Textový súbor

Premenné doteraz používaných údajových typov ukladajú svoje hodnoty len do operačnej pamäte a preto, po ukončení behu pythonovského skriptu, sa údaje nenávratne strácajú. Údaje uložené na vonkajšom pamäťovom médiu (najčastejšie na pevnom disku) sú však kedykoľvek k dispozícii, t.j. môžeme ich po spustení nášho programu načítať z disku a spracovať alebo, najneskôr pred ukončením behu programu, uložiť na disk pre neskoršie využitie. Python umožňuje čítať údaje zo súboru na disku aj ich na disk zapisovať. Výmena dát medzi súborom na disku a programom sa môže realizovať vo viacerých formátoch (napr. binárnom, textovom). Výhodou použitia textového formátu, t.j. textových súborov, je jednoduchá editovateľnosť textových súborov užívateľom. Na vytvorenie, zobrazenie obsahu alebo editovanie stačí jednoduchý textový editor, napríklad v systéme Microsoft Windows aplikácia Poznámkový blok. Pre textový súbor je typické členenie na riadky, t.j. súbor obsahuje znak `\n`. Či už chceme čítať z textového súboru alebo do neho zapisovať, musíme najprv vytvoriť prepojenie medzi konkrétnym súborom na disku a programom; hovoríme, že súbor musíme otvoriť na čítanie alebo zápis, prípadne na zápis na koniec súboru. Vo všetkých situáciách používame funkciu `open()`, avšak s rôznymi parametrami.

Čítanie zo súboru

začíname príkazom `premenná = open(názov_súboru, režim)`

kde

`premenná` sa najčastejšie označuje f (zo slova file),

`názov_súboru` je reťazec - názov súboru na disku, môže obsahovať aj cestu k súboru; po stustení programu je súbor hľadaný v priečinku, kde je aj Python skript, pokiaľ tam súbor umiestníme, cestu nemusíme zadať; ak súbor umiestníme do iného priečinka a neuvedieme k nemu cestu, nastane výnimka,

`režim` znamená použiť znak "r" (zo slova read), čím sme zvolili režim čítania zo súboru (je predvoleným režimom); pre zápis do súboru sa používa "w" (write) a pre zápis na koniec súboru "a" (append); pre čítanie zo súboru a zápis na koniec súboru "r+".

Funkcia open môže mať ďalšie parametre, napríklad v prípade problémov s kódovaním môžeme použiť parameter `encoding`.

Po otvorení súboru na čítanie v textovom režime môžeme pomocou funkcií:

`read()` načítať celý súbor do jedného reťazca,

`readline()` načítať ďalší riadok do reťazca,

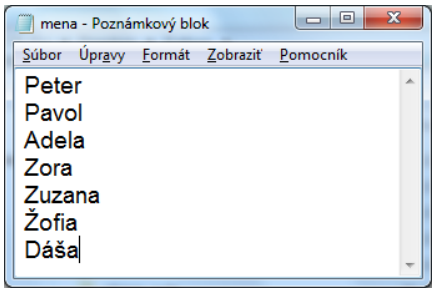
`readlines()` načítať všetky riadky do zoznamu reťazcov,

Všetko si to ozrejmíme na konkrétnych úlohách.

Po ukončení práce so súborom ho treba zavrieť, t.j. zrušiť prepojenie medzi programom a súborom na disku. Zavretie súboru sa realizuje funkciou `close()`.

Súbor 1

Čítanie zo súboru si začneme ozrejmovať na spracovaní súboru mena.txt, ktorý sme vytvorili v Poznámkovom bloku (obrázok vpravo; príponu pridá aplikácia sama!) a umiestnili do priečinka so zdrojovým skriptom.

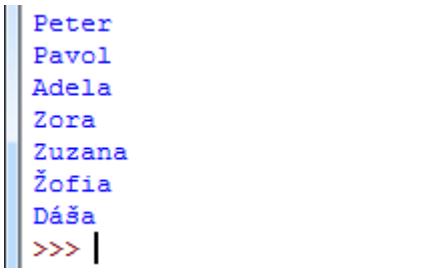


Úloha F.1

Vytvorte program, ktorý zobrazí obsah textového súboru mena.txt.

Riešenie 1.1

```
f = open("mena.txt", "r")
obsahSuboru = f.read()
f.close()
print(obsahSuboru)
```



Príkaz v prvom riadku otvorí súbor mena.txt na čítanie.

`obsahSuboru = f.read()` uloží celý obsah súboru do premennej obsahSuboru typu str vrátane znakov `\n`.

`f.close()` uzavrie súbor a

`print(obsahSuboru)` ho zobrazí, ako to je na obrázku vpravo.

