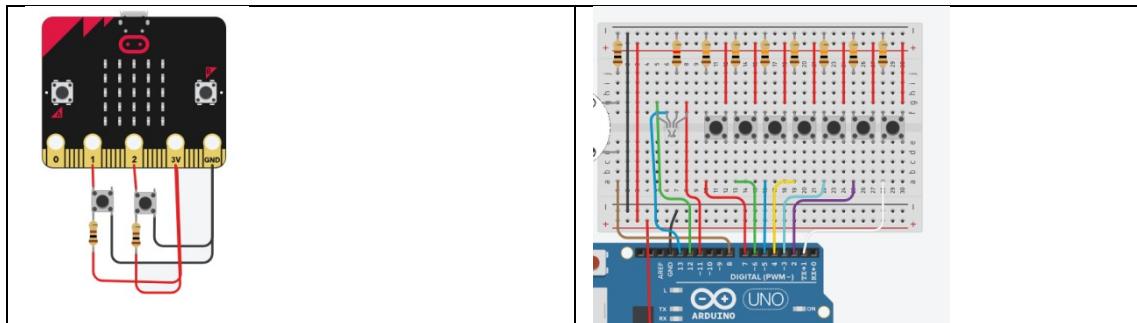
The Scratch script consists of three main sections:

- Section A:** Sets the volume to 100, plays the tone "Lambda" (Μεσαία Λα) for 1 beat until done, and loops back to the start.
- Section B:** Plays the tone "Sigma" (Μεσαία Σι) for 1 beat until done, loops back to the start, and then loops back to Section A.
- Section C:** Sets the volume to 0 and ends the script.

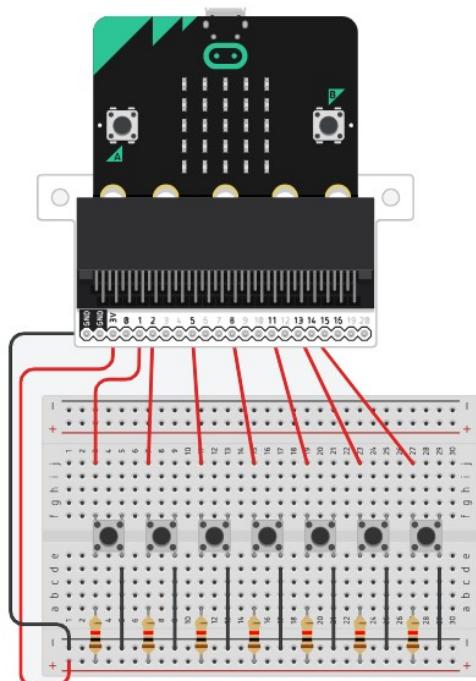
Comments in Greek are present above each section: "κατά την έναρξη" (at the beginning), "για πάντα" (for always), "καθαρισμός οθόνης" (screen clear), "ψηφιακή ανάγνωση ακροδέκτης" (digital reading from microphone), and "τότε" (then).

Αλλάζουμε τους κτύπους αν θέλουμε μεγαλύτερη διάρκεια σε 2 ή σε 4

Παρατηρήστε ότι στο micro:bit έχουμε αντίθετη λογική στους διακόπτες, από ότι στο Arduino.



Πιάνο micro:bit στο tinkercad με 7 νότες στα P1, P2, P5,P8,P11,P13,P14



Μέσα στο για πάντα, βάζουμε εάν για να παίξουμε τη μουσική

```
when green flag clicked
repeat (until done)
  play tone [440 v] for (1 sec) until done
end
costume switch [καθαρισμός οθόνης v]
```

Αιάρασε το τηλεχειριστήριο IR

Κύκλωμα, ir receiver στο P0.



<https://makecode.microbit.org/>

Επεκτάσεις → Φάξε για ir receiver, →makerbit-ir-receiver



makerbit-ir-receiver
IR receiver blocks for Keyestudio
Infrared Wireless Module Kit.

Συμβουλέψου το datasheet του Remote Control:

<https://www.beemong.com/product/17-key-ir-remote-control>

(decode NEC)

Γράψε τον παρακάτω κώδικα:

```
Remote Control Read Codes.hex
κατά την έναρξη
connect IR receiver at pin P0 ▾ and decode NEC ▾
on IR button any ▾ pressed ▾
    εμφάνισε αριθμό IR button
```

Καταγραφή κωδικών του Control στον πίνακα.

<i>Button</i>	<i>Code</i>
Λ	
>	
<	
▼	
OK	
1	
2	
3	
4	
5	
6	
7	
8	
9	
0	
*	
#	

<i>Button</i>	<i>Code</i>
^	98
>	194
<	34
v	168
OK	2
1	104
2	152
3	176
4	48
5	24
6	122
7	16
8	56
9	90
0	74
*	66
#	82



<https://www.beemong.com/product/17-key-ir-remote-control>



IR Remote Control Codes

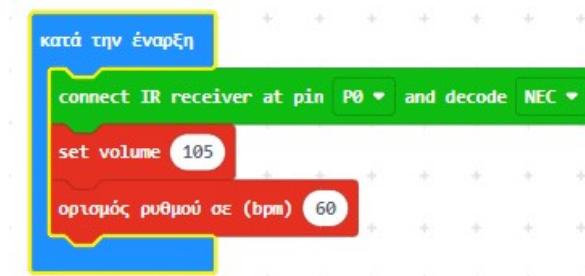
Key	IR Code	Key	IR Code
1	FFA25D	0	FF9867
2	FF629D	*	FF6897
3	FFE21D	#	FFB04F
4	FF22DD	UP / FORWARD	FF18E7
5	FF02FD	RIGHT	FF5AA5
6	FFC23D	DOWN / REVERSE	FF4AB5
7	FFE01F	LEFT	FF10EF
8	FFA857	OK	FF38C7
9	FF906F	REPEAT	FFFFFF

Juke Box Micro:bit

	BPM +	
Volume -		Volume +
	BPM -	
Τραγούδια		
0.		
1.		
2.		
3.		
4.		
5.		
6.		
7.		
8.		
9.		

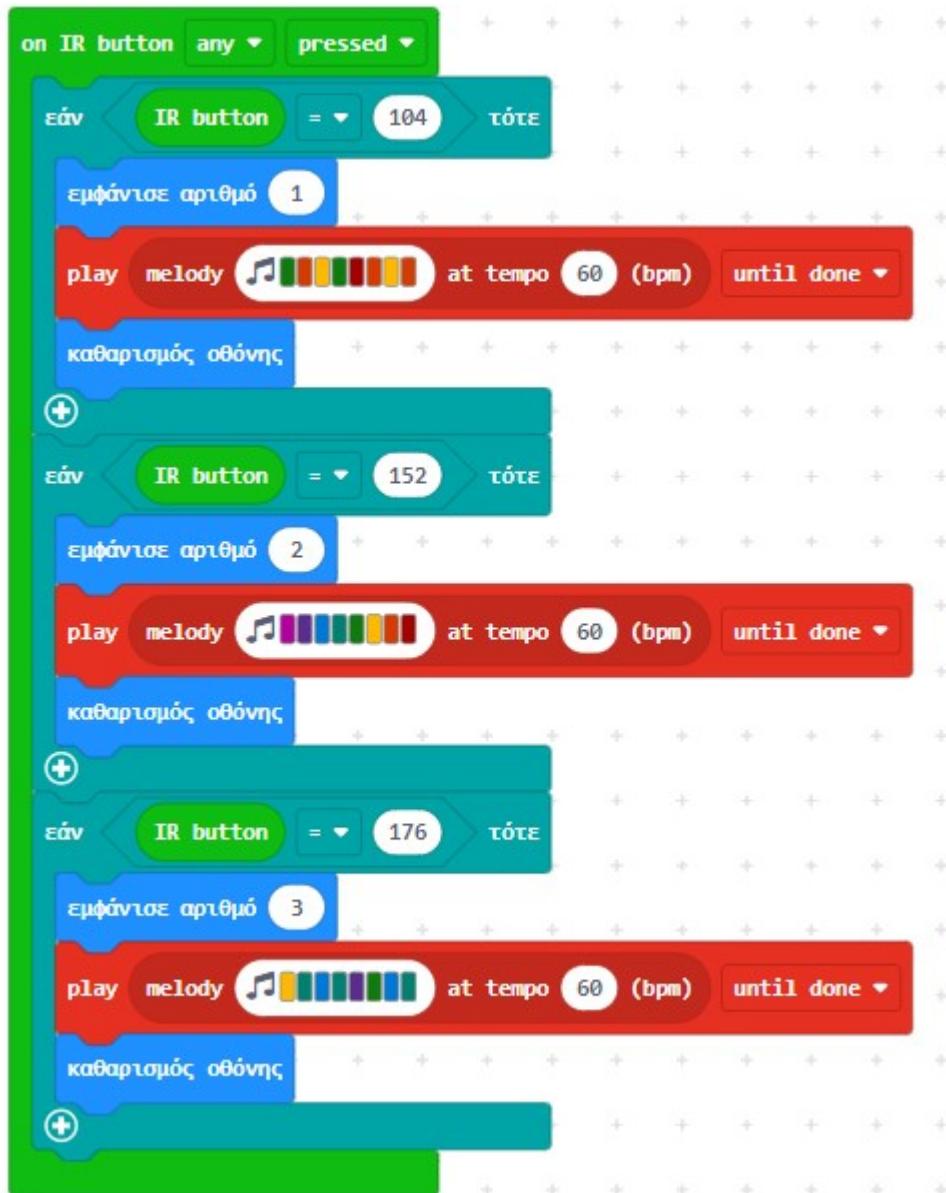
<https://www.beemong.com/product/17-key-ir-remote-control>

Αρχικοποίηση:



