**A csoport**

A feladatokat külön modulokban oldd meg, az modulok nevei a részfeladatokban találhatók! A fő programból (main.py) hívd meg az egyes modulok eljárásait. A projektet aCsoport néven mentsd, majd a munkádat saját\_neved.zip (Pl.: Nagy\_Virág.zip) állományban add le! (4p)

**1. feladat:**    **(könnyű) 7p konnyuA.py**

**minta:**

**I/a,b:**

**Adja meg a(z) 1. negatív egész számot: 4**

**Adja meg újra a(z) 1. negatív egész számot: -5**

**Adja meg a(z) 2. negatív egész számot: 0**

**Adja meg újra a(z) 2. negatív egész számot: -56**

**Adja meg a(z) 3. negatív egész számot: -3**

**I/c:**

**A legkisebb négyzetű szám: -3, megadási sorrendje: 3.**

a.       Írj egy függvényt (beker() néven), ami a minta alapján alkalmas három negatív egész szám bekérésére! (2p)

b.       Ha a felhasználó nem jó adatot adott meg, kérd be újra, míg jót nem ad meg! (1p)

c.       A program az adatbekérés után megkeresi, hogy a három beolvasott valós szám **négyzete** közül melyik a legkisebb, és hányadikként adta meg a felhasználó! A döntés eredményét a program írja ki a konzolra, a minta alapján! (4p)

**2. feladat:**     **(közepes) 13p sorozatA.py**

**minta:**

**II/a, c:**

**11\* 27\* 123\* 15\* 15\* 15\* 171\* 149\* 29\* 13\* 65\* 46\* 13**

**II/b:**

**Kétjegyűek száma: 9.**

**ketjegyu.txt** tartalma**:**

**Kétjegyűek száma: 9.**

a.       Írass ki a konzolra 13 páratlan számból álló véletlen számsorozatot [40,150] zárt intervallumon. A generált értékeket tárold lista adatszerkezetben. (3p)

b.       Számold meg, hogy hány kétjegyű van! (2p)

c.       Az értékeket csillaggal (\*) elválasztva írd ki a minta szerint (csak az elemek között van elválasztójel). (1p)

d.       Szervezd úgy a kódodat, hogy a számok előállítását egy függvény végzi, aminek visszatérési értékével (lista) dolgozik tovább a b. feladat (újabb metódus). Bemenő paramétere a generált számok darabszáma (itt 13) legyen. (3p)

e.       A b. feladat paramétere legyen az a. feladatból visszakapott érték (lista).(2p)

f.        Írasd ki a b. feladat megoldását  a ketjegyu.txt nevű fájlba is.(2p)

**3. feladat:**    **(nehéz)  16p stadion1A.py**

A stadionok.txt forrásállomány, Forma-1-es verseny pilótáinak adatait tartalmazza, a feladatok megoldása során ezeket az adatokat használd!

A stadionok.txt állomány szerkezete:

·         a stadion neve: Pl.: Metropolitan Park

·         a stadion helyszínének városa: Pl.: New York

·         a stadionnak hányas csapata: Pl.: 1

·         mikor léptek előszőr pályára: pl.: 1984-05-13

·         mikor léptek utoljára pályára: pl.: 1985-08-23

minta:

**III/b:**

**A csapatok darabszáma: 40**

**III/c:**

**New York-i stadionok, csapatszámaikkal:**

**Citi Field - 6**

**Hilltop Park - 11**

**Metropolitan Park - 1**

**Polo Grounds I (Southeast Diamond) - 9**

**Polo Grounds II (Southwest Diamond) - 1**

**Polo Grounds III - 3**

**Polo Grounds IV - 22**

**Polo Grounds V - 60**

**Shea Stadium - 48**

**St. George Cricket Grounds - 3**

**Yankee Stadium I - 84**

**Yankee Stadium II - 6**

**III/d:**

**A szeptemberi kezdő mérkőzések száma: 1 db.**

a.       Olvassa be a stadionok.txt fájlból az a stadionok adatait és tárolja el összetett adatszerkezetben, ami elősegíti a további feladatok könnyű megoldását! Ügyelj arra, hogy az állományok első sora az adatok fejlécét tartalmazza! (6p)

b.       Írassa ki a csapatok darabszámát a mintának megfelelően a konzolra!(2p)

c.       Határozza meg és írja ki a minta szerint, hogy mely csapatok találhatók New York-ban! A kiírásban a stadion neve és a csapatok száma szerepeljen! A megjelenítésnél nem kell rendezést alkalmazni! (4p)

d.       Számolja meg, hány olyan csapat van, ami szeptemberben játszotta az első mérkőzését. A program akkor is működjön, ha más hónappal helyettesítjük. (4p)