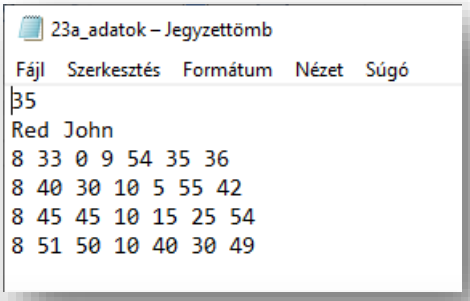
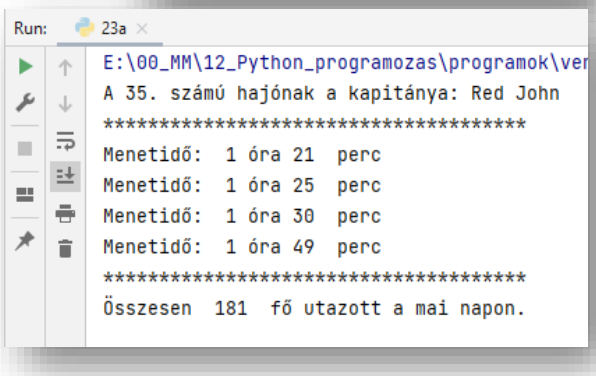
KOMPLEX FELADATOK MEGOLDÁSA

(23a.py)

Készítsük el a következő összetetteb feladatot!
Van egy hajótársaság, akik kiránduló járatokat indítanak csütörtöktől vasárnapig, a tengeren az egyik városból a másikba. Van egy 23a_adatok.txt nyersanyag. Ebben a példában nem homogén adatok vannak a szövegfájlban. Az első sorban a hajójárat számát találjuk, a másodikban a kapitány nevét. Aztán a harmadik sortól viszont már szóközökkel elválasztott számokat találunk. Az első az az indulás órája, majd a perce és másodperce; a negyedik adat a sorban az az érkezés órája, perce és másodperce. Az utolsó adat a sorban az utazók száma!
Tehát a harmadik sorban a csütörtöki adatok, az utolsóban a vasárnapi adatok találhatóak.
A feladatunk az, hogy a minta szerint az elején kiírja (fájlból kiolvasva), hogy hányas számú hajónak, ki a kapitánya!
Majd, hogy melyik napon, mennyi volt a menetidő.
Végül, hogy hányan utaztak a hajóval ezen a héten!