<i>Button</i>	<i>Code</i>
^	98
>	194
<	34
v	168
OK	2
1	104
2	152
3	176
4	48
5	24
6	122
7	16
8	56
9	90
0	74
*	66
#	82

Προγραμματισμός να παίζει μελωδία στα button 1, button 2, button 3:
(όμοια και οι άλλοι αριθμοί)



Πιό «προχωρημένος» προγραμματισμός: Με τα πλήκτρα < > ^ V Αυξάνω και μειώνω Volume και Ρυθμό, αντίστοιχα

Περιθώρια:	Volume	min: 0	max: 255	Αρχική Τιμή: 105	Μεταβολή: ±10
	Ρυθμός	min: 40	max: 500	Αρχική Τιμή: 60	Μεταβολή: ±20



microbit-remote-control-working (2).hex

Frequency Music Notes App Inventor

 <p>Frequency Music Notes</p> <p>C4 (262Hz)</p> <p>D4 (294 Hz)</p> <p>E4 (330 Hz)</p> <p>F4 (349 Hz)</p> <p>G4 (392 Hz)</p> <p>A4 (440 Hz)</p> <p>B4 (494 Hz)</p> <p>C5 (523 Hz)</p> <p>Analog Colors</p> <p>https://ai2.appinventor.mit.edu/#6584190199988224</p>	 <p>Frequency Music Notes</p> <p>C4 (262Hz)</p> <p>D4 (294 Hz)</p> <p>E4 (330 Hz)</p> <p>F4 (349 Hz)</p> <p>G4 (392 Hz)</p> <p>A4 (440 Hz)</p> <p>B4 (494 Hz)</p> <p>C5 (523 Hz)</p> <p>Digital Colors (Simpler for RGB Led)</p> <p>https://ai2.appinventor.mit.edu/#6473920471433216</p>
--	--

Δημιουργία Εφαρμογής στο App Inventor

The screenshot shows the App Inventor workspace. On the left, the component tree displays a screen named "Screen1" containing eight buttons labeled "Button1" through "Button8" and one sound component "Sound1". Below the screen is a "Media" section listing eight files: 1.wav, 2.wav, 3.wav, 4.wav, 5.wav, 6.wav, 7.wav, and 8.wav. To the right, there are eight separate scripts, one for each button's "Click" event. Each script uses the "when [button].Click" condition, followed by a "do" block that sets "Sound1.Source" to a specific wav file (e.g., "1.wav" for Button1) and then calls "Sound1.Play". The URL <https://ai2.appinventor.mit.edu/#6584190199988224> is visible at the top.

Components: 8 Buttons, 1 Sound (το θεωρούμε σαν «ηχείο» που παράγει ήχους).

Button → Height 12% (8 buttons μοιράζονται την οθόνη $100 \div 8$).

Button → Width 100% (buttons να «απλώνεται» σε όλη την οθόνη).

Χρώματα Button Αναλογικά ή Ψηφιακά + Χρώμα Κειμένου σε αντίθεση.

Media: Ήχοι (1.wav... 8.wav) παράγονται με το πρόγραμμα **Audacity**:

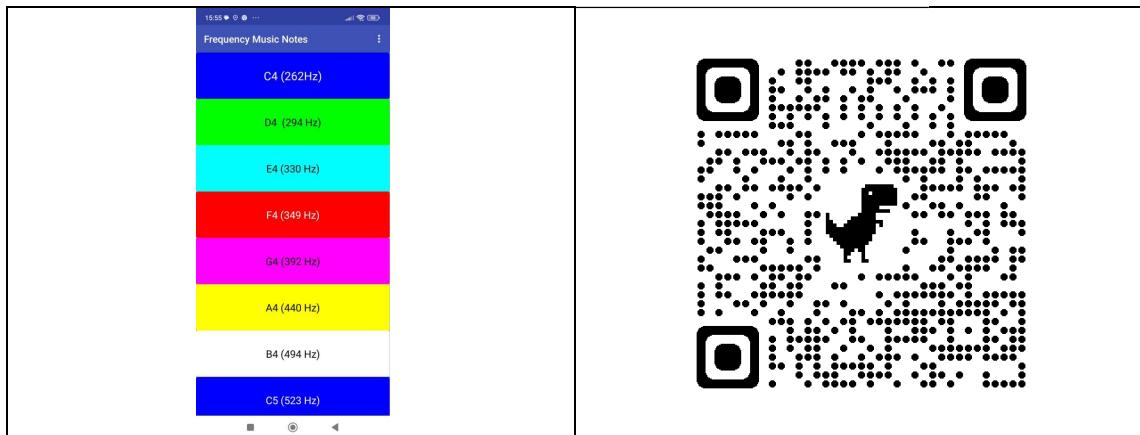


Ενεργοποίηση → Tone

Διάρκεια ήχων: 2sec

C ₄	DO	48 (C4 = 262 Hz)	1.wav
D ₄	RE	50 (D4 = 294 Hz)	2.wav
E ₄	MI	52 (E4 = 330 Hz)	3.wav
F ₄	FA	53 (F4 = 349 Hz)	4.wav
G ₄	SOL	55 (G4 = 392 Hz)	5.wav
A ₄	LA	57 (A4 = 440 Hz)	6.wav
B ₄	SI	59 (B4 = 494 Hz)	7.wav
C ₅	DO+	60 (C5 = 523 Hz)	8.wav

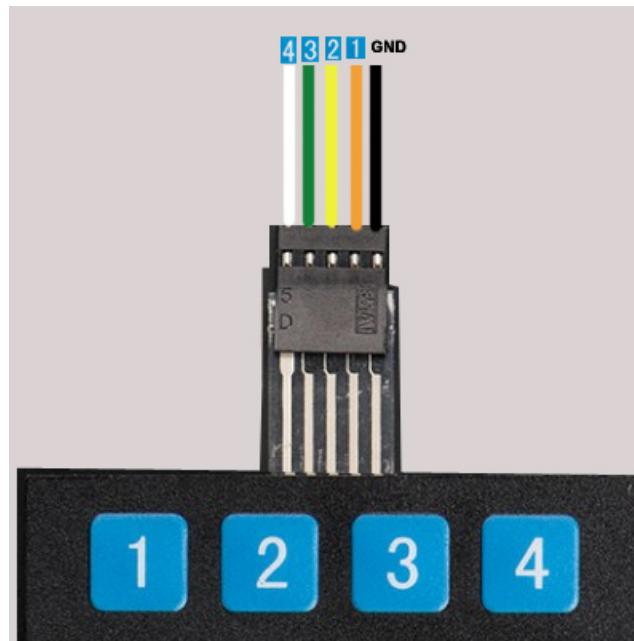
Κατεβάστε στο κινητό την εφαρμογή από το grafis:



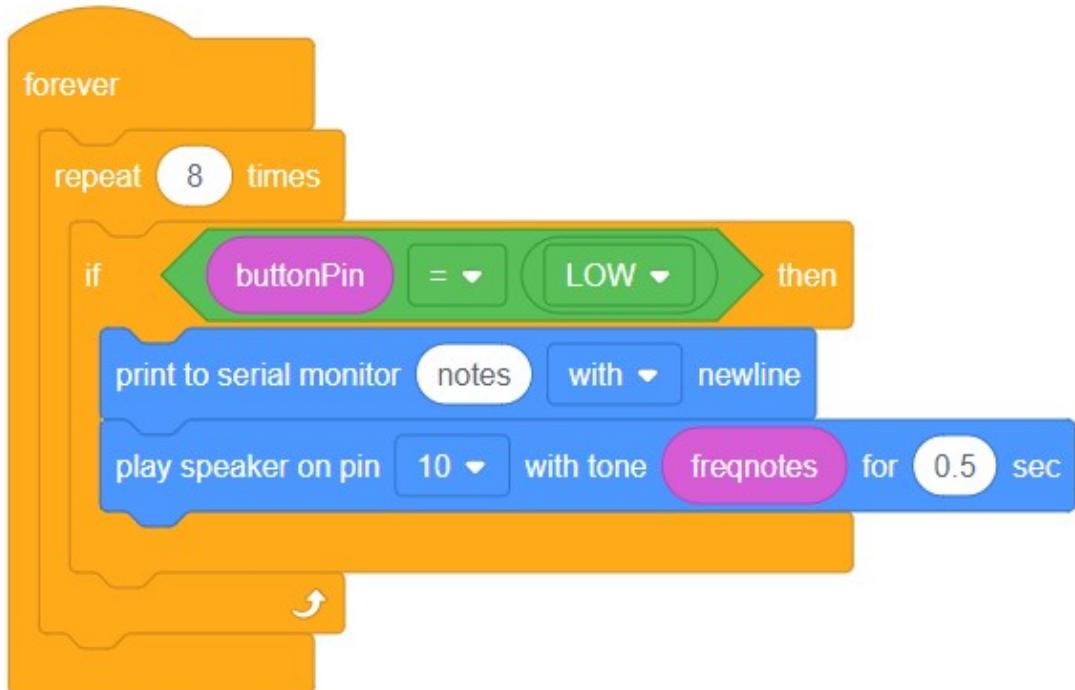
Ρυθμίσεις: Επιτρέψτε να εγκατασταθεί από άγνωστες πηγές.

