8. alkalom (fájlkezelés)

Tematikai egység	Alkalmazott módszerek, munkaformák	Időtartam
A fájlkezelés alapjai	Közös megbeszélés	5 perc
A micro:bit fájlrendszere	Frontális tanári magyarázat	5 perc
Fájlok tartalmának olvasása	Közös programírás	10 perc
Fájlba írás	Közös programírás, önálló munka	15 perc
Fájlmanipulációs műveletek	Közös munka	10 perc
A microfs eszköz	Közös munka, önálló munka, csoportban	20 perc
Önálló feladatmegoldás	önálló munka, pármunka	25 perc

1. A fájlkezelés alapjai

Adatok tárolására rengeteg esetben szükség lehet. Beszéljük meg a diákokkal, hogy hogyan történhet ez! Valószínűleg eljutunk a számítógépes adattárolásig is, ezen belül is ahhoz, hogy hol tároljuk az adatokat, például pendrive, merevlemez. Ám a mi szempontunkból most érdekesebb az, hogy „miben”, milyen formában tároljuk az adatokat. Erre az adatbázis lehet egy logikus válasz a gyerekek részéről, de vezessük rá őket az egyszerűbb megoldásra, a fájlokra. Ezután előkerülhetnek olyan fogalmak, mint a fájlrendszer és a különböző fájlműveletek. Ezekkel fogunk most megismerkedni a MicroPython nyelv segítségével.

2. A micro:bit fájlrendszere

Egy programozási nyelv megismerésénél szinte mindig előjön a fájlkezelés témaköre, nincs ez másképp a MicroPythonnál sem. Az micro:bit beépített tárhelyének köszönhetően megvan a lehetőségünk a fájlok létrehozására és eltárolására. A tárhely mérete ugyan nem nagy, mindössze 30 kilobájt, néhány szöveges fájl tárolására azonban így is megfelelő. Fontos megjegyezni, hogy ez a tárhely nem egyenlő azzal a tárhellyel amit a számítógépen látunk, miután csatlakoztattuk hozzá az eszközt. Ez a mód csak a HEX fájlok másolására szolgál, nem is fogjuk itt látni a létrehozott fájlokat. Tudni érdemes még, hogy a fájlrendszer nem képes mappák kezelésére, minden fájl ugyanazon a helyen van tárolva, valamint, hogy a fájlok kikapcsolás után is megmaradnak. Maga a

Python nyelv rengeteg könnyen és jól használható módot kínál a fájlok kezelésére, ezekből jópár dolog implementálva lett a MicroPythonban is.

3. Fájlok tartalmának olvasása

A fájlok írása és olvasása az `open()` függvény segítségével valósítható meg. Egy fájl megnyitása után annak bezárásáig tudunk vele dolgozni. Minden esetben szükséges a fájl bezárása, hogy a program tudja, már nem akarunk vele dolgozni. A legjobb módszer ennek biztosítására, ha az ún. `with` kulcsszót használjuk, a következő módon:

```
with open("story.txt") as my_file:
    content = my_file.read()
print(content)
```

A `with` kulcsszó után az `open()` függvénnyel megnyitjuk a `story.txt` nevű fájlt, majd a `my_file` nevű objektumhoz rendeljük azt. A következő, beljebb igazított sorban a `content` változóba olvassuk be a fájl tartalmát, a `read()` metódus segítségével.

Figyeljük meg, hogy a következő sor már nincs beljebb kezdve. Ebből láthatjuk, hogy a `with` kulcsszóhoz melyik sorok tartoznak. Jelen esetben csak az a sor, ahol beolvassuk a fájl tartalmát. Amint a `with` blokk végére értünk, a Python automatikusan bezárja a fájlt, hiszen már nincs rá szükségünk. A `with` kulcsszóval jeleztük, hogy ezt a fájlt csak ebben a blokkban szeretnénk használni. Ez a fajta szerkezet ismerős lehet például a Java nyelvből, ahol a `try-with-resources` tölt be hasonló szerepet.

4. Fájlba írás

A fájlokat azonban természetesen nem csak olvasásra, hanem írásra is megnyithatjuk. Ehhez az `open()` függvényben kell jeleznünk a szándékunkat, a következő módon:

```
with open("hello.txt", 'w') as my_file:
    my_file.write("Hello, World!")
```

Figyeljük meg a `'w'` paramétert a függvény hívásában. Ez jelzi, hogy a fájlt `write` módban szeretnénk megnyitni. Akár az `'r'` paramétert is használhattuk volna az előbb, de mivel az olvasási mód az alapértelmezett, ezért ezt el szoktuk hagyni.