V prípade, že ste súbor mena.txt zabudli vytvoriť alebo neumiestnili k zdrojovému skriptu, program pri pokuse otvoriť súbor mena.txt havaruje (nastane neošetrená výnimka) a zobrazí sa chybové hlásenie

... `FileNotFoundError`: Žiadny takýto súbor alebo adresár: "mena.txt". Ošetrenie výnimky konštrukciou try-except je pomerne jednoduché. Chceme, aby sa beh programu, ak nastane výnimka, ukončil. Použijeme príkaz `exit` z knižnice `sys` (preto `import sys`):

```
import sys
try:
    f = open("mena.txt", "r")
except FileNotFoundError:
    print("Súbor nenájdený, ukončil som program!")
    sys.exit()
obsahSuboru = f.read()
f.close()
print(obsahSuboru)
```

V ďalších riešeniach predpokladáme, že pri otváraní súboru nedôjde k výnimke a ošetrenie kvôli prehľadnosti riešenia vynecháme.

Riešenie 1.2

Použijeme funkciu `readlines`. Keďže funkcia `readlines` automaticky načíta všetky riadky otvoreného súboru do zoznamu reťazcov, na vytvorenie zoznamu stačí jediný príkaz `zoznamRiadkov = f.readlines()`.

```
f = open("mena.txt")          # otvorenie súboru na čítanie je predvoleným režimom - vynecháme "r"
zoznamRiadkov = f.readlines()
f.close()
print(zoznamRiadkov)
```

```
>>>
['Peter\n', 'Pavol\n', 'Adela\n', 'Žofia\n', 'Dáša\n', 'Zuzana\n', 'Zora']
>>>
```

Analýza tohto výstupu je veľmi dôležitá. Možno z neho vyčítať, že na konci každého riadka je zapísaný aj znak \n - prikazujúci prejsť na nový riadok. Tento znak nie je za posledným menom v súbore, pretože pri písaní súboru mena.txt sme za posledným riadkom nestlačili kláves Enter. Dosiahnúť „krajší“ výpis prvkov zoznamu možno použitím príkazu `for riadok in zoznamRiadkov: print(riadok)` - pozri Výpis 1. Zrejme nám budú vadiť prázdne riadky, ktorých sa môžeme zbaviť viacerými spôsobmi. Buď ovplyvníme len výpis použitím parametra `end` v príkaze `print`: `for riadok in zoznamRiadkov: print(riadok, end="")` alebo pred vypísaním z konca každého riadka dáme odstrániť znak \n (pozri funkciu strip): `for riadok in zoznamRiadkov: print(riadok.rstrip())`. Oba prípady vedú k Výpisu 2.

Výpis 1

```
>>>
Peter

Pavol

Adela

Žofia

Dáša

Zuzana

Zora
>>>
```

Výpis 2

```
>>>
Peter
Pavol
Adela
Žofia
Dáša
Zuzana
Zora
>>>
```

Výpis 3

```
>>>
Peter
Pavol
Adela
Žofia
Dáša
Zuzana
Zor
>>>
```

Výpis 4

```
>>>
Adela
Dáša
Pavol
Peter
Zora
Zuzana
Žofia
>>>
```

Požiadavka zbaviť sa znaku \n môže viesť aj k myšlienke odrezať znak \n na konci každého riadka, t.j. použiť napríklad príkaz `for riadok in zoznamRiadkov: print(riadok[:-1])`, čo ale spôsobí aj nežiaduce odrezanie posledného znaku v poslednom mene - pozri Výpis 3. **Vhodnejšie aj univerzálnejšie je použiť funkciu rstrip, ktorá vyrieši problém aj s prípadnými bielymi znakmi vloženými za posledný platný údaj, najčastejšie stlačenie klávesu Enter - vložený prázdny riadok na konci súboru.**

Použitie funkcie `readlines()` má aj tú výhodu, že môžeme ľahko docieliť utriedenie reťazcov v zozname použitím funkcie `sort()` (príkaz `zoznamRiadkov.sort()`), ako to vidieť vo Výpise 4.

Riešenie 1.3a
Ďalšou alternatívou je použiť funkciu readline, teda čítať súbor po riadkoch. Riešenie 3a využíva príkaz for:

```
f = open("mena.txt", "r")
for riadok in f:
    print(riadok.rstrip())
f.close()
```

Toto riešenie nám dá Výpis 2.

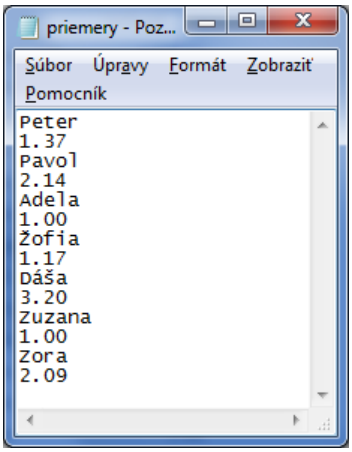
Riešenie 1.3b
Riešenie 3b vychádza z úvahy, že koniec súboru možno interpretovať ako neexistenciu ďalšieho riadka, t.j. „načíta sa“ prázdny riadok resp. vráti hodnota None (vyhodnotená ako False). Riešenie 3b využíva príkaz while:

```
f = open("mena.txt", "r")
riadok = f.readline()
while riadok != "":                 # stačí aj while riadok:
    print(riadok.rstrip())
    riadok = f.readline()
f.close()
```

Všetky riešenia otestujte aj na prázdny súbor, súbor obsahujúci len jeden riadok, súbor obsahujúci aj prázdny riadok a súbor, v ktorom je po poslednom neprázdnom riadku ešte stlačený kláves Enter prípadne vložených viac bielych znakov.