Membrane Keypad Piano

Membrane Keypad 1X4 connection



Membrane Keypad	Membrane Keypad pin	Arduino pin	note	note code on C++ note frequency
#1	2	3	DO	48 (C4 = 262 Hz)
	3	2	RE	50 (D4 = 294 Hz)
	4	5	MI	52 (E4 = 330 Hz)
	5	4	FA	53 (F4 = 349 Hz)
#2	2	7	SOL	55 (G4 = 392 Hz)
	3	6	LA	57 (A4 = 440 Hz)
	4	9	SI	59 (B4 = 494 Hz)
	5	8	DO+	60 (C5 = 523 Hz)



^ Η λογική του προγράμματος σε Blocks

Για πάντα

Διάβασε με τη σειρά τα button 0..7 (8 buttons)

Αν button=LOW

Δείξε τη νότα στο Serial Printer

Παιξε τη νότα στο buzzer

Σημείωση: όταν ορίζουμε ένα pin ως INPUT_PULLUP, συμπεριφέρεται σαν να είναι συνεχώς HIGH. Αν πατήσουμε το button, γίνεται LOW.

myMembraneKeypadTry_2Keypads_copy_20240327194932.ino

```

/*
1st Membrane Keypad
Membrane Keypad pin 1->GND
Membrane Keypad pin 2-> C4 Arduino pin 3 orange wire
Membrane Keypad pin 3-> D4 Arduino pin 2 yellow wire
Membrane Keypad pin 4-> E4 Arduino pin 5 green wire
Membrane Keypad pin 5-> F4 Arduino pin 4 white wire

```

2nd Membrane Keypad

```

Membrane Keypad pin 1->GND
Membrane Keypad pin 2-> G4 Arduino pin 7 orange wire
Membrane Keypad pin 3-> A4 Arduino pin 6 yellow wire
Membrane Keypad pin 4-> B4 Arduino pin 9 green wire
Membrane Keypad pin 5-> C5 Arduino pin 8 white wire

```

orange-yellow-green-white-orange-yellow-green-white

*/

```

// set pin numbers:

const int buttonPin[] = {3,2,5,4,7,6,9,8};      // the number of the
pushbutton pins 4/3/2/1/5GND Wires like XX XX and a big X for 2 keypads
const String notes[] = {"C4", "D4", "E4", "F4", "G4", "A4", "B4", "C5"}; // 
universal notes names
const int freqnotes[]={262,294,330,349,392,440,494,523};//C4 to C5
frequencies, no decimals

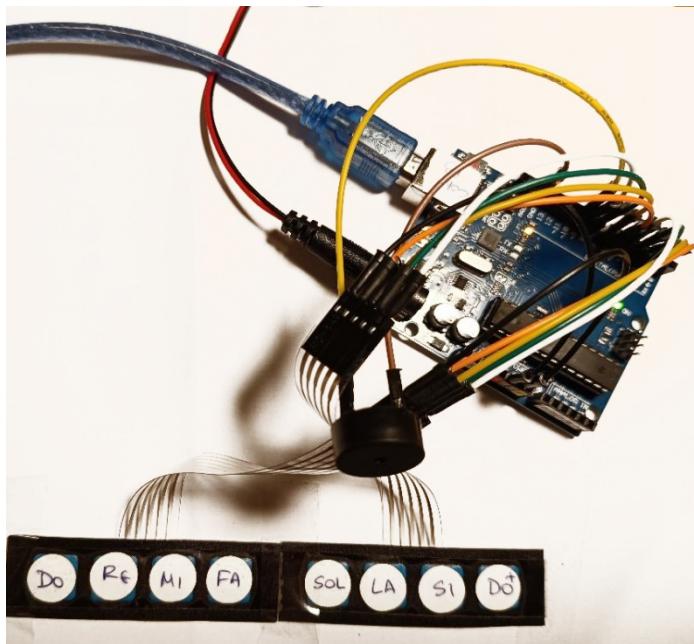
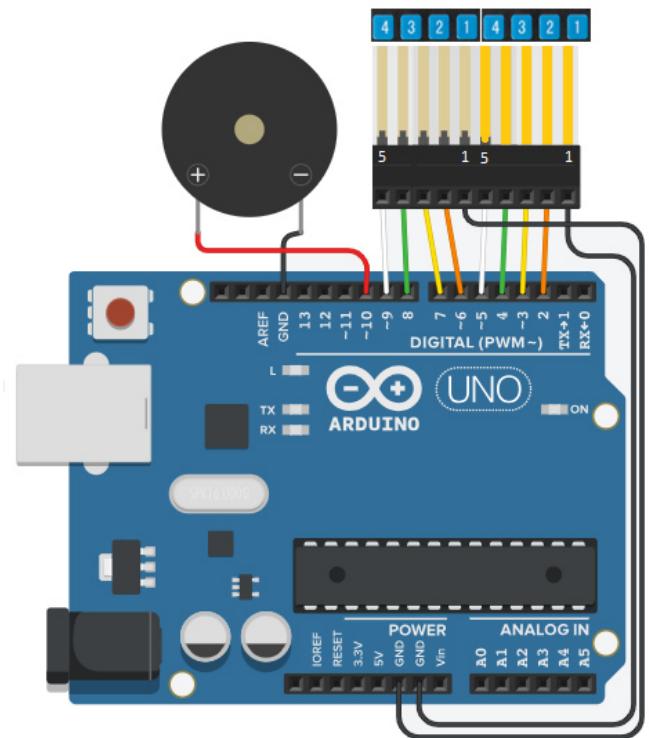
//use of parallel arrays: buttonPin, notes, freqnotes

void setup() {
    Serial.begin(9600);
    // initialize the keypad pin(s) as an INPUT_PULLUP, it reads LOW when
button pressed
    for(int x=0; x<8; x++)  {
        pinMode(buttonPin[x], INPUT_PULLUP);
    }
}

void loop(){

int buttonState; // variable to read current button State
// Read sequentially all the keys, if you read LOW, key is pressed
for(int i=0; i<8; i++){
    buttonState=digitalRead(buttonPin[i]);
    if(buttonState == LOW){
        Serial.println(notes[i]); // show the note
        tone(10,freqnotes[i],500); // play the note at buzzer, pin 10
    }
}
}

```



Παράδειγμα Έρευνας και Εργασίας



©Disney, "Gyro Gearloose"
«Κύρος Γρανάζης»



Για να αντιληφθεί κανείς την εργασία που γίνεται στο παρασκήνιο της δημιουργίας αυτού του έργου, παραθέτουμε ένα παράδειγμα για να δείξουμε τον τρόπο που κάναμε έρευνα, εργαστήκαμε και προσαρμόσαμε τις πληροφορίες στα δικά μας δεδομένα. Να υπενθυμίσουμε ότι κάνουμε Tinkering = επιδιορθώνω προχείρως, συνεπώς ψάχνουμε και πειραματίζόμαστε ώστε να καταλήξουμε σε σωστό αποτέλεσμα.

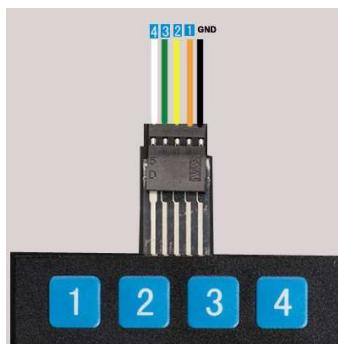
Η ιδέα είναι να κατασκευάσουμε ένα «πιάνο» με Arduino, piezo buzzer και 2 Membrane Keypads 1X4. Δεν βρήκαμε επαρκείς πληροφορίες (datasheet) για το εξάρτημα Membrane Keypad 1X4. Δεν γνωρίζαμε ποιό καλώδιο συνδέεται με ποιό Key και αν το πάτημα ενός Key παράγει HIGH ή LOW.

Βρήκαμε ένα tutorial στο AUTODESK Instructables:



<https://www.instructables.com/1x4-Membrane-Keypad-w-Arduino/>

Δοκιμάσαμε το πρόγραμμα και διαπιστώσαμε ότι η σειρά καλωδίων – Keys είναι η παρακάτω:



Επίσης, διαπιστώσαμε ότι τα pins που διαβάζουν τα Keys πρέπει να οριστούν ως INPUT_PULLUP. Αυτό σημαίνει ότι συμπεριφέρονται σαν να είναι μονίμως HIGH, και όταν πατήσουμε ένα Key, θα διαβάσουμε LOW στο αντίστοιχο pin.

Ο κώδικας από το AUTODESK Instructables:

```
/ Constants won't change. They're used here to
// set pin numbers:
const int buttonPin[] = {9,10,11,12};      // the number of the pushbutton pins
const int ledPin = 13;          // the number of the LED pin

// variables will change:
int buttonState = 0;           // variable for reading the pushbutton status

void setup() {
    // initialize the LED pin as an output:
    pinMode(ledPin, OUTPUT);
    // initialize the Serial Monitor @ 9600
    Serial.begin(9600);
    // initialize the keypad pin(s) as an input:
    for(int x=0; x<2; x++)
    {
        pinMode(buttonPin[x], INPUT_PULLUP);
    }
}

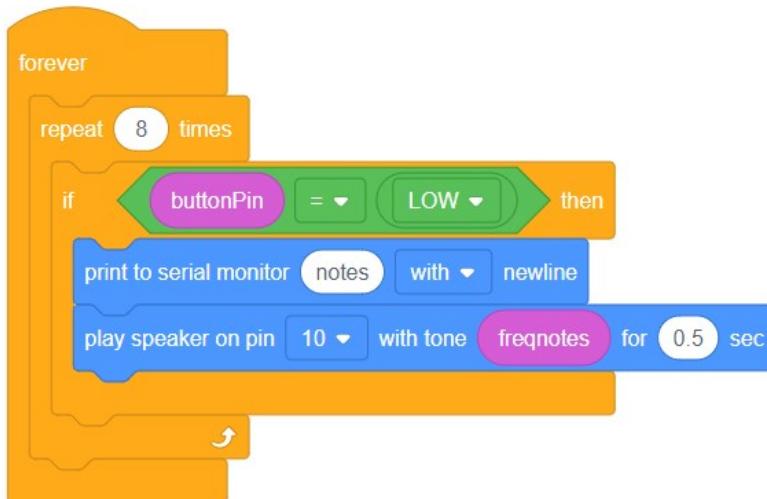
void loop(){
    // read the state of the keypad value:
    for(int x=0; x<2; x++)
    {
        //signifying the state of which the button is in by reading the appropriate pin
        #
        buttonState = digitalRead(buttonPin[x]);

        // check if the pushbutton on the keypad is pressed.
        // if it is, the buttonState is LOW:
        if (buttonState == LOW && buttonPin[x] == 9) {
            // turn LED off:
            Serial.print("OFF *");
            // digitalWrite(ledPin, LOW);
        }
        if (buttonState == LOW && buttonPin[x] == 10) {
            // turn LED off:
            Serial.print("ON **");
            //digitalWrite(ledPin, LOW);
        }
    }
}
```

Σημείωση: Δοκιμάζει μόνο 2 keys, στα pins 9 και 10.

