4. alkalom (zene)

Tematikai egység	Alkalmazott módszerek, munkaformák	Időtartam
A zene modul	Frontális tanári magyarázat	5 perc
Fülhallgató csatlakoztatása	Egyéni munka, tanári monitorozással	5 perc
A beépített dallamok kipróbálása	Egyéni munka, megbeszélés	15 perc
Hangok használata	Frontális tanári bemutató	10 perc
Gyerekdal leírása	Közös programírás	15 perc
Kedvenc dal leírása	Önálló munka	15 perc
Sziréna készítése	Közös programírás	10 perc
Kísérletezés, egyéni dallamok létrehozása	Egyéni munka, pármunka	15 perc

1. A zene modul, fülgallható csatlakoztatása

A micro:bit és a MicroPython zenei képességei is kiemelkedőek. Nagyon egyszerűen írhatunk dallamokat, játszhatunk le különböző hangeffekteket. Mivel az eszközön magán nincs hangszóró, ezért ennek, vagy egy fülhallgatónak a csatlakoztatásáról nekünk kell gondoskodnunk. Ehhez ismét a pinekhez kell nyúlnunk. A róluk szóló fejezetben látható módon, a 0 és a GND pinek segítségével csatlakoztathatjuk a hang kibocsátására használt eszközünket oly módon, hogy a krokodilcsipeszeket a jack dugó alsó és felső részéhez csatlakoztatjuk. Segítségünkre lehet az előző fejezetben látható kép, ahol krokodilcsipeszek helyett alufólia csíkokkal lett megoldva a csatlakozás, de a csipeszekkel is ugyanígy járjunk el!

2. A beépített dallamok kipróbálása

Ha csatlakoztattuk egymáshoz a két eszközt, teszteljük le, hogy sikerült-e! Gépeljük be az alábbi sorokat:

```
import music

music.play(music.NYAN)
```

Ha minden jól ment, hallanunk kell a megadott dallamot. Ez a zene modul egy beépített dallama, amiből több is áll rendelkezésünkre. A teljes lista megtalálható a dokumentáció zene modullal foglalkozó oldalán.[12] Vegyük észre, hogy a zene modult a randomhoz hasonlóan külön is importálnunk kell, ha használni szeretnénk, nem elég a már eddig használt import.

Feladat a diákok számára

Próbáld ki a zene modul beépített dallamait! A fenti kódban írt át a dallam nevét, majd hallgasd meg őket! Ezután beszéljétek meg, kinek melyik a kedvence! Hogyan használhatóak ezek a dallamok valamilyen célra, például jelzések, nyomok?

3. Hangok

De nem csak a beépített dallamok állnak rendelkezésünkre, sajátot is nagyon egyszerűen írhatunk. Minden hangnak van egy neve (például C# vagy F), egy oktávja (milyen magasan van az adott hang), és hossza (mennyi ideig tart a lejátszása). Az oktávot számmal jelöljük, 0 a legalacsonyabb oktáv, a 4-esben van a középső C, a 8-as pedig a legmagasabb amire szükségünk lehet, hacsak nem kutyáknak próbálunk zenét írni. A hosszt szintén egy szám jelöli, minél nagyobb, annál tovább tart az adott hang. Ezek arányosak egymással, például a 4-es hossz pontosan kétszer annyi ideig tart, mint a 2-es. Szünet is rendelkezésünkre áll, ezt R betűvel jelöljük (rest). Minden hang egy stringgel van jelölve, az alábbi módon:

`HANG[oktáv][:hossz]`

Például az `A1:4` az A jelű hang az 1-es oktávban, ami 4 hosszú ideig játszódik le.

Egy dallam lejátszásához ezeket a hangokat listába kell szerveznünk, akárcsak az animációknál tettük azt a képekkel. Példaként itt van a „János bácsi keljen fel” című gyerekdal a Pythonban:

```
import music

tune = ["C4:4", "D4:4", "E4:4", "C4:4", "C4:4", "D4:4", "E4:4
",        "C4:4", "E4:4", "F4:4", "G4:8", "E4:4", "F4:4", "G4
:8"]
music.play(tune)
```

[12] https://microbit-micropython.readthedocs.io/en/latest/music.html Elérés dátuma: 2018.08.04.

Elsőre kicsit olvashatatlannak tűnhet a dallam, valamint túl hosszú ideig kell gépelni. De ne aggódjunk, lehetőségünk van az egyszerűsítésre. A Python emlékszik az oktávra és a hosszra amíg meg nem változtatjuk azokat. Emiatt a fenti dallam egyszerűbben is leírható, a következő módon:

```
import music

tune = ["C4:4", "D", "E", "C", "C", "D", "E", "C", "E", "F",
        "G:8", "E:4", "F", "G:8"]
music.play(tune)
```

Vegyük észre, hogy az oktáv és az időtartam értékek csak akkor kerülnek feltüntetésre, amikor megváltoznak az előző hanghoz képest. Ezáltal sokkal kevesebbet kell gépelnünk, és a kódunk is egyszerűbben olvasható.

Feladat a diákok számára

Keresd meg interneten az énekórán tanult egyik kedvenc dalod kottáját! Játszd le a dal első pár sorát a micro:biten!

4. Sziréna készítése

De nemcsak zenei hangokat hozhatunk létre, egyéb hanghatásokat is használhatunk. Például egy sziréna hangját így:

```
import music

while True:
    for freq in range(880, 1760, 16):
        music.pitch(freq, 6)
    for freq in range(1760, 880, -16):
        music.pitch(freq, 6)
```

Ebben az esetben a `music` modul `pitch()` metódusát kell használnunk. Paraméteréül egy frekvenciát – például a 440-es érték felel meg az `A` hangnak –, valamint egy hosszt vár, ami azt jelöli, hogy hány ms-ig kell lejátszani a hangot.

A ciklusban a `range()` függvényt használtunk, amivel számsorozatokat generálhatunk. Ezeket a számokat használjuk a hang frekvenciájaként. A `range()` függvényt három paramétere a kezdeti érték, a végső érték, és a lépésköz. Tehát az első `range()` függvény egy számsorozatot generál 880 és 1760 között, 16-osával. A második pedig 1760 és 880 közötti számokat generál, -16-osával.

Mivel azt szeretnénk, hogy folyamatosan hallatszódjon a sziréna, ezért egy végtelen ciklusban helyeztük el a kódot. Az újdonság ezen belül található, egy újfajta ciklus, a `for`. Ezt úgy fogalmazhatnánk meg, hogy a megadott halmazban minden egyes elemmel csináljunk valamit. Ebben az esetben konkrétan a számsorozat minden egyes elemét adjuk meg frekvenciaként a `pitch()` metódusnak, és játsszuk le azt a hangot 6 ms-ig. Vegyük észre az igazítások jelentőségét, amivel már korábban is találkozhattunk.

Feladat a diákok számára

Kísérletezz a `for` ciklussal és a `pitch` metódussal! Milyen dallamokat tudsz így lejátszani? Mutassátok meg egymásnak a kész dallamaitokat!