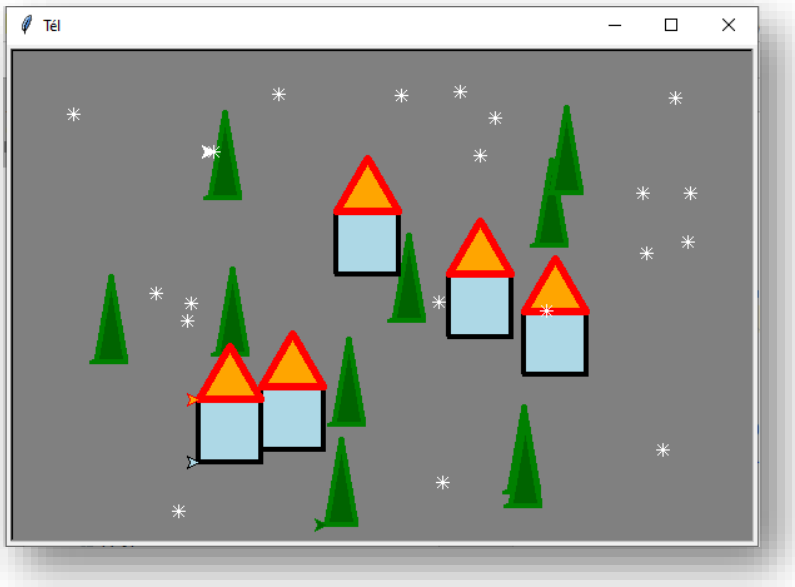
- A program elején létrehozunk egy „t” nevű üres tömböt. (1)
- A járat számát tároló „j_szam” változót, amit lenullázunk. (2)
- Egy „j_kap nevű változót, amit sztringre állítunk, mert ebben a kapitány nevét fogjuk tárolni. (3)
- Majd megnyitjuk olvasásra a nyersanyagot! (4)
- A következő két sorban beletesszük a megfelelő változókba a forrás első két sorát! Ehhez a readlin().strip() utasítást alkalmazzuk. (5-6)
- Aztán indítunk egy ciklust melyben soronként elhelyezzük az adatokat a „t” tömbbe, szóközök és sorvégi bekezdésjel nélkül. Erre a sor.strip().split() utasítást alkalmazzuk. (8)
- A következő 12 sorban létrehozunk egy szótár típust.
- Melyben először megadjuk a szótár nevét. (9).

```
23a.py
1  t=[]
2  j_szam=0
3  j_kap=str()
4  with open('./nyersanyag/23a_adatok.txt', 'r', encoding='utf-8') as forras:
5      j_szam=forras.readline().strip()
6      j_kap = forras.readline().strip()
7      for sor in forras:
8          t.append(sor.strip().split())
9  adat={}
10 adatok=[]
11 for elem in t:
12     hajo_adatok = elem
13     adat['ind_ora'] = int(hajo_adatok[0])
14     adat['ind_perc'] = int(hajo_adatok[1])
15     adat['ind_mp'] = int(hajo_adatok[2])
16     adat['erk_ora'] = int(hajo_adatok[3])
17     adat['erk_perc'] = int(hajo_adatok[4])
18     adat['erk_mp'] = int(hajo_adatok[5])
19     adat['szem_szam'] = int(hajo_adatok[6])
20     adatok.append(adat)
21     adat={}
22 print('A ',j_szam,'. számú hajónak a kapitánya: ',j_kap, sep='')
23 print('**************************************')
24 for adat in adatok:
25     print('Menetidő: ',int(abs((adat['ind_ora']*60+adat['ind_perc'])
26         -(adat['erk_ora']*60+adat['erk_perc']))/60),'óra',
27         int(abs((adat['ind_ora']*60+adat['ind_perc'])
28         -(adat['erk_ora']*60+adat['erk_perc']))%60),' perc')
29 print('**************************************')
30 ossz=0
31 for adat in adatok:
32     ossz=ossz+adat['szem_szam']
33 print('Összesen ',ossz,' fő utazott a mai napon.')
```

- Majd egy üres tömbre is szükségünk lesz ennek a neve „adatok” lesz. (10)
- Aztán elindítunk egy ciklust amiben megadjuk a címkéket, másnéven megnevezéseket és hozzárendeljük az értékeket. (11)
- A „t” tömbünkben lévő – most még - sorokban megyünk végig és bevezetünk egy ideiglenes változót „hajo_adatok” néven.
- Tehát az „elem”-ben a legelején a „t” tömb első sorát találjuk. Ami a txt harmadik sora és a számok vannak benne. (Most még szövegként.) (12)
- „Felcímkézzük” a különböző „mezőinket”, és számmá alakítjuk int() függvénnyel az indexelt adatainkat. Pl.: adat['ind_ora'] = int(hajo_adatok[0]) (13-19)
- Végül hozzáadjuk az adatok tömbünkhöz a feldolgozott rekordjainkat: adatok.append(adat); és üritjuk az adat{}-ot. (20-21)
- Az első alfeladatunkban kiíratjuk mondatszerűen, hogy melyik hajónak, ki a kapitánya. Ehhez a program elején elkészített két változót használjuk fel. A txt-nk első két sorát íratjuk ki tulajdonképpen. (22)
- Aztán egy elválasztó vonalat íratunk ki a minta szerint.
- A második alfeladatban azt kell kiszámolnunk, hogy mennyi volt a menetidő a négy napon. Ehhez a indulás órájára, percére, és az érkezés órájára és percére van szükségünk. A másodpercekre nincs is szükségünk. Tulajdonképpen egy hosszú képletet kell készítenünk. Lehetne különböző ideiglenes változókat is létrehozni, de mi válasszuk azt a módszert, hogy teljes képletet készítünk több sorban.
- Úgy gondolkodunk, hogy percekkel számolunk, ezért az órákat megszorozzuk 60-al és hozzáadjuk a perceket. Mind az indulásnál mind az érkezésnél. Majd kivonjuk az egyiket a másikból.
- Alkalmazom az abs() függvényt, hogy semmi képpen ne kapjunk negatív számokat.
- Tehát a képlet: abs((ind_óra*60+ind_perc)-(erk_ora*60+erk_perc)). Ennek az eredményét percben kapjuk meg. Viszont a kiíratáshoz vissza kell térnünk az óra és percre. Úgyhogy a kapott percet elosztjuk 60-al és az egész részét felhasználjuk órának, a maradék részét pedig percnek.
- És ezt mind mondatba formázzuk a minta szerint. (24-28)
- Ismét választóvonalat szúrunk be!
- A harmadik részfeladat nagyon könnyű, hiszen az a kérdés, hogy a négy nap alatt hányan utaztak összesen a hajóutakon. Ezt pedig a txt-nk utolsó oszlopában szerepló számok összeadásával kapjuk meg. Ez a szótár típusunk utolsó mezője a „szem_szam” adatainak összeadásából jön ki. (30-33)
- Teszteld, ellenőrizd, javítsad a programot írás közben!