Εμείς βελτιστοποιήσαμε το παραπάνω πρόγραμμα, τοποθετώντας σε έναν βρόχο 3 παράλληλους πίνακες, και ελέγχοντάς τους σειριακά, τον έναν μετά τον άλλο. Ελέγχουμε 2 Membrane Keypads 1X4.

```
const int buttonPin[] = {3,2,5,4,7,6,9,8};
const String notes[] = {"C4", "D4", "E4", "F4", "G4", "A4", "B4", "C5"};
const int freqnotes[]={262,294,330,349,392,440,494,523};
```



Παρουσίαση του προγράμματος για να γίνει αντιληπτό από τους μαθητές. Χρήση Blocks Tinkercad.

Ο κώδικας ακολουθεί τους κανόνες του **δομημένου προγραμματισμού**, είναι πιο συμπαγής και ευνόητος.

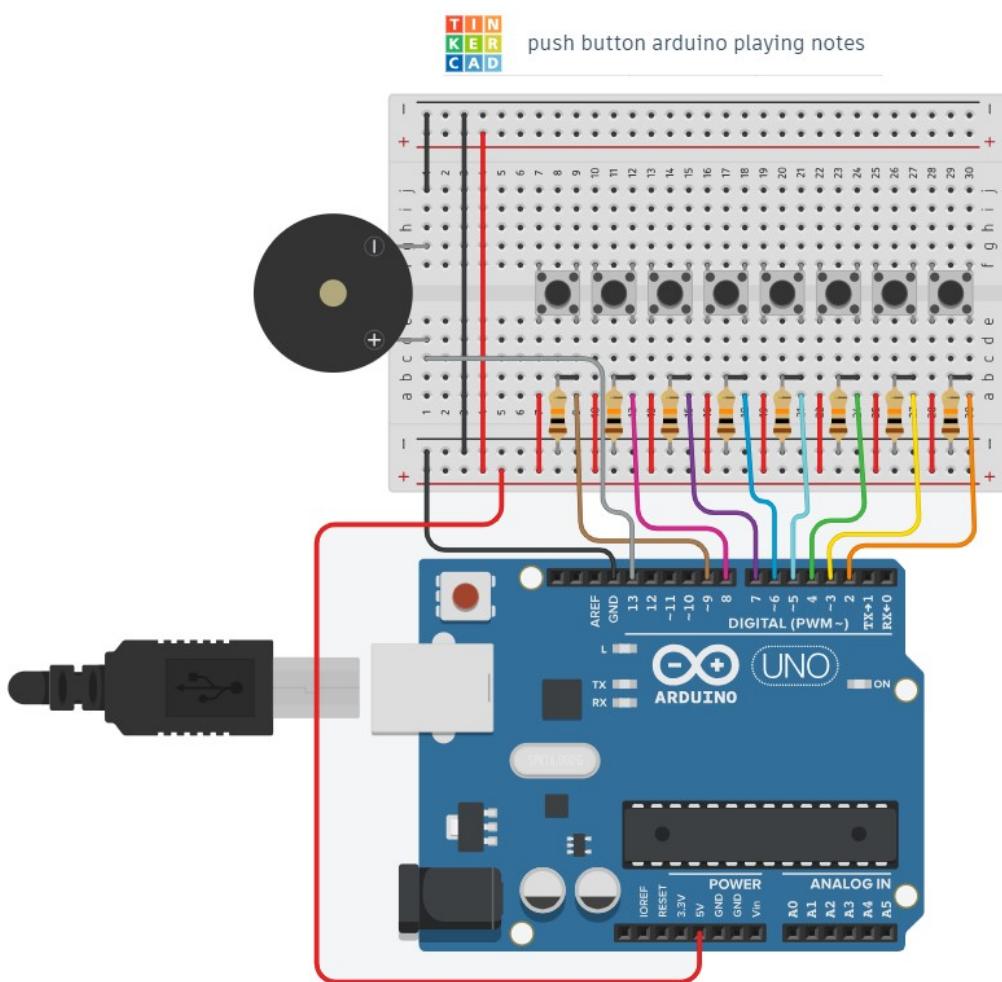
Ο κύριος βρόχος:

```
void loop(){
  int buttonState;
  for(int i=0; i<8; i++){
    buttonState=digitalRead(buttonPin[i]);
    if(buttonState == LOW){
      Serial.println(notes[i]);
      tone(10,freqnotes[i],500);
    }
  }
}
```

Βλέπε και Membrane Keypad Piano

Push Button Piano

Arduino



Αντιστάσεις 10kΩ

<pre> if [read digital pin 9] = [HIGH] then play speaker on pin [13] with tone [48] for [0.5] sec </pre> <p>8 IF μέσα σε ένα Forever</p>	Pin	Wire color	note	tone
	9	Brown	DO	48
	8	Pink	RE	50
	7	Purple	MI	52
	6	Blue	FA	53
	5	Turquoise	SOL	55
	4	Green	LA	57
	3	Yellow	SI	59
	2	Orange	DO+	60

Πατώντας το push button, κλείνουμε το κύκλωμα (επιτρέπουμε να περάσει ρεύμα). Αν διαβάσουμε HIGH παίζει η νότα στο buzzer. Buzzer ενωμένο στο pin 13 (gray).

Αναγνώριση Συχνότητας με FFT

Τι είναι το FFT;

Πληροφορίες βασισμένες στο:

[An Interactive Guide To The Fourier Transform – BetterExplained](#)

FFT = **F**ast **F**ourier **T**ransform = Γρήγορος Μετασχηματισμός Fourier

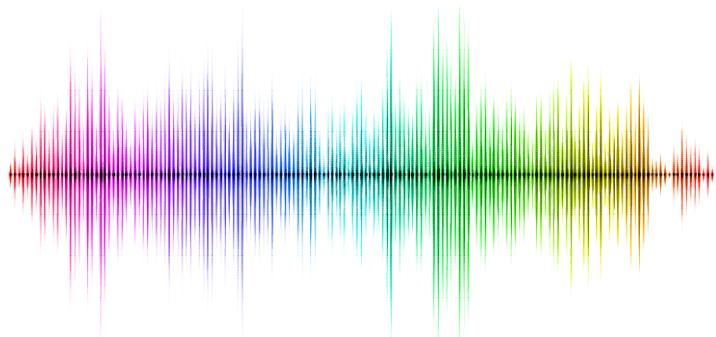
Περιγράφεται από τους παρακάτω μαθηματικούς τύπους:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi kn/N}$$

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k \cdot e^{i2\pi kn/N}$$

Όμως, θα το απλουστεύσουμε και θα δώσουμε ένα μεταφορικό παράδειγμα: Ο Μετασχηματισμός Fourier αν εφαρμόζονταν στη μαγειρική, θα μπορούσε, αν του δίναμε μια σούπα, να βρει τη συνταγή.

Ο Μετασχηματισμός Fourier εφαρμόζεται σε μοτίβα που βασίζονται σε χρόνο. Μετράει κάθε δυνατό κύκλο μέσα τους και μας δίνει τα μεγέθη όλων των κύκλων που βρίσκει μέσα στο μοτίβο. Ο ίχος έχει μοτίβο. Ο Μετασχηματισμός Fourier βρίσκει το κυρίαρχο μοτίβο, και κάνει αναγνώριση της βασικής συχνότητας μέσα σε αυτό.



Βρήκαμε ένα πρόγραμμα για Arduino που αναγνωρίζει την επικρατούσα συχνότητα του ήχου:

<https://projecthub.arduino.cc/lbf20012001/audio-frequency-detector-d300e3>

```
#include "arduinoFFT.h"

#define SAMPLES 128           //SAMPLES-pt FFT. Must be a base 2 number. Max 128 for Arduino Uno.
#define SAMPLING_FREQUENCY 2048 //Ts = Based on Nyquist, must be 2 times the highest expected frequency.

arduinoFFT FFT = arduinoFFT();

unsigned int samplingPeriod;
unsigned long microSeconds;

double vReal[SAMPLES]; //create vector of size SAMPLES to hold real values
double vImag[SAMPLES]; //create vector of size SAMPLES to hold imaginary values

void setup()
{
    Serial.begin(115200); //Baud rate for the Serial Monitor
    samplingPeriod = round(1000000*(1.0/SAMPLING_FREQUENCY)); //Period in microseconds
}

void loop()
{
    /*Sample SAMPLES times*/
    for(int i=0; i<SAMPLES; i++)
    {
        microSeconds = micros(); //Returns the number of microseconds since the Arduino board began running the current script.

        vReal[i] = analogRead(0); //Reads the value from analog pin 0 (A0), quantize it and save it as a real term.
        vImag[i] = 0; //Makes imaginary term 0 always

        /*remaining wait time between samples if necessary*/
        while(micros() < (microSeconds + samplingPeriod))
        {
            //do nothing
        }
    }

    /*Perform FFT on samples*/
    FFT.Windowing(vReal, SAMPLES, FFT_WIN_TYP_HAMMING, FFT_FORWARD);
    FFT.Compute(vReal, vImag, SAMPLES, FFT_FORWARD);
    FFT.ComplexToMagnitude(vReal, vImag, SAMPLES);

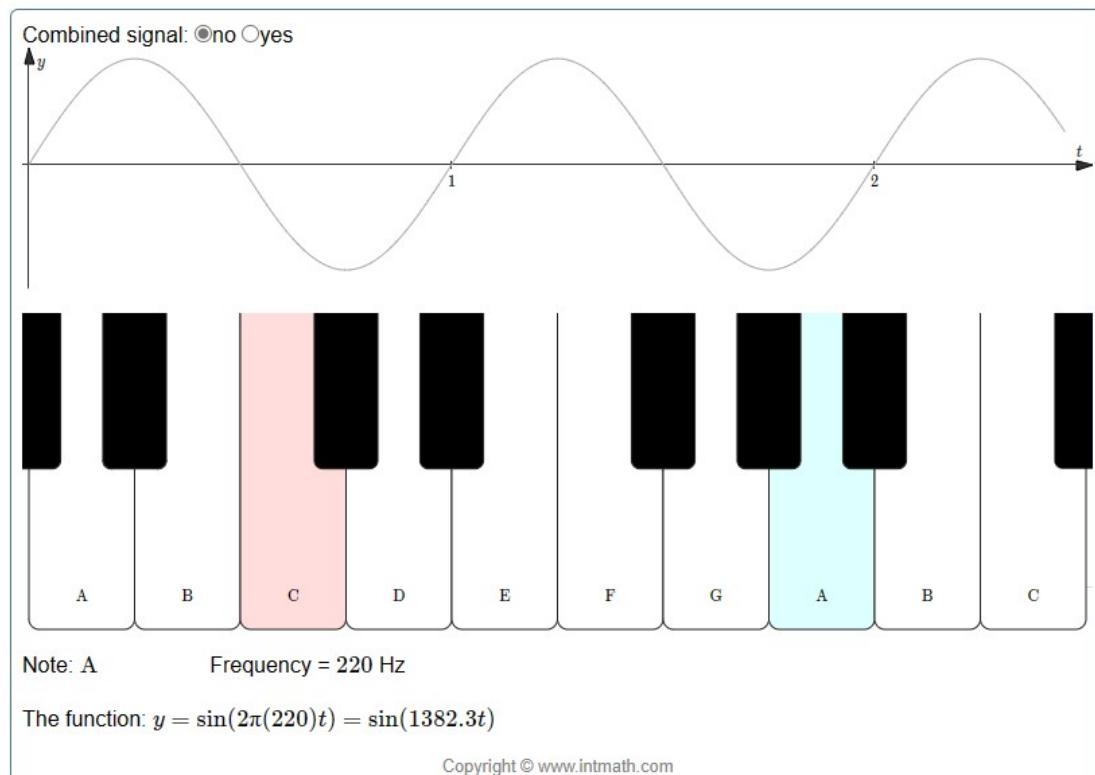
    /*Find peak frequency and print peak*/
    double peak = FFT.MajorPeak(vReal, SAMPLES, SAMPLING_FREQUENCY);
    Serial.println(peak); //Print out the most dominant frequency.
```

Θέτουμε στο COM to 115200 baud για να δουλέψει σωστά.

Πρέπει να πατάμε το RESET κάθε φορά. Μας βγάζει στον Serial Printer τη συχνότητα που αναγνώρισε.

Κάναμε δοκιμές με τη βοήθεια του παρακάτω εργαλείου:

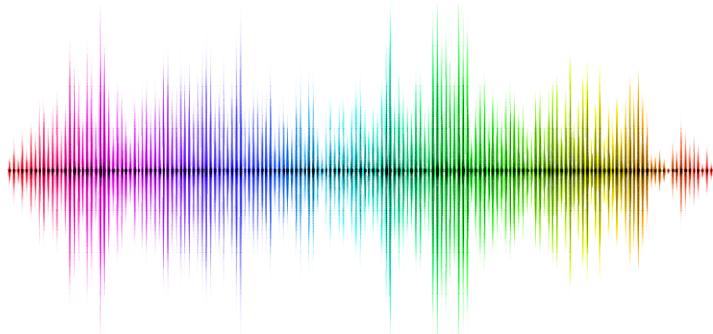
<https://www.intmath.com/trigonometric-graphs/music-note-frequencies-interactive.php>



Θα πρέπει να «εκπαιδεύσουμε»* το πρόγραμμα να «ακούει» τις νότες C4, D4, E4, F4, G4, A4, B4, C5 και να το τροποποιήσουμε ώστε κάθε φορά να βγάζει το αποτέλεσμα σε δυαδική μορφή πχ. σε pins και 7-segment display?? Και να το προωθεί στα στο αντίστοιχο servo motor.

* Σημείωση: Μια βασική τεχνική που χρησιμοποιείται σε εφαρμογές Τεχνητής Νοημοσύνης είναι ότι χρησιμοποιούμε απλούς αλγορίθμους, αλλά παρέχουμε μεγάλη ποσότητα δεδομένων, για να εκπαιδευτεί το σύστημα. (Artificial Intelligence A Modern Approach, Russel & Norvig).

Frequency Generators



Πρόγραμμα για PC:

<https://www.intmath.com/trigonometric-graphs/music-note-frequencies-interactive.php>

The screenshot shows a piano keyboard with keys labeled A through C. Above the keyboard, a sine wave graph oscillates between two points on a coordinate system. Below the graph, text indicates "Note: A" and "Frequency = 220 Hz". A note also states "The function: $y = \sin(2\pi(220)t) = \sin(1382.3t)$ ".

A QR code is displayed, which, when scanned, links to the interactive mathematics website where the piano and sine wave visualization can be found.

Εφαρμογή για Tablet (Frequency Generator):

<https://play.google.com/store/apps/details?id=com.luxdelux.frequencygenerator&hl=en>

The screenshots show the Frequency Generator app interface. It features a piano keyboard at the bottom, a waveform visualization in the middle, and various control buttons and frequency values (e.g., 370.05 Hz, 440 Hz, 1306.40 Hz) on the right side.

A QR code is displayed, which, when scanned, links to the Google Play Store page for the Frequency Generator app.

Εκπαιδεύοντας το Arduino για Μουσικό Όργανο

Arduino FFT music notes recognition Καταγραφή τιμών από Serial Monitor

Θέλουμε το Arduino να ακούει από το μικρόφωνο και να «μάθει» να αναγνωρίζει τις νότες από ένα Μουσικό Όργανο ή Frequency Generator που παίζουμε

C ₄	D ₄	E ₄	F ₄	G ₄	A ₄	B ₄	C ₅
----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------

Χρησιμοποιούμε το αρχείο *Registry Music Notes.xlsx* για να καταγράψουμε σε μητρώο (registry) 20 τιμές που «διαβάζει» το Arduino, όταν παίζουμε τις νότες C₄...C₅, σε κάποιο μουσικό όργανο που επιθυμούμε.

Πρόγραμμα για PC:

<https://www.intmath.com/trigonometric-graphs/music-note-frequencies-interactive.php>

Εφαρμογή για Tablet:

<https://play.google.com/store/apps/details?id=com.luxdelux.frequencygenerator&hl=en>

Arduino FFT music notes range							
	C ₄	D ₄	E ₄	F ₄	G ₄	A ₄	B ₄
	DO	RE	MI	FA	SOL	LA	SI
48 (C4 = 262 Hz)	50 (D4 = 294 Hz)	52 (E4 = 330 Hz)	53 (F4 = 349 Hz)	55 (G4 = 392 Hz)	57 (A4 = 440 Hz)	59 (B4 = 494 Hz)	60 (C5 = 523 Hz)
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							
20							
minimum	0	0	0	0	0	0	0
maximum	0	0	0	0	0	0	0
[..]	0	1	2	3	4	5	6

Registry Music Notes.xlsx

Οι δύο τελευταίες γραμμές θα εξάγουν αυτόματα την ελάχιστη και μέγιστη τιμή που καταγράψαμε. Αυτές τις τιμές θα εισάγουμε στους πίνακες `double minimum[8]` `double maximum[8]` στο πρόγραμμα *Ard_FFT_music_notes_registry_withArray.ino*, για να ρυθμίσουμε το Arduino στο επιθυμητό μουσικό όργανο.

Ολόκληρο το πρόγραμμα στην επόμενη σελίδα.

Ard_FFT_music_notes_registry_withArray.ino

arduinoFFT library version: 1.5.0 by Enrique Condes (Tools→Manage Libraries)

Serial baud rate: 115200

microphone at A0

Highlighted values have to change.

