Napok

Készítsünk függvényt `napok` néven, amelynek megadjuk egy nem szökőévben egy hónap sorszámát, és a függvény visszaadja, hogy hány napos az adott hónap! Például `napok(3)` függvényhívás esetén az eredmény 31 vagy `napok(6)` függvényhívás mellett a visszaadott érték 30.

Készítsünk főprogramot `nap_honap.py` néven, amely bekéri két hónap sorszámát, majd megadja az előbbi függvény segítségével, hogy az elsőként megadott hónap első napja és a másodikként megadott hónap első napja között hány nap telt el! Az eredményt az alábbi formában jelenítsük meg!

Minta:

```
Első hónap: 5
Második hónap: 10
Eltelt napok: 31+30+31+31+30=153
```

Szövegben

Készítsünk függvényt `szovegben` néven, amely egy szöveget és karaktert kap bemenetként, majd visszaadja egy listában azokat a karakterpozíciókat, ahol a szövegben a karakter előfordul. A visszaadott karakterpozíciók egy lista számai. Ha a szövegben nem szerepel a karakter, akkor adjunk vissza üres listát! Például `szovegben("ablak alatt”,”a”)` függvényhívás esetén a visszaadott érték `[1,4,7,9]`, vagy `szovegel(“ablak alatt”,”e”)` függvényhívás mellett a visszaadott érték [], tehát üres lista.

Készítsünk főprogramot `szovegben.py` néven, amely bekéri a függvényben szereplő szöveget és karaktert, majd meghívja a függvényt, eltárolja a kapott eredményt és kiírja azt.

Minta:

```
Szöveg: Programozás
Karakter: o
Előfordulás: [3,8]
```

Csak5ös

Készítsünk függvényt `csak5os` néven, amely egy egész számokból álló listát kap bemenetként, és kiszámítja majd visszaadja a listában szereplő, 5-tel osztható számok összegét! Ha a listában nincs 5-tel osztható szám, akkor adjon vissza 0-t! Például `csak5os({3,5,1,10,20})` függvényhívás esetén 35-t kapunk, vagy `csak5os([6,1,4,8])` függvényhívás mellett 0-t kapunk.

Készítsünk főprogramot `otososszeg.py` néven, amely bekér a függvényben szereplő lista számait egy sorban, szóközzel elválasztva, majd meghívja a függvényt, eltárolja a kapott eredményt és kiírja azt.

Minta:

```
Számok: 6 15 3 7 20 10
Az ötösök összege: 45
```

Leghosszabb

Készítsünk függvényt `legho` néven, amely egy szavakból álló listát kap bemenetként, és megadja a listában előforduló szót! Például `legho(["ablak", "ajto", "kilincs", "ajto"])` függvényhívás esetén a visszaadott érték “kilincs”. A listában legalább egy szó szerepel, tehát biztosan van leghosszabb! Ha több azonos hosszúságú leghosszabb szó van, akkor a legelső leghosszabbat adjuk vissza!

Készítsen főprogramot `hosszu.py` néven, amely bekéri a függvényben szereplő szavakat egy-egy szóközzel elválasztva, elkészíti a függvényhíváshoz szükséges listát, meghívja a függvényt, eltárolja a kapott eredményt és kiírja az eredményt!

Minta:

```
Szavak: ez nem lehet ennyire egyszeru
Leghosszabb: egyszeru
```

Minta2:

```
Szavak: erre vagy arra
Leghosszabb: erre
```

Minta3:

```
Szavak: buda
Leghosszabb: buda
```

Palindrom

Készítsünk függvényt `pali` néven, amely egy szövegről eldönti, hogy az elölről hátrafelé és hátulról előre olvasva ugyanaz-e? Például `pali("ABBA")` függvényhívás esetén True (igaz) értéket kapjunk, míg `pali("Bence")` függvényhívás mellett False (hamis) értéket adjon a függvény! Ha a szöveg üres, akkor adjon igaz értéket, hiszen egy üres szöveg mindkét irányból olvasva ugyanaz – semmi.

Készítsen főprogramot `odavissza.py` néven, amely bekér egy szöveget, majd válaszként kiírja, hogy palindrom vagy sem!

Minta:

```
Szöveg: indul a gorog aludni
Nem palindrom.
```

Minta2:

```
Szöveg: indul a gorog a ludni
Palindrom.
```

Elsőhöz képest

Készítsünk függvényt `elso_kette` néven, amely egy egész számokból álló listát kap bemenetként. A függvény készítsen két listát: az egyikben a lista első számánál kisebb, a másikban a lista első számánál nagyobb számok legyenek. Maga az első szám ne kerüljön egyik listába sem! A függvény adja vissza a két listát egy kételemű listába csomagolva. Például `elso_kette([3,5,10,4])` függvényhívás esetén a visszakapott kételemű lista `[[],[5,10,4]]`, vagy `elso_kette([6,1,4,8])` függvényhívás mellett `[[1,4],[8]]` -t kapunk.

Készítsen főprogramot `elso_kettevag.py` néven, amely bekéri a függvényben szereplő lista számait egy sorban, szóközzel elválasztva, majd meghívja a függvényt, eltárolja a kapott eredményt és kiírja azt.

Minta:

```
Számok: 6 8 3 7 2 1
Első előtt: [3, 2, 1]
Első után: [8, 7]
```

Angol szavak 2

Az `eng5000.txt` egyszerű szöveges állományban az interneten előforduló 5000 leggyakoribb angol szó szerepel. Minden sorban egy szó, majd szóközzel utána a szó gyakorisága. Az állományban gyakoriság szerint csökkenő sorrendben vannak a szavak, tehát a leggyakoribb a legelső, a második leggyakoribb a második stb.

Készítsen programot **angol2** néven, abban oldja meg az alábbi feladatokat! A megoldás során minden esetben jelezze, hogy milyen eredményt ír ki vagy milyen inputot vár a felhasználótól. Ehhez vegye figyelembe a feladat végén lévő mintát!

Feladatok:

1. Olvassa be a szöveges állomány szavait és számait, és tárolja el egy megfelelő adatsorozatban!
2. Kérjen be egy gyakoriság értéket, és keresse meg az első angol szót, amelynek gyakorisága meghaladja a bekért értéket!
3. Készítsen függvényt `ugras` néven, amelynek bemenete két egész szám, két gyakoriság, és a függvény megadja, hogy a nagyobb gyakoriságnak hány százaléka a kisebb gyakoriság. Például `ugras(120,40)` mellett a 40/120 = 0,33, amit szövegként és százalékban kell visszaadni egy tizedes jegyre kerekítve, tehát “33,3”.
4. Az előző függvényt felhasználva kérjen be a program két olyan szót, amely szerepel a szavak között, és számítsa ki, hogy hány százaléka a kisebb gyakoriságú szó gyakorisága a nagyobb gyakoriságú szónak! Ha a bemenet egyik szava nem szerepel a szavak között, akkor írja ki, hogy “Ez a szó nem szerepel: budapest”.

Minta:

```
Gyakoriság: 14000000
Az első ezt meghaladó szó: nec


Első szó: program
Második szó: code
Gyakoriságuk aránya: 81,6%
```