Az adatok írása a `write()` metódussal történik, aminek a paraméterébe kerül az a string, amit a fájlba szeretnénk írni. A fenti példában a „`Hello, World!`” szöveg íródik egy `hello.txt` nevű fájlba. Vigyáznunk kell azonban a fenti metódus működésével!

Amennyiben már létezik az a fájl, amibe írunk, a tartalma felülíródik. Amennyiben hozzáfűzni szeretnénk egy fájlhoz, be kell olvasnunk a tartalmát egy változóba, bezárni a fájlt, hozzáfűzni ehhez a változóhoz a kiírandó szöveget, megnyitni a fájlt írásra, végül pedig kiírni a változó tartalmát a fájlba. Ez eltérés a szokásos Python és a microPython között, mivel a Pythonban található ún. `append` mód is, amivel hozzáfűzhetünk egy fájlhoz. A lehető legegyszerűbb implementáció miatt azonban ez a funkció kimaradt a MicroPythonból. Legyen a diákok feladata az előző fájlhoz még egy mondat hozzáfűzése!

5. Fájlmanipulációs műveletek

Utaljunk vissza az óra eleji bevezetőre a fájlrendszerekről! Most az ezek által felkínált műveletekre fogunk visszatérni.

A fájlok olvasása és írása mellett lehetőségünk van még különböző egyéb manipulációs műveletek végrehajtására is, úgy, mint a fájlok listázása vagy törlése. A számítógépen ezt az operációs rendszer végzi, azonban hasonló funkcionalitás került a Pythonba is az `os` nevű modul által. Ezt a modult használva tudunk egy asztali operációs rendszer működéséhez hasonló műveleteket elvégezni. Alapvetően három műveletet tudunk végezni a fájlrendszerrel: kilistázni a fájlokat, törölni egy fájlt, valamint lekérdezni egy fájl méretét. Ezekhez természetesen mind szükség van az `os` modul importálására.

A fájlok listázása a `listdir()` függvénnyel történik. Ez visszaad egy stringekből álló listát, ami a fájlneveket tartalmazza.

```
import os
my_files = os.listdir()
```

Egy fájl törléséhez a `remove()` függvényt használhatjuk. Egy paramétere van, annak a fájlnak a neve, amit törölni szeretnénk.

```
import os
os.remove("filename.txt")
```

Végül, egy fájl méretének lekérdezésére szolgál a `size()` függvény. Ez egy egész számot ad vissza, a paraméterében megadott fájl által foglalt bájtok számát.

```
import os
file_size = os.size("filename.txt")
```

6. A microfs eszköz

Programjainkból tehát már tudunk létrehozni és olvasni is fájlokat, de mi a helyzet akkor, ha szeretnénk az eszközre másolni, vagy az eszközről lementeni azokat? Erre szolgál a `microfs` eszköz. Ezt akkor tudjuk használni, ha a Python telepítve van a számítógépünkre. A telepítés és a használat módját részletesen megtekinthetjük a dokumentációban[17], itt csak a legfontosabb funkciókat mutatom be.

Az eszköz telepítéséhez nyissunk parancssor, majd adjuk ki az alábbi utasítást:

```
pip install microfs
```

Ne feledjük, hogy ez csak akkor fog működni, ha a Python telepítve van a számítógépre!

Ezután használatba is vehetjük, a parancsokat minden esetben az `ufs` előtaggal kezdve.

A fájlok kilistázásához a Unixból már ismert parancsot használhatjuk:

```
ufs ls
```

Ha a micro:bitről szeretnénk lementeni egy fájlt, akkor egy FTP-ből ismerős parancshoz kell nyúlnunk:

```
ufs get story.txt
```

Ez a parancs lementi a story.txt fájlt a jelenlegi helyre.

Fájlt az `rm` paranccsal törölhetünk:

```
ufs rm story.txt
```

Végül, ha a számítógépről szeretnénk fájlt másolni a micro:bitre, az alábbi parancs szolgál segítségül:

```
ufs put story2.txt
```

Ezzel a pár egyszerű paranccsal a micro:bit egyszerű fájlrendszerét megfelelően tudjuk kezelni.

[17] https://microfs.readthedocs.io/en/latest/ Elérés dátuma: 2019. 01. 14.