Ολόκληρο το πρόγραμμα : *Ard_FFT_music_notes_registry_withArray.ino*

arduinoFFT library version: 1.5.0 by Enrique Condes (Tools→Manage Libraries)

Serial baud rate: 115200 --- microphone at A0 Highlighted values have to change.

```
/* Notes: Copyright (c) 2019 by C. A. Lettsome Services, LLC
For more information visit https://clydelettsome.com/blog/2019/12/18/my-weekend-project-audio-frequency-detector-using-an-arduino */
#include <arduinoFFT.h>

#define SAMPLES 128          //SAMPLES-pt FFT. Must be a base 2 number. Max 128 for Arduino Uno.
#define SAMPLING_FREQUENCY 2048 //Ts = Based on Nyquist, must be 2 times the highest expected frequency.
arduinoFFT FFT = arduinoFFT();
unsigned int samplingPeriod;
unsigned long microSeconds;

double vReal[SAMPLES]; //create vector of size SAMPLES to hold real values
double vImag[SAMPLES]; //create vector of size SAMPLES to hold imaginary values

void setup()
{
    Serial.begin(115200); //Baud rate for the Serial Monitor
    samplingPeriod = round(1000000*(1.0/SAMPLING_FREQUENCY)); //Period in microseconds
}

void loop()
{
    /*Sample SAMPLES times*/
    for(int i=0; i<SAMPLES; i++)
    {
        microSeconds = micros(); //Returns the number of microseconds since the Arduino board began running the current script.

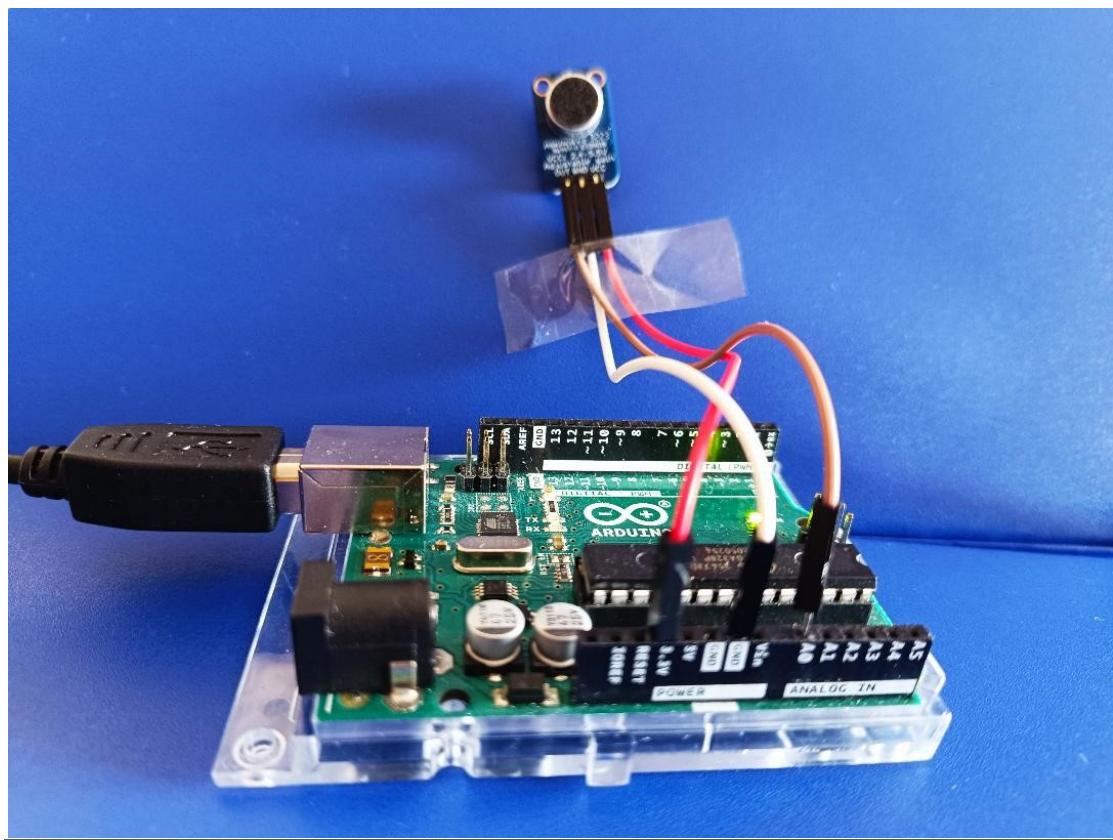
        vReal[i] = analogRead(0); //Reads the value from analog pin 0 (A0), quantize it and save it as a real term.
        vImag[i] = 0; //Makes imaginary term 0 always

        /*remaining wait time between samples if necessary*/
        while(micros() < (microSeconds + samplingPeriod))
        {
            //do nothing
        }
    }
    /*Perform FFT on samples*/
    FFT.Windowing(vReal, SAMPLES, FFT_WIN_TYP_HAMMING, FFT_FORWARD);
    FFT.Compute(vReal, vImag, SAMPLES, FFT_FORWARD);
    FFT.ComplexToMagnitude(vReal, vImag, SAMPLES);

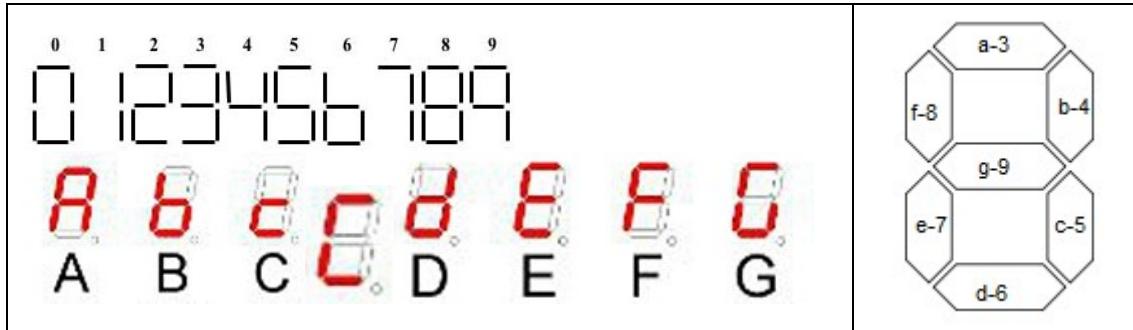
    /*Find peak frequency and print peak*/
    double peak = FFT.MajorPeak(vReal, SAMPLES, SAMPLING_FREQUENCY);
    double minimum[8];
    double maximum[8];
    // copy and paste registry values here
    //double minimum[] = {0,1,2,3,4,5,6,7};
    //double maximum[] = {0,1,2,3,4,5,6,7};
    Serial.println(peak); //Print out the most dominant frequency. ΑΥΤΟ ΚΑΤΑΓΡΑΦΟΥΜΕ!

    if (peak>minimum[0] and peak<maximum[0]){
        Serial.println("C4");
    }
    if (peak>minimum[1] and peak<maximum[1]){
        Serial.println("D4");
    }
    if (peak>minimum[2] and peak<maximum[2]){
        Serial.println("E4");
    }
    if (peak>minimum[3] and peak<maximum[3]){
        Serial.println("F4");
    }
    if (peak>minimum[4] and peak<maximum[4]){
        Serial.println("G4");
    }
    if (peak>minimum[5] and peak<maximum[5]){
        Serial.println("A4");
    }
    if (peak>minimum[6] and peak<maximum[6]){
        Serial.println("B4");
    }
    if (peak>minimum[7] and peak<maximum[7]){
        Serial.println("C5");
    }

    /*Script stops here. Hardware reset required.*/
    while (1); //do one time
}
```



7 Segment Display

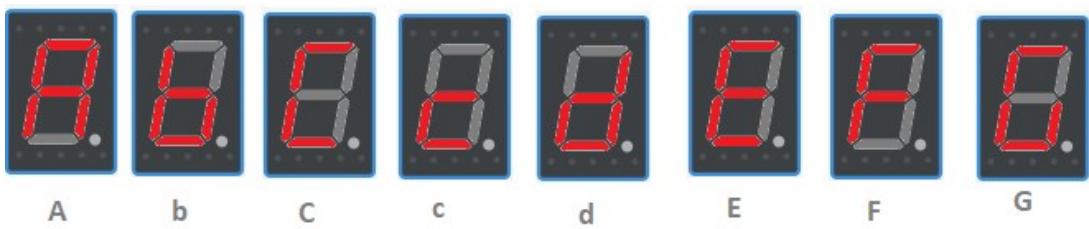
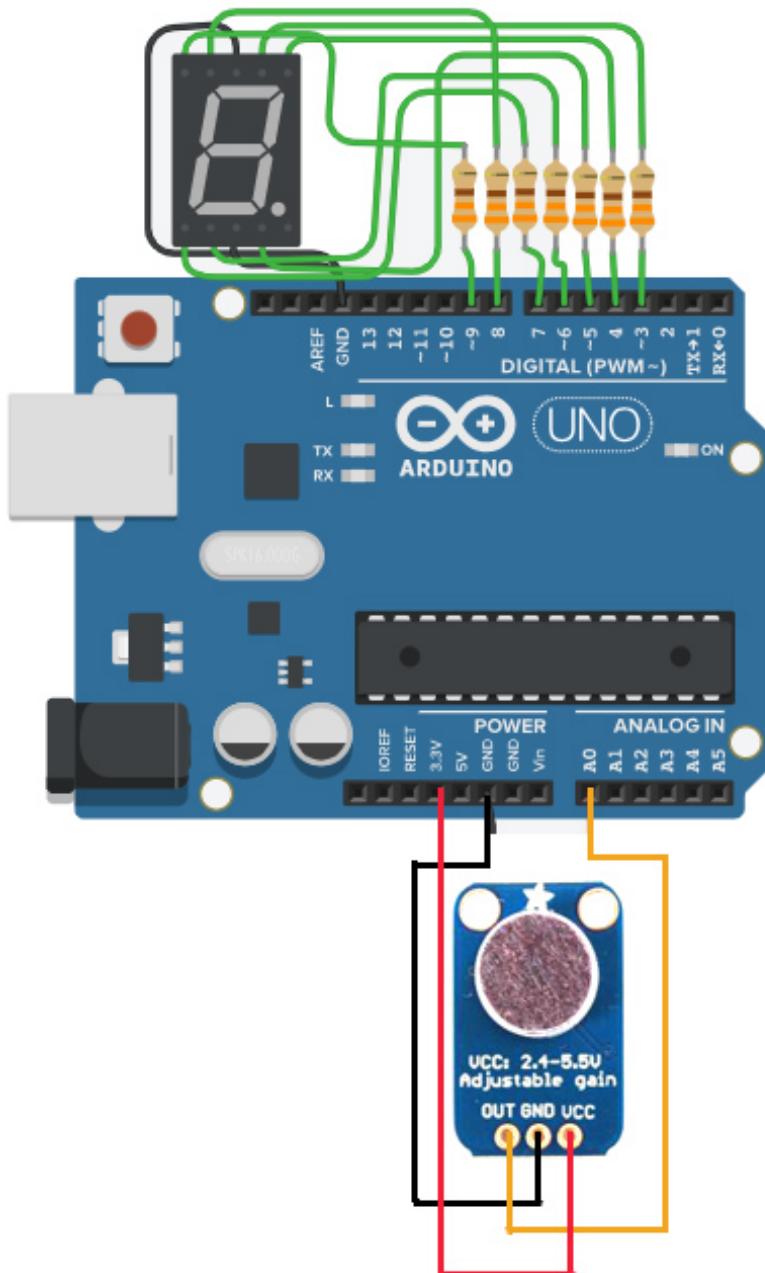


Προγραμμάτισε την οθόνη επτά τμημάτων για να δείχνει τα ψηφία. Γράψε **HIGH** για όποιο τμήμα είναι φωτεινό και **LOW** για όποιο τμήμα είναι σκοτεινό.

segment	a	b	c	d	e	f	g
Arduino Pin	3	4	5	6	7	8	9
0							
1							
2							
3							
4							
5							
6							
7							
8							
9							
A							
b							
c							
C							
d							
E							
F							
G							

Εικόνες από google εικόνες. Έχουν υποστεί επεξεργασία.