Postúpme vo výklade na vyššiu úroveň.

Súbor 2
Textový súbor mena.txt doplňte o riadky so študijným priemerom pod každým menom a uložte pod názvom priemery.txt (obrázok vpravo).



Pokiaľ môžeme rozhodnúť o tom, ako budú údaje v textovom súbore organizované, je veľmi praktické, keď údaj, ktorý reprezentuje nejaký text (meno, adresu,...) je na samostatnom riadku. Číselných údajov môže byť aj viacej v jednom riadku, dajú sa ľahko vyselektovať (väčšinou bývajú jednotlivé čísla oddelené medzerou) najčastejšie funkciou split. Jednotlivým prípadom sa teraz budeme venovať.

Úloha F.2
Vytvorte program, ktorý zobrazí obsah textového súboru priemery.txt.

Riešenie 2.1
Riešenie je opakovaním riešenia 1.1 z predchádzajúcej úlohy, takže bez komentára:

```
import sys
try:
    f = open("priemery.txt", "r")
except FileNotFoundError:
    print("Súbor nenájdený, ukončil som program!")
    sys.exit()
obsahSuboru = f.read()
f.close()
print(obsahSuboru)
```

Úlohu si sťažíme a budeme požadovať, aby vo výpise bolo v jednom riadku meno aj k nemu prislúchajúci priemer (obrázok vpravo).

```
>>>
Peter       1.37
Pavol       2.14
Adela       1.00
Žofia       1.17
Dáša        3.20
Zuzana      1.00
Zora        2.09
>>>
```

Riešenie 2.2a
Zrejme musíme čítať súbor po riadkoch. Treba si tiež uvedomiť usporiadanie dát v súbore a pri čítaní dát zo súboru to využiť. V našom prípade sa striedajú riadky s menom a priemerom. Je viacej možností, ako zobraziť načítané dáta. Nám sa zdá najpraktickejšie použiť while-cyklus a hneď po načítaní reťazca priemer využiť konverziu na float (zbavíme sa tým problému so znakom \n).

```
import sys
try:
    f = open("priemery.txt", "r")
except FileNotFoundError:
    print("Súbor nenájdený, ukončil som program!")
    sys.exit()

meno = f.readline()
while meno != "":                          # stačí aj while meno:
    meno = meno.strip()
    priemer = float(f.readline())
    print("{meno:10} {priemer:.2f}".format(**locals()))
    meno = f.readline()
f.close()
```

Riešenie 2.2b

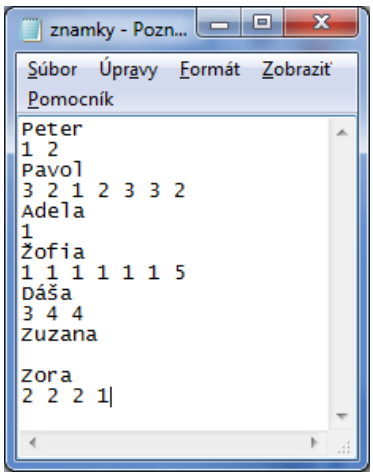
„Násilné“ použitie for-cyklus. Keďže každým prechodom for-cyklom sa načíta nový riadok, či ide o nepárny obsahujúci meno alebo párny obsahujúci priemer, musíme „ustrážiť“ my. Použijeme premenú `neparny_riadok`, ktorá bude striedať hodnoty True (pravda) a False (nepravda). Počiatočná hodnota je True (riadok č.1 je nepárny) a po načítaní riadka sa má hodnota zmeniť na opačnú (z True na False alebo z False na True), čo zabezpečí negácia aktuálnej hodnoty, teda príkaz `neparny_riadok = not(neparny_riadok)`. Použiť by sa mohla aj číselná premenná.

```
try:
    f = open("priemery.txt", "r")
except FileNotFoundError:
    print("Súbor nenájdený, ukončil som program!")
    sys.exit()

neparny_riadok = True
for riadok in f:
    if neparny_riadok:
        meno = riadok.strip()
    else:
        priemer = float(riadok)
        print("{meno:10} {priemer:.2f}".format(**locals()))
    neparny_riadok = not(neparny_riadok)
f.close()
```

Súbor 3

Textový súbor mena.txt doplňte o riadky so známkami jednotlivých študentov a uložte pod názvom znamky.txt (v súbore sa striedajú riadky s menami a známkami, ako to vidieť na obrázku vpravo).



Úloha F.3.1

Vytvorte program, ktorý zobrazí obsah textového súboru znamky.txt.

Riešenie 3.1a

je analogické s riešeniami 2.1 a 1.1, iný je len názov súboru, ktorý treba otvoriť `f = open("znamky.txt", "r")`.