(23b.py)

Ebben az összetett feladatban visszatérünk a grafikus felülethez. Ezen kívül az eljárások használatához. A példában a jobb oldalon látható „téli tájat” kell elkészítenünk.
Tehát, egy 600*400-as szürke felületre kell elhelyeznünk véletlenszerű helyre 10 db fenyőfát 5 db házat és 20 db hópelyhet!
A házak fekete keretű, világoskék kitöltésű négyzetből, és egy piros keretű, narancssárga kitöltésű háromszögből álljon!
A „fenyőfák”, zöld rajzolószínnel és sötétzöld kitöltőszínnel legyenek megrajzolva!
A hópelyhek fehérek legyenek!



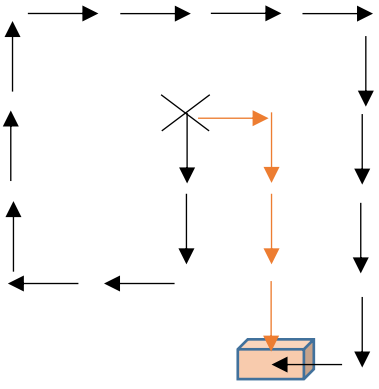
- A program elején beállítjuk az ablakunk tulajdonságait! Nevet adunk az ablaknak, megadjuk a háttérszínt, a címsor feliratát és az ablak méretét! (3-6.)
- Aztán négy „teknőst” állítunk be, az előző oldalon leírtak alapján, hogy a későbbiekben fel tudjuk használni őket. (8-25.)
- Majd elkészítjük az eljárásokat, amelyek megrajzolják a különböző alakzatokat! Ezek a négyzet, a háromszög, a hópihe, és a fenyőfa! (27-67.)
- Végül kirajzoltatjuk véletlen koordináták helyére először a 10 db fenyőfát, majd az 5 db házat, végül a 20 db hópehelyt! (69-83.)

```python
import turtle
from random import *
win = turtle.Screen()
win.bgcolor("gray")
win.title("Tél")
win.setup(600,400,0,0)

teki_1 = turtle.Turtle()
teki_1.color("black")
teki_1.pensize(4)
teki_1.fillcolor("lightblue")

teki_2 = turtle.Turtle()
teki_2.color("red")
teki_2.pensize(6)
teki_2.fillcolor("orange")

teki_3 = turtle.Turtle()
teki_3.color("white")
teki_3.pensize(1)

teki_4 = turtle.Turtle()
teki_4.color("green")
teki_4.pensize(4)
teki_4.fillcolor("darkgreen")

def negyzet(a,x,y):
    teki_1.up()
    teki_1.goto(x,y)
    teki_1.down()
    teki_1.begin_fill()
    for i in range(4):
        teki_1.forward(a)
        teki_1.left(90)
    teki_1.end_fill()

def haromszog(a,x,y):
    teki_2.up()
    teki_2.goto(x, y+a)
    teki_2.down()
    teki_2.begin_fill()
    for i in range(3):
        teki_2.forward(a)
        teki_2.left(120)
    teki_2.end_fill()

def ho(b,xb,yb):
    teki_3.up()
    teki_3.goto(xb,yb)
    teki_3.down()
    for i in range(8):
        teki_3.forward(b)
        teki_3.backward(b)
        teki_3.left(45)

def fa(z,xz,yz):
    teki_4.up()
    teki_4.goto(xz,yz)
    teki_4.down()
    teki_4.begin_fill()
    teki_4.forward(2*z)
    teki_4.left(100)
    teki_4.forward(5*z)
    teki_4.left(160)
    teki_4.forward(5*z)
    teki_4.left(100)
    teki_4.end_fill()

for i in range(10):
    xz=randrange(-280,240,1)
    yz = randrange(-180,100,1)
    fa(14,xz,yz)

for i in range(5):
    x=randrange(-200,200,10)
    y = randrange(-180,180,10)
    negyzet(50,x,y)
    haromszog(50,x,y)

for i in range(20):
    xb=randrange(-280,280,1)
    yb = randrange(-180,180,1)
    ho(5,xb,yb)

win.mainloop()
```