Περισσότερες πληροφορίες: http://en.wikipedia.org/wiki/Seven-segment_display



Αυτή είναι μια πρώιμη έκδοση της Κυμώς. Το μικρόφωνο ακούει τον ήχο, βρίσκει την επικρατούσα συχνότητα με FFT και αν είναι μέσα στην οκτάβα C4..C5, απεικονίζει τη νότα σε ένα seven segment display με γράμμα, όπως φαίνεται στον πίνακα στην επόμενη σελίδα.

Function lightSegment Arduino

Μπορούμε να χρησιμοποιήσουμε αυτό το Function για να μας δείξει το όνομα της νότας (συνδυάζουμε και το πρόγραμμα FFT).

Νότα	Συμβολισμός	
C4	C	
D4	D	
E4	E	
F4	F	
G4	G	
A4	A	
B4	B	
C5	C	

```

void setup() {
    // put your setup code here, to run once:
}

void loop() {
    // put your main code here, to run repeatedly:

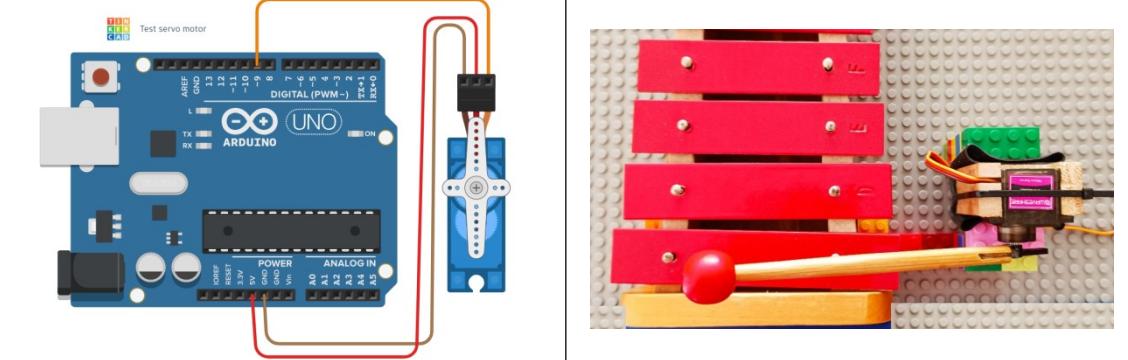
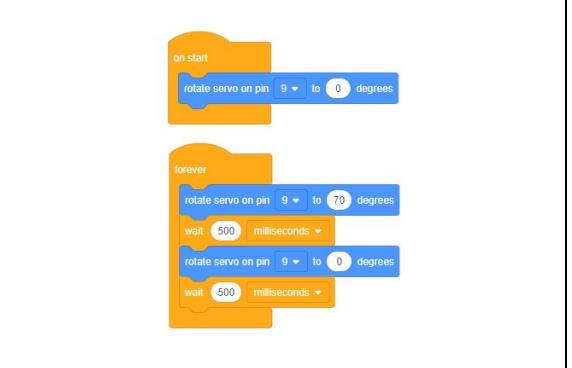
    lightSegment(0,0,0,1,1,0,1); //c
    lightSegment(0,1,1,1,1,0,1); //d
    lightSegment(1,0,0,1,1,1,1); //E
    lightSegment(1,0,0,0,1,1,1); //F
    lightSegment(1,0,1,1,1,1,0); //G
    lightSegment(1,1,1,0,1,1,1); //A
    lightSegment(0,0,1,1,1,1,1); //b
    lightSegment(1,0,0,1,1,1,0); //C
}

void lightSegment(int a, int b, int c, int d, int e, int f, int g){
    digitalWrite(3,a);
    digitalWrite(4,b);
    digitalWrite(5,c);
    digitalWrite(6,d);
    digitalWrite(7,e);
    digitalWrite(8,f);
    digitalWrite(9,g);
}

```

Arduino Test Servo Motor

Δοκιμή ενός servo motor

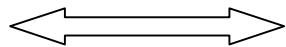
	
	<pre>#include <Servo.h> Servo servo_9; void setup() { servo_9.attach(9, 500, 2500); servo_9.write(0); } void loop() { servo_9.write(70); delay(500); // Wait for 500 millisecond(s) servo_9.write(0); delay(500); // Wait for 500 millisecond(s) }</pre>

I2C Arduino → micro:bit⁰

Πρόγραμμα για το Arduino FFT σε σήμα που εισάγεται στο A0, εύρεση επικρατούσας συχνότητας (peak) με FFT. Οι πίνακες minimum[8] και maximum[8] περιέχουν τις ελάχιστες και μέγιστες τιμές συχνότητας που έχουμε καταγράψει για τις νότες C4,D4,...,C5

Αν το peak βρίσκεται στο εύρος κάποιας νότας, τότε η μεταβλητή note παίρνει τιμή από 1 έως 8 και αποστέλλεται στο micro:bit.

minimum	maximum	noteLetter	note
267	269	"C4"	1
299	302	"D4"	2
336	338	"E4"	3
354	356	"F4"	4
399	401	"G4"	5
448	450	"A4"	6
502	504	"B4"	7
532	534	"C5"	8



εύρος

Κώδικας για Arduino

```
#include <arduinoFFT.h>
#include <Wire.h>

#define SAMPLES 128           //SAMPLES-pt FFT. Must be a base 2 number. Max 128 for Arduino Uno.
#define SAMPLING_FREQUENCY 2048 //Ts = Based on Nyquist, must be 2 times the highest expected frequency.

arduinoFFT FFT = arduinoFFT();
int note; //Global variable
unsigned int samplingPeriod;
unsigned long microSeconds;

double vReal[SAMPLES]; //create vector of size SAMPLES to hold real values
double vImag[SAMPLES]; //create vector of size SAMPLES to hold imaginary values

void setup()
{
    Wire.begin(13);           // join I2C bus with address #13
    Wire.onRequest(requestEvent); // register event
    Serial.begin(115200); //Baud rate for the Serial Monitor
    samplingPeriod = round(1000000*(1.0/SAMPLING_FREQUENCY)); //Period in microseconds
}

void loop()
{
    /*Sample SAMPLES times/
    for(int i=0; i<SAMPLES; i++)
    {
        microSeconds = micros(); //Returns the number of microseconds since the Arduino board began
running the current script.

        vReal[i] = analogRead(0); //Reads the value from analog pin 0 (A0), quantize it and save it as a
real term.
        vImag[i] = 0; //Makes imaginary term 0 always

        /*remaining wait time between samples if necessary*/
        while(micros() < (microSeconds + samplingPeriod))
        {
            //do nothing
        }
    }
}
```

```

/*Perform FFT on samples*/
FFT.Windowing(vReal, SAMPLES, FFT_WIN_TYP_HAMMING, FFT_FORWARD);
FFT.Compute(vReal, vImag, SAMPLES, FFT_FORWARD);
FFT.ComplexToMagnitude(vReal, vImag, SAMPLES);

/*Find peak frequency and print peak*/
double peak = FFT.MajorPeak(vReal, SAMPLES, SAMPLING_FREQUENCY);

//double minimum[8];
//double maximum[8];
// copy and paste registry values here
double minimum[] = {267,299,336,354,399,448,502,532};
double maximum[] = {269,302,338,356,401,450,504,534};
string noteLetter[] = {"C4", "D4", "E4", "F4", "G4", "A4", "B4", "C5"};

Serial.println(peak);      //Print out the most dominant frequency.

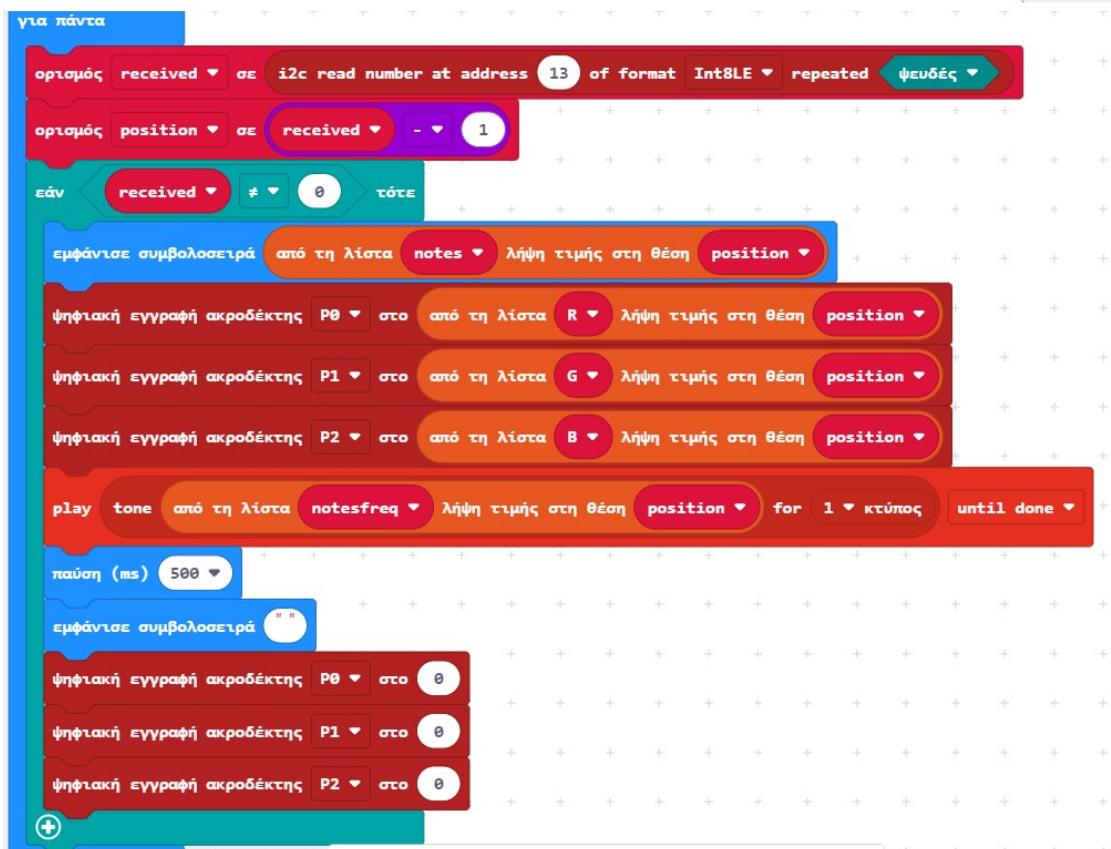
for (int i=0; i<8; i++){
    if (peak>minimum[i] and peak<maximum[i]){
        note=i+1;           //1..8
        Wire.write(note);
        Serial.println(noteLetter[i]);
    }
}

/*Script stops here. Hardware reset required.*/
while (1); //do one time
}

void requestEvent() {
    Wire.write(note);
}