Úlohu si opäť sťažíme a budeme požadovať, aby vo výpise bolo v jednom riadku meno aj k nemu prislúchajúce známky (obrázok vpravo nižšie).

Riešenie 3.1b

Použijeme while-cyklus. Keďže známky netreba spracovať, stačí s celým riadkom so známkami pracovať ako s reťazcom.
Výstup riešenia je na obrázku vpravo.

```
import sys

try:
    f = open("znamky.txt", "r")
```

```
Peter      1 2
Pavol      3 2 1 2 3 3 2
Adela      1
Žofia      1 1 1 1 1 1 5
Dáša       3 4 4
Zuzana
Zora       2 2 2 1
```

```
except FileNotFoundError:
    print("Súbor nenájdený, ukončil som program!")
    sys.exit()

meno = f.readline()
while meno != "":
    meno = meno.strip()
    znamky = f.readline().strip()
    print("{meno:10} {znamky}".format(**locals()))
    meno = f.readline()
f.close()
```

Úloha F.3.2

Vytvorte program, ktorý zo známok zadaných v súbore znamky.txt vypočíta a zobrazí priemery prislúchajúce k jednotlivým menám.

V súbore znamky.txt sme spravili jednu zmenu, Zuzane sme nedali ani jednu známku (riadok so známkami má prázdny). Program má pri chýbajúcich známkach vypísať priemer 0.00. Pozri obrázok nižšie.

Riešenie 3.2

Výpočet priemeru sme umiestnili do samostatnej funkcie `vratPriemerZnamok` s parametrom `znamky` – zoznam známok typu str získaný funkciou split.

```
>>>
Peter      1.50
Pavol      2.29
Adela      1.00
Žofia      1.57
Dáša       3.67
Zuzana     0.00
Zora       1.75
>>>
```

```
import sys

def vratPriemerZnamok(znamky):
    if len(znamky) == 0:
        return 0.0
    else:
        sucet = 0
        for znamka in znamky:
            sucet += int(znamka)
        return sucet/len(znamky)

# ========================= HLAVNÝ PROGRAM ====================
try:
    f = open("znamky.txt", "r")
except FileNotFoundError:
    print("Súbor nenájdený, ukončil som program!")
    sys.exit()

meno = f.readline().strip()
while meno != "":
    znamky = f.readline().split()
    priemer = vratPriemerZnamok(znamky)
    print("{meno:10} {priemer:.2f}".format(**locals()))
    meno = f.readline().strip()
f.close()
```

Program otestujte aj na prázdny súbor.

Úloha F.4 (rovnaký problém ako F.3.1 a F.3.2)

Vytvorte program, ktorý spracuje dáta uložené v textovom súbore v tvare meno žiaka a na novom riadku jeho známky oddelené medzerami, napríklad:

```
Fero
1 1 4
Katka
1 1
Karol

Dušan
5 4 4 2
```

Program nech umožňuje zadať meno textového súboru, ktorého dáta sa majú spracovať; vo funkciách vypísať súbor; vypísať po riadkoch formátovane meno a známky žiaka a vypísať po riadkoch formátovane meno a priemer jeho známok. Ak študent nemá ani jednu známku, jeho priemer je 0,00.

Ukážka výstupu - vypísať súbor:

```
Spracuj triedu (data) zo súboru: tretiaci
Fero
1 1 4
Katka
1 1
Karol

Dušan
5 4 4 2
```

Ukážka výstupu - vypísať po riadkoch formátovane meno a známky žiaka:

```
Meno       Známky
Fero       1 1 4
Katka      1 1
Karol
Dušan      5 4 4 2
```

Ukážka výstupu - vypísať po riadkoch formátovane meno žiaka a priemer jeho známok:

```
Meno       Priemer
Fero       2.00
Katka      1.00
Karol      0.00
Dušan      3.75
```

Jednotlivé funkcie:

```
def vypisSubor():
    subor = open(nazov_triedy,"r")
    print(subor.read())
    subor.close()

def vypisMenoZnamky():
    subor = open(nazov_triedy,"r")
    print("\n{:10} {}".format("Meno", "Známky"))
    meno = subor.readline().strip()
    while meno:
        znamkyStr = subor.readline().rstrip()
        print("{:10} {}".format(meno, znamkyStr))
        meno = subor.readline().strip()
    subor.close()

def vypisMenoPriemer():
    subor = open(nazov_triedy,"r")
    print("\n{:10} {}".format("Meno", "Priemer"))
    meno = subor.readline().strip()
    while meno:
        znamky = subor.readline().split()
        sucet = 0
        for i in range(len(znamky)):
            sucet += int(znamky[i])
        if len(znamky) > 0:
            priemer = sucet/len(znamky)
        else:
            priemer = 0.0
        print("{:10} {:4.2f}".format(meno, priemer))
        meno = subor.readline().strip()
    subor.close()

def main():
    vypisSubor()
    vypisMenoZnamky()
    vypisMenoPriemer()

# =====================================================
nazov_triedy = input("Spracuj triedu (data) zo súboru: ")+ ".txt"
```

```
try:                                                # ošetrenie zle zadaného názvu súboru
    f = open(nazov_triedy)
except FileNotFoundError:
    print("Súbor so zadaným názvom v aktuálnom priečinku neexistuje, ukončil som beh programu!")
    sys.exit()                                      # nezabudnite na import sys
main()
```