A feladatban a méretek és koordináták változhatnak. Az a lényeg, hogy a végeredmény hasonlítson az eredeti képernyőtervhez!
A program végén használjuk az ablak bezárására váró, mainloop() parancsot!

(23c.py)

A következő egyszerű példa a „kincskereső” feladat.
Melyben bekérjük karakterenként, hogy a kiindulási ponttól északra (E), délre (D), keletre (K), és nyugatra (N) mennyi egységet kell lépni! Majd a program megadja, egyszerűsítve a kincs helyét!



```
Run: 23c
E:\00_MM\12_Python_programozas\programok\venv\Sc
Kérem a robot parancsait: EEKEKEKKKNNDDDND
KK
Process finished with exit code 0
```

```
Run: 23c
E:\00_MM\12_Python_programozas\programok\ven
Kérem a robot parancsait: EEKNNNNDO
Nem megfelelő kód!
NNNE
Process finished with exit code 0
```

A képernyőkép mintája szerint vigyünk be kódokat!
Ha véletlenül hibás karaktert viszünk be, azt jelezze a program!
A feladatot próbáljuk úgy megoldani, hogy nem fordítunk a következő oldalra!

```python
ut = input("Kérem a robot parancsait: ")
hossz = len(ut)
e,d,k,n =0,0,0,0
for i in range(hossz):
    if ut[i] == 'E':
        e += 1
    elif ut[i] == 'D':
        d += 1
    elif ut[i] == 'K':
        k += 1
    elif ut[i] == 'N':
        n += 1
    else:
        print('Nem megfelelő kód!')
y=(e-d)
x=(k-n)
```

```python
if x < 0:
    for i in range(abs(x)):
        print('N',end='')
else:
    for i in range(x):
        print('K',end='')
if y < 0:
    for i in range(abs(y)):
        print('D',end='')
else:
    for i in range(y):
        print('E',end='')
```

(23d.py)

Az utolsó példa egy emelt szintű érettségi feladat megoldása. Először a leírást kell alaposan átolvasni, és megérteni!

A feladat címe: Építményadó

Egy Balaton-parti önkormányzat építményadót vezet be. Az adó mértéke a telken lévő építmény alapterületétől és a teleknek a Balatontól mért távolságától függ.