```

Κώδικας για micro:bit



Δηλωμένοι πίνακες ([κατά την έναρξη](#)):

noteString	received	position	colors RGB	notefreq	R	G	B	note (5x5)
C4	1	0	blue	262	0	0	1	C
D4	2	1	green	294	0	1	0	D
E4	3	2	cyan	330	0	1	1	E
F4	4	3	red	349	1	0	0	F
G4	5	4	magenta	392	1	0	1	G
A4	6	5	yellow	440	1	1	0	A
B4	7	6	white	494	1	1	1	B
C5	8	7	blue	523	0	0	1	c

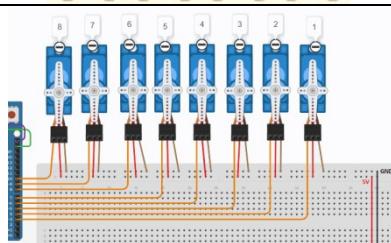
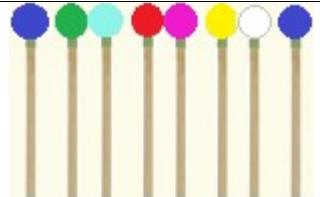
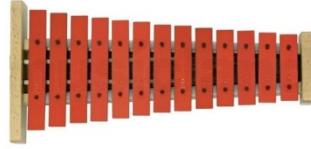
Το note που στέλνει το Arduino είναι η μεταβλητή received του micro:bit.

Το RGB led ανάβει με το αντίστοιχο χρώμα της νότας. Στο matrix 5x5 φαίνεται το λεκτικό της νότας note (5x5). Επίσης ακούγεται από το buzzer του micro:bit ο ήχος της νότας (μπορούμε και να το παραλείψουμε).

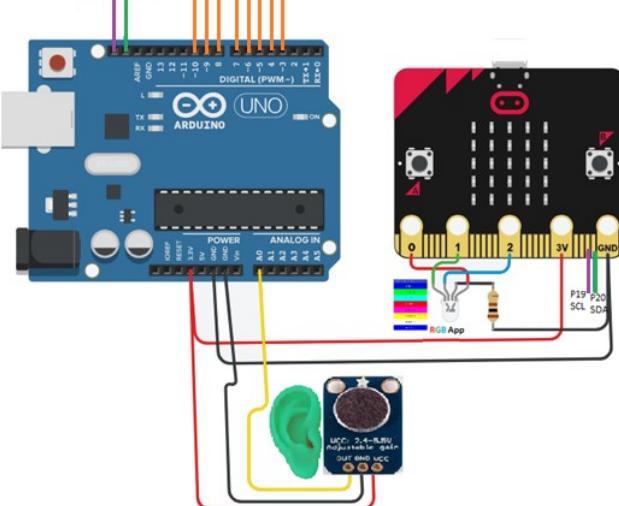
	I ² C ή IIC (Inter-Integrated Circuit): Δίαυλος (Bus) επικοινωνίας μεταξύ δύο ηλεκτρονικών συστημάτων. Απλός, αργός, φθηνός. Εφευρέθηκε το 1982 από την Philips Semiconductors.
---	--

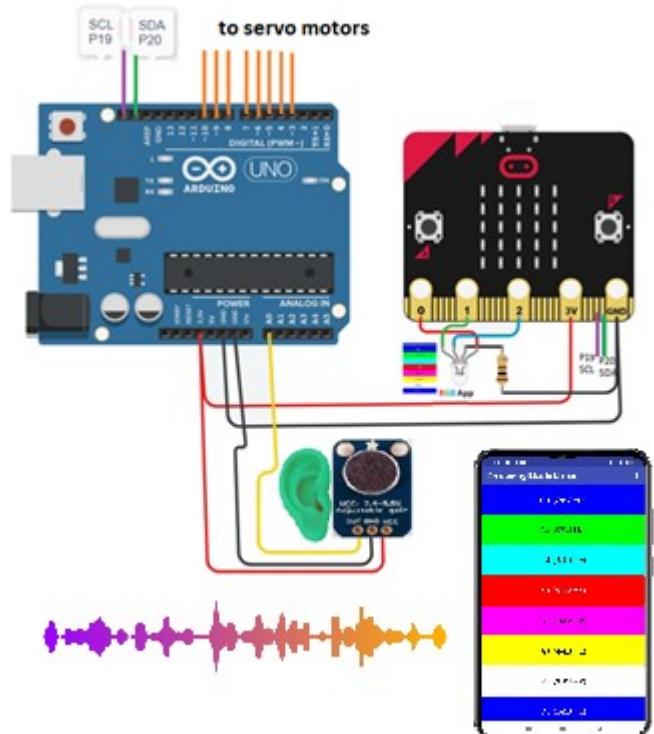


Η σύνθεση της Κυμώς, σχηματικά

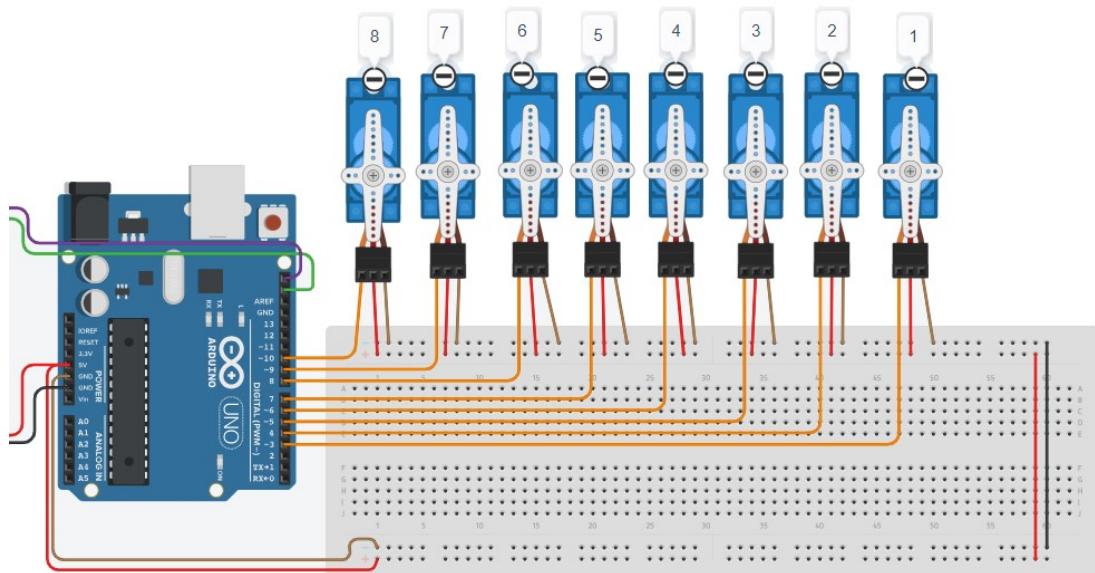


SCL P19 SDA P20 to servo motors





Arduino 8 Servo Motors



Κώδικας για νότα C4, παρόμοια και για τα υπόλοιπα

combined_FFT_Servos_I2C_copy_20240407161119.ino

```
#include <Servo.h>

Servo servo_3;
void setup()
{
    .
    .
    servo_3.attach(3, 500, 2500);
    .
    servo_3.write(0); // place to initial position

void loop()
{
    .
    .
    if (peak>minimum[0] and peak<maximum[0]){
        note=1;
        servopin=note+2; // servos attached to pins 3..10
        servo_3.write(75); // test degrees
        delay(100); // Wait for 100 millisecond(s) // test time
        servo_3.write(0);
        Serial.println("C4");
    }
}
```

note	Arduino Pin	servo
C4	3	servo_3
D4	4	servo_4
E4	5	servo_5
F4	6	servo_6
G4	7	servo_7
A4	8	servo_8
B4	9	servo_9
C5	10	servo_10