Feladat a diákok számára

Hozz létre egy fájlt a saját neveddel, majd adjátok tovább a micro:biteket a valakinek a teremben! A megkapott micro:bitre másolj rá egy fájlt, szintén a saját neveddel! Adjátok még tovább néhányszor az eszközöket. Az utolsó ember írjon egy programot, ami kiírja az eszközön látható fájlok nevét! Próbáljátok meg rekonstruálni az eszköz által bejárt útvonalat!

7. Önálló feladatmegoldás

Most pedig következzen néhány, a csoport tudásáról függően önállóan, esetleg párban elkészítendő feladat.

Feladat a diákok számára

Hozz létre néhány fájlt különböző tartalommal! Számold meg, hogy hány bájt tárhelyet foglalnak az eszközön található fájlok összesen!

Egy lehetséges megoldás:

```
1.  import os
2.
3.  with open("hello.txt", 'w') as my_file:
4.      my_file.write("Hello, World!")
5.
6.  with open("hello2.txt", 'w') as my_file:
7.      my_file.write("Hello, Hello World!")
8.
9.  with open("hello3.txt", 'w') as my_file:
10.     my_file.write("Hello, Hello, Hello World!")
11.
12. files = os.listdir()
13. sizeSum = 0
14. for f in files:
15.     sizeSum += os.size(f)
16. display.scroll(str(sizeSum) + " b")
```

A fájlok létrehozása után a `files` változóba kerül az eszközön található fájlok listája. Az összesített méretnek létrehozunk egy változót, ennek kezdetben 0 az értéke. Ezután egy `for` ciklusra van szükségünk, amit ezúttal azért használunk, mert egy sorozat összes elemén végig kell menni. Az elemeken végigiteráló változót `f`-nek nevezzük el. A cikluson belül a méretet tartalmazó változó jelenlegi értékéhez hozzáadjuk az aktuális fájl méretét. Erre szolgál a több más programnyelvben is használatos `+=` operátor. Ennek

használatára a diákok előzetes ismereteinek megfelelően térjünk ki kisebb-nagyobb részletességgel. Végezetül kiírjuk a kijelzőre az összeget, hozzáfűzve a bájt jelét.

Feladat a diákok számára

Hozz létre néhány fájlt különböző tartalommal! Írd ki a kijelzőre a legtöbb karaktert tartalmazó fájlt nevét, és azt is, hogy hány karaktert tartalmaz! Figyelj arra, hogy az eszközön mindig megtalálható `main.py` fájlt ne vizsgáld!

Egy lehetséges megoldás:

```
1.  import os
2.
3.  with open("hello.txt", 'w') as my_file:
4.      my_file.write("Hello, World!")
5.
6.  with open("hello2.txt", 'w') as my_file:
7.      my_file.write("Hello, Hello World!")
8.
9.  with open("hello3.txt", 'w') as my_file:
10.     my_file.write("Hello, Hello, Hello World!")
11.
12. files = os.listdir()
13. files.remove("main.py")
14. longestName = ""
15. longest = 0
16. for f in files:
17.     with open(f, 'r') as my_file:
18.         content = my_file.read()
19.         if len(content) > longest:
20.             longestName = f
21.             longest = len(content)
22. display.scroll(longestName)
23. display.scroll(longest)
```

A fájlok létrehozásában a kreativitást a tanulókra bízom. A fájlok listáját ismét változóba mentjük, majd a listából eltávolítjuk a `main.py` fájlt, hogy csak az általunk létrehozott fájlokat vizsgáljuk. Ez a fájl minden eszközön megtalálható, magát az eszközre másolt programot tartalmazza (tehát jelen esetben a fenti kódot). A leghosszabb fájlnévnek és a fájl hosszának létrehozunk egy-egy változót a megfelelő kezdőértékekkel. Ezután az előző feladatban már látott módon bejárjuk a listát. A cikluson belül megnyitjuk az egyes fájlokat, beolvassuk a tartalmukat egy változóba, majd ennek a változónak a hosszát fogjuk összehasonlítani az eddigi legnagyobb értékkel. Erre szolgál a `len()` függvény,

ami megmondja egy stringről, hogy hány karakterből áll. Amennyiben egy új leghosszabb fájltartalmat találtunk, értékül adjuk a változóinknak a fájl nevét és a hosszát. A ciklus végén kiírjuk ezeket az értékeket.