A telkeket a Balatonparttól mért távolságtól függően három sávba sorolták be. Az A sávba azok a telkek kerültek, amelyek 300 méternél közelebb vannak a tóhoz a B sáv az előzőn túl 600 méter távolságig terjed, a többi telek a C sávba tartozik. Az építmény után négyzetméterenként fizetendő összeg sávonként eltérő, azonban, ha az így kiszámított összeg nem éri el a 10.000 Ft-ot, akkor az adott építmény után nem kell adót fizetni.

```
Run:  23d
E:\00_MM\12_Python_programozas\programok\ven
2. feladat. A mintában 543 telek szerepel.
3. feladat. Egy tulajdonos adószáma: 79906
Aradi 3
A sávba 165 telek esik, az adó 20805600 Ft.
B sávba 144 telek esik, az adó 13107000 Ft.
C sávba 234 telek esik, az adó 3479600 Ft.
6. feladat. A több sávba sorolt utcák:
Besztercei
Gyurgyalag
Icce
Kurta
Rezeda
Szepesi
```

A testületi döntést az Adó Ügyosztály egy mintával készítette elő, amely csupán néhány utca adatait tartalmazza. Ezek az adatok az utca.txt fájlban vannak. A fájl első sorában a három adósávhoz tartozó négyzetméterenként fizetendő összeg található A, B, C sorrendben, egy-egy szóközzel elválasztva:

800 600 100

…

33366 Aradi 8A C 180

22510 Aradi 8B C 137

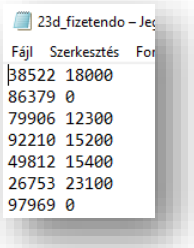
90561 Aradi 10 C 168

…

A többi sorban egy-egy építmény adatai szerepelnek egy-egy szóközzel elválasztva. Az első a telek tulajdonosának ötjegyű adószáma; egy tulajdonosnak több telke is lehet. A második adat az utca neve, amely nem tartalmazhat szóközt. A harmadik adat a házszám, majd az adósáv megnevezése, végül az építmény alapterülete következik. A minta harmadik sorában például azt látjuk, hogy a 33366 adószámú tulajdonos telke az Aradi utca 8A-ban található, és a C sávba eső telken álló építmény alapterülete 180 m2.

A fájl legfeljebb 1000 telek adatait tartalmazza. A feladat megoldása során kihasználhatja, hogy a fájlban az adatok utca, azon belül pedig házszám szerinti sorrendben következnek.
Készítsen programot, amely az utca.txt állomány adatait felhasználva az alábbi kérdésekre válaszol! A program forráskódját mentse epitmenyado.py néven! (A program megírásakor a felhasználó által megadott adatok helyességét, érvényességét nem kell ellenőriznie, és feltételezheti, hogy a rendelkezésre álló adatok a leírtaknak megfelelnek.)
A képernyőre írást igénylő részfeladatok esetén – a mintához tartalmában hasonlóan – írja ki a képernyőre a feladat sorszámát (például: 3. feladat), és utaljon a kiírt tartalomra is!
Ha a felhasználótól kér be adatot, jelenítse meg a képernyőn, hogy milyen értéket vár! Mindkét esetben az ékezetmentes kiírás is elfogadott.

1. Olvassa be és tárolja el az utca.txt állományban talált adatokat, és annak felhasználásával oldja meg a következő feladatokat!
2. Hány telek adatai találhatók az állományban? Az eredményt írassa ki a mintának megfelelően a képernyőre!
3. Kérje be egy tulajdonos adószámát, és írassa ki a mintához hasonlóan, hogy melyik utcában, milyen házszám alatt van építménye! Ha a megadott azonosító nem szerepel az adatállományban, akkor írassa ki a „Nem szerepel az adatállományban.” hibaüzenetet!
4. Készítsen függvényt ado néven, amely meghatározza egy adott építmény után fizetendő adót! A függvény paraméterlistájában szerepeljen az adósáv és az alapterület, visszaadott értéke pedig legyen a fizetendő adó! A következő feladatokban ezt a függvényt is felhasználhatja.
5. Határozza meg, hogy hány építmény esik az egyes adósávokba, és mennyi az adó összege adósávonként! Az eredményt a mintának megfelelően írassa ki a képernyőre!
6. Bár az utcák többé-kevésbé párhuzamosak a tó partjával, az egyes porták távolsága a parttól az utcában nem feltétlenül ugyanannyi. Emiatt néhány utcában – az ottani tulajdonosok felháborodására – egyes telkek eltérő sávba esnek. Listázza ki a képernyőre, hogy melyek azok az utcák, ahol a telkek sávokba sorolását emiatt felül kell vizsgálni! Feltételezheti, hogy minden utcában van legalább két telek.
7. Határozza meg a fizetendő adót tulajdonosonként! A tulajdonos adószámát és a fizetendő összeget írassa ki a mintának megfelelően a fizetendo.txt állományba! A fájlban minden tulajdonos adatai új sorban szerepeljenek, a tulajdonos adószámát egy szóközzel elválasztva kövesse az általa fizetendő adó teljes összege.