Úloha F.5

Vytvorte program, ktorý spracuje dáta uložené v textovom súbore v tvare meno pracovníka a na novom riadku počet jeho odpracovaných hodín, napríklad:

```
Novák
187.5
Kováčová
168
Malý
200
Úzky
180
```

Program nech umožňuje zadať meno textového súboru, ktorého dáta sa majú spracovať; vypísať súbor; vypísať po riadkoch formátovane meno a odpracované hodiny zamestnanca a vypísať po riadkoch formátovane meno a plat zamestnanca (plat zamestnanca sa vypočíta ako súčin počtu odpracovaných hodín a zadanej konštanty hodinová mzda). Koniec posledného výstupu nech je doplnený o celkovú vyplatenú sumu, t.j. súčet platov jednotlivých zamestnancov.

Ukážka výstupu:

```
Spracovať súbor: platy
Novák
187.5
Kováčová
168
Malý
200
Úzky
180

Meno       Odpr.hod.
Novák          187.5
Kováčová       168.0
Malý           200.0
Úzky           180.0

Meno       Plat v €
Novák         937.50
Kováčová      840.00
Malý         1000.00
Úzky          900.00
==================
Súčet        3677.50
```

Riešenie:

```
def vypisSubor():
    subor = open(nazov_suboru, "r")
    print(subor.read())
    subor.close()

def vypisMenoHodiny():
    print("\n{:10} {}".format("Meno", "Odpr.hod."))
    subor = open(nazov_suboru)
    meno = subor.readline().rstrip()
    while meno:
        hodinyStr = subor.readline().rstrip()
        print("{:10} {:8.1f}".format(meno, float(hodinyStr)))
        meno = subor.readline().rstrip()
    subor.close()
```

```
def vypisMenoPlat():
    print("\n{:10} {:>8}".format("Meno", "Plat v €"))
    subor = open(nazov_suboru)
    sucet = 0
    meno = subor.readline().rstrip()

    while meno:
        hodiny = float(subor.readline().rstrip())
        sucet += hodiny*HM
        print("{:10} {:8.2f}".format(meno, hodiny*HM))           #
        meno= subor.readline().rstrip()
    subor.close()
    print("="*19)
    print("{:10} {:8.2f}".format("Súčet", sucet))


def main():
    vypisSubor()
    vypisMenoPlat()

# =======================================
HM = 5 # hodinová mzda v eurách
nazov_suboru = input("Spracovať súbor: ") + ".txt"
main()
```

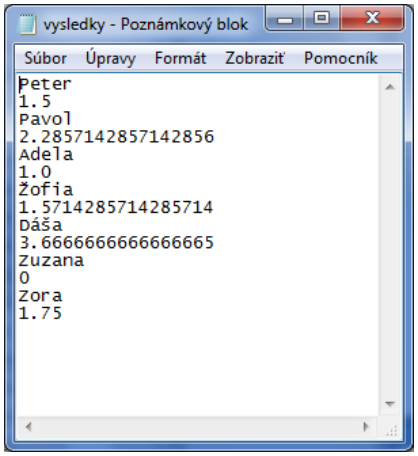
Čo sa zmení, ak riadok označený # nahradíme riadkom `print("{:10} {:7.1f}0".format(meno, hodiny*HM)`?

Zápis do súboru

Prvým krokom je otvorenie súboru na zápis funkciou open s názvom súboru a s parametrom "w" alebo "a". Parameter "w" zabezpečí, že sa vytvorí v aktuálnom priečinku nový prázdny súbor. Ak už súbor so zadaným menom na disku existuje, vyprázdni sa. Parameter "a" (append - pridať) zabezpečí nastavenie zápisu na koniec súboru. Vlastný zápis do súboru sa realizuje príkazom `write(s)`, kde s je reťazec.
Po ukončení práce so súborom ho treba zavrieť, t.j. zrušiť prepojenie medzi programom a súborom na disku; zároveň sa na vonkajšiu pamäť zapíšu ešte nezapísané dáta z operačnej pamäte. Zavretie súboru sa realizuje funkciou `close()`.

Úloha F.4: Vytvorte program, ktorý bude čítať údaje zo súboru znamky.txt, vypočíta priemer pre každé meno a meno a priemer zapíše do súboru vysledky.txt.

Riešenie:
Súborovú premennú sme nazvali g a súbor sa najprv musí otvoriť na zápis, teda použiť príkaz `g = open("vysledky.txt", "w")`. Pri otváraní súboru na zápis môže nastať chyba len ak zadáme zlú cestu. Každé meno aj priemer majú byť v samostatných riadkoch, čo znamená po každom zápise zapísať aj prechod na nový riadok (okrem posledného priemeru!). Pri zápise mena využijeme fakt, že pri čítaní zo súboru znamky.txt je načítaný aj prechod na nový riadok. Preto stačí zapísať neupravené meno príkazom `g.write(meno)`. Po výpočte aritmetického priemeru funkciou vratPriemerZnamok (v programe jej definíciu neuvádzame, je analogická s riešením 3.2) treba hodnotu priemeru zapísať do súboru, čo možno po konverzii priemeru (reálne číslo) na reťazec. Po zapísaní priemeru do súboru má nový zápis pokračovať na novom riadku, čo ale neplatí po zápise posledného priemeru. Tento problém sme vyriešili príkazmi `meno = f.readline()` a `if meno != "": g.write("\n")` – ak existuje ďalšie meno v súbore f, zapíš do súboru g prechod na nový riadok (za ostatný priemer). Keďže program z pohľadu užívateľa zdanlivo nič nerobí, po zavretí oboch súborov sme dali vypísať „OK“ a môžeme editovať nový súbor vysledky.txt (obrázok vpravo).

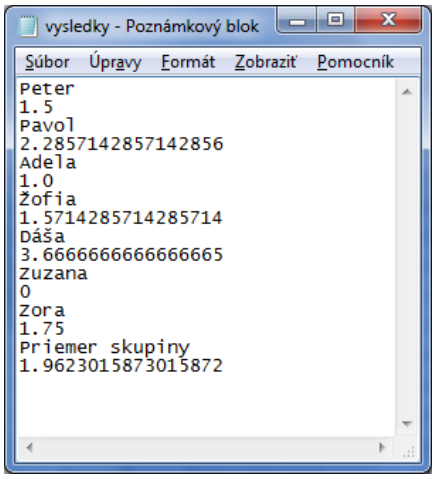



```
# ======================== HLAVNÝ PROGRAM ====================
import sys
try:
    f = open("znamky.txt", "r")
    g = open("vysledky.txt", "w")
except FileNotFoundError:
    print("Chyba pri otváraní súborov, ukončil som program!")
    sys.exit()
```

```
meno = f.readline()
while meno:
    g.write(meno)
    znamky = f.readline().split()
    priemer = vratPriemerZnamok(znamky)          # funkcia vratPriemerZnamok je v úlohe F.3.2
    g.write(str(priemer))
    meno = f.readline()
    if meno: g.write("\n")
f.close()
g.close()
print("OK")
```

Úloha F.5: Vytvorte program, ktorý bude čítať priemery zo súboru vysledky.txt, vypočíta priemer priemerov a na koniec súboru zapíše dva riadky: text „Priemer skupiny“ a hodnotu priemeru priemerov (pozri obrázok nižšie).

Riešenie:
Úlohu rozdelíme na dve časti. V prvej časti otvoríme súbor na čítanie (režim "r") a vypočítame priemer priemerov. V druhej časti otvoríme súbor na zápis na koniec súboru (režim "a") a vykonáme zápis (ak je čo zapísať).




```
import sys
nazov_suboru = "vysledky.txt"
try:
    f = open(nazov_suboru, "r")
except FileNotFoundError:
    print("Chyba pri otváraní súboru, ukončil som program!")
    sys.exit()

sucet,pocet = 0,0
meno = f.readline()
while meno:
    priemer = float(f.readline())
    if priemer > 0:
        sucet += priemer
        pocet += 1
    meno = f.readline()
f.close()
f = open(nazov_suboru, "a")
try:
    priemer = sucet/pocet
    f.write("\nPriemer skupiny\n")
    f.write(str(priemer))
except ZeroDivisionError:
    pass
f.close()
print("OK")
```

alebo použijeme vo funkcii `open` parameter `r+`, ktorý nastavuje čítanie aj zápis do súboru

```
f = open(nazov_suboru, "r+")
```

a preto môžeme v programe vynechať príkazy pred blokom na zápis, t.j. vynechať

```
f.close()
f = open(nazov_suboru, "a")
```

Najpoužívanejšie funkcie pre prácu s textovým súborom

f.close()	uzavre súbor f
f.closed()	vráti True, ak je súbor uzavretý
f.flush()	vynúti zápis zo súboru na disk
f.name	vráti názov súboru (ak existuje)
f.read()	vráti reťazec obsahujúci znaky od aktuálnej pozície po koniec súboru; **ak už nie je čo čítať, vráti prázdny reťazec**
f.readable()	vráti True, ak bol súbor otvorený na čítanie
f.readline()	prečíta ďalší riadok vrátane \n
f.readlines()	prečíta všetky riadky až do konca súboru a vráti ich ako zoznam
f.writable()	vráti True, ak bol súbor otvorený na zápis
f.write(s)	zapíše do súboru reťazec s
f.writelines(p)	zapíše do súboru zadanú postupnosť p reťazcov

Poznámka

Príkazy `f = open(nazov_suboru, parameter)` a `f.close()` možno nahradiť jediným príkazom `with open(nazov_suboru, parameter) as f:` príkazy, takže ostatná úloha F.5 by mohla mať aj riešenie:

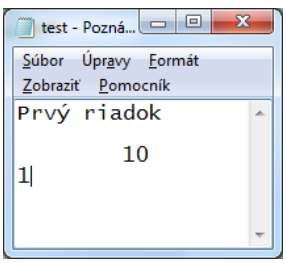
```
import sys
nazov_suboru = "vysledky.txt"
try:
    with open(nazov_suboru, "r+") as f:
        sucet,pocet = 0,0
        meno = f.readline()
        while meno:
            priemer = float(f.readline())
            if priemer > 0:
                sucet += priemer
                pocet += 1
            meno = f.readline()
        try:
            priemer = sucet/pocet
            f.write("\nPriemer skupiny\n")
            f.write(str(priemer))
        except ZeroDivisionError:
            pass
except FileNotFoundError:
    print("Chyba pri otváraní súboru, ukončil som program!")
    sys.exit()
print("OK")
```

Úlohy na precvičenie

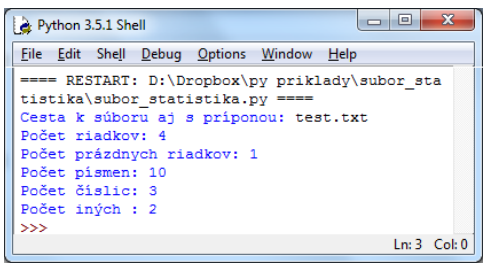
Pri riešení úloh so súbormi by ste mali uvažovať aj o takej alternatíve, že dát v spracovávanom textovom súbore je tak veľa, že sa naraz nezmestia do operačnej pamäte počítača (môže nastať pri príkazoch read() a readlines()). Preto ideálnym je riešenie, keď čítate a spracúvate údaje postupne po riadkoch.

Úloha F.6: Vytvorte program, ktorý vypíše štatistiku o textovom súbore. Štatistika nech obsahuje počet riadkov, počet prázdnych riadkov, počet písmen, číslic a iných znakov (okrem znaku prechod na nový riadok – \n).

Napríklad pre textový súbor test.txt



máme dostať výstup



(pred číslom 10 je stlačený tabulátor, ktorý je započítaný medzi iné znaky spolu s medzerou).

Riešenie:
Jednou z možností je načítať obsah celého súboru do reťazca príkazom read() a analyzovať jednotlivé, v ňom sa vyskytujúce, znaky. Napríklad prázdny riadok sa prejaví ako dva bezprostredne za sebou idúce znaky \n\n. Ošetriť môžete aj prázdny súbor.

```
import sys

nazov_suboru = input("Cesta k súboru aj s príponou: ")
try:
    f = open(nazov_suboru)
except:
    print("Súbor nenájdený, končím!")
    sys.exit()
obsah = f.read()
f.close()

if obsah == "":
    print("Súbor",nazov_suboru,"je prázdny!")
    sys.exit()

pocet_riadkov = 1
for znak in obsah:
    if znak == '\n':
        pocet_riadkov += 1
print("Počet riadkov:",pocet_riadkov)

pocet_prazdnych_riadkov = 0
for i in range(len(obsah)-1):
    if obsah[i] == '\n' and obsah[i+1] == '\n':
        pocet_prazdnych_riadkov += 1
print("Počet prázdnych riadkov:",pocet_prazdnych_riadkov)

pocet_pismen = 0
pocet_cislic = 0
for znak in obsah:
    if znak.isalpha():
        pocet_pismen += 1
    elif znak.isdigit():
        pocet_cislic += 1
print("Počet písmen:", pocet_pismen)
print("Počet číslic:", pocet_cislic)
print("Počet iných :", len(obsah)-pocet_pismen-pocet_cislic-pocet_riadkov+1)
```

Úloha F.7.1: V textovom súbore bmi.txt sú riadky obsahujúce hmotnosti respondentov v kg (celé čísla) a ich výšky v metroch (reálne čísla). Vytvorte program, ktorý do súboru bmi_doplnene.txt prepíše hmotnosť a výšku respondenta a riadok doplní vypočítanou hodnotou BMI. BMI sa vypočíta ako podiel hmotnosti v kg a druhej mocniny výšky v m.
Pripomíname, že podobný príklad je riešený v študijnom texte Štruktúrované údajové typy Príklad S.6.