23d.py

```python
from pprint import pprint

epitmenyek = []
with open('./nyersanyag/23d_utca.txt', 'r', encoding='utf-8') as fajl:
    adok = fajl.readline().strip().split()
    adosavok = {'A': int(adok[0]), 'B': int(adok[1]), 'C': int(adok[2])}
    for sor in fajl:
        adatok = sor.strip().split()
        epitmeny = {'adoszam': adatok[0],
                    'utca': adatok[1],
                    'hazszam': adatok[2],
                    'adosav': adatok[3],
                    'terulet': int(adatok[4])}
        epitmenyek.append(epitmeny)
# print(adosavok)
# print(epitmenyek)

# 2. feladat
print(f'2. feladat. A mintában {len(epitmenyek)} telek szerepel.')

```

23d.py

```python
# 3. feladat
adoszam = input('3. feladat. Egy tulajdonos adószáma: ')
talalat = False
for epitmeny in epitmenyek:
    if adoszam == epitmeny['adoszam']:
        print(epitmeny['utca'], epitmeny['hazszam'])
        talalat = True
if not talalat:
    print('Nem szerepel az adatállományban')


# 4. feladat
def ado(ado_savonkent, adosav, alapterulet):
    ado = alapterulet * ado_savonkent[adosav]
    if ado < 10000:
        return 0
    else:
        return ado


# 5. feladat
ado_osszesites = {'A': [0, 0], 'B': [0, 0], 'C': [0, 0]}
for epitmeny in epitmenyek:
    ado_osszesites[epitmeny['adosav']][0] += 1
    ado_osszeg = ado(adosavok, epitmeny['adosav'], epitmeny['terulet'])
    ado_osszesites[epitmeny['adosav']][1] += ado_osszeg
# print(ado_osszesites)
print(f'A sávba {ado_osszesites["A"][0]} telek esik, az adó {ado_osszesites["A"][1]} Ft.')
print(f'B sávba {ado_osszesites["B"][0]} telek esik, az adó {ado_osszesites["B"][1]} Ft.')
print(f'C sávba {ado_osszesites["C"][0]} telek esik, az adó {ado_osszesites["C"][1]} Ft.')

# 6. feladat
utca_sav = {}
for epitmeny in epitmenyek:
    if epitmeny['utca'] in utca_sav:
        utca_sav[epitmeny['utca']].add(epitmeny['adosav'])
    else:
        utca_sav[epitmeny['utca']] = set(epitmeny['adosav'])
# print(utca_sav)
print('6. feladat. A több sávba sorolt utcák:')
for utca in utca_sav:
    if len(utca_sav[utca]) > 1:
        print(utca)

# 7. feladat
ado_tulajdonosonkent = {}
for epitmeny in epitmenyek:
    if epitmeny['adoszam'] in ado_tulajdonosonkent:
        ado_tulajdonosonkent[epitmeny['adoszam']] += ado(adosavok, epitmeny['adosav'], epitmeny['terulet'])
    else:
        ado_tulajdonosonkent[epitmeny['adoszam']] = ado(adosavok, epitmeny['adosav'], epitmeny['terulet'])
# pprint(ado_tulajdonosonkent)
with open('./nyersanyag/23d_fizetendo.txt', 'w', encoding='utf-8') as fizetendo:
    for azonosito in ado_tulajdonosonkent:
        print(azonosito, ado_tulajdonosonkent[azonosito], file=fizetendo)
```