Riešenie:

```
import sys

def doplnBMI():
    f = open(nazov_suboru+".txt","r")
    g = open(nazov_suboru+"_doplneny.txt","w")
    dataStr = f.readline().strip()
    while dataStr:
        hmotnostStr, vyskaStr = dataStr.split()
        hmotnost, vyska = int(hmotnostStr), float(vyskaStr)
        bmi = hmotnost / vyska / vyska
        g.write(dataStr + " " + str(bmi))
        dataStr = f.readline().strip()
        if dataStr: g.write('\n')
    f.close()
    g.close()
```

```
#================================================
nazov_suboru = input("Názov súboru (bez prípony): ")
try:
    f = open(nazov_suboru+'.txt')
    f.close()
except FileNotFoundError:
    print("Súbor nenájdený, končím!")
    sys.exit()

doplnBMI()
```

Úloha F.7.2: Program z úlohy F.7.1 doplňte tak, aby čítaním súboru bmi_doplnene.txt zistil a vypísal percento obéznych respondentov. Respondent je obézny, ak jeho BMI je väčšie ako 30.

```
def vypisStatistiku():
    f = open(nazov_suboru+"_doplneny.txt")
    pocet_obeznych = 0
    pocet = 0
    dataStr = f.readline().strip()
    while dataStr:
        bmi = float(dataStr.split()[2])          # všimnite si prístup k hodnote bmi!
        if bmi > 30:
            pocet_obeznych += 1
        pocet += 1
        dataStr = f.readline().strip()
    f.close()

    try:
        percent = pocet_obeznych / pocet * 100
        print("Obéznych: {:.1f}%".format(percent))
    except ZeroDivisionError:
        pass
```

Úlohu F.7.1 si môžete sťažiť tým, že predpokladajte, že textový súbor obsahuje aj mená respondentov a výšky sú uvedené v cm, napríklad prvý riadok: Dalibor, druhý riadok: 69 182 atď.

Riešenie študentky Natálie Holkovej:

```
import sys

def doplnBMI(zoznam):
    i = 1                                        # na párnom indexe (0,2,4,...) riadka je meno!
    while i < len(zoznam):
        hmotnostStr, vyskaStr = zoznam[i].split()
        hmotnost, vyska = int(hmotnostStr), (int(vyskaStr)/100)
        bmi = hmotnost / vyska / vyska
        zoznam[i] = zoznam[i].rstrip() + " " + str(round(bmi, 1)) + "\n"
        i += 2
    zoznam.append(zoznam.pop()[:-1])             #  odstránenie znaku \n na konci súboru

def zapisDoSuboru(zoznam):
    f = open(nazov_suboru+"_doplneny.txt","w")
    for zaznam in zoznam:
        f.write(zaznam)
    f.close()
    print("BMI ZAPÍSANÉ!")
#================================================
nazov_suboru = input("Názov súboru (bez prípony): ")
try:
    f = open(nazov_suboru+".txt", "r")
except FileNotFoundError:
    print("Súbor nenájdený, končím!")
    sys.exit()
obsah = f.readlines()
f.close()
```

```
doplnBMI(obsah)
zapisDoSuboru(obsah)
```

Úloha F.8: Z útulku Sloboda zvierat vypustili po vyliečení vlka. V snahe zistiť jeho zvyky, namontovali mu vysielačku, ktorá vždy po 24 hodinách vyslala aktuálnu polohu vlka (súradnice x, y) vzhľadom k miestu vypustenia ([0,0]). Údaje boli zaznamenávané do textového súboru vlk.txt ako celočíselné hodnoty, každý riadok obsahuje súradnicu x a y. Prvý riadok v súbore je záznam polohy z dňa vypustenia, t.j. 0 0. Použitá bola km sieť.
Aká je celková dĺžka trasy, ktorú prešiel vlk od vypustenia, ak sa medzi meraniami pohyboval viac-menej priamočiaro? Prejdené vzdialenosti v jednotlivých dňoch nech sa zapíšu do súboru trasy.txt.
V koľký deň prešiel vlk najdlhšiu trasu za 24 hodín (využite údaje zo súboru trasy.txt)?

Príklad súboru vlk.txt: Pri výpočte vystačíme len s x_stare a x_nove resp. y_stare a y-nove!

0 0	bod vypustenie	0 0	
4 3	za 1.deň prešiel 5 km ($\sqrt{[(4-0)^2 + (3-0)^2]} = \sqrt{[(x\_nove-x\_stare)^2 + (y\_nove-y\_stare)^2]}$)		4 3
3 4	za 2 .deň prešiel $\sqrt{2} \approx 1{,}4$ km ($\sqrt{[(3-4)^2 + (4-3)^2]}$)	4 3	3 4
3 7	za 3. deň 3 km ($\sqrt{[(3-3)^2 + (7-4)^2]}$)	3 4	3 7
7 9	za 4. deň ≈ 4,5 km	3 7	7 9
1 0	za 5. deň ≈ 10,8 km	7 9	1 0

Riešenie študentky Natálie Holkovej:

```
import sys, math

def vratPrejdenuVzdialenost(nazov_suboru):
    try:
        f = open(nazov_suboru, "r")
    except FileNotFoundError:
        print("Súbor nenájdený, končím!")
        sys.exit()
    g = open("trasy.txt", "w")

    sucet_vzd = 0
    den = 0
    x_stareStr, y_stareStr = f.readline().split()
    x_stare, y_stare = float(x_stareStr), float(y_stareStr)
    for riadok in f.readlines():
        x_noveStr, y_noveStr = riadok.split()
        x_nove, y_nove = float(x_noveStr), float(y_noveStr)
        vzd = math.sqrt(math.pow(x_nove-x_stare,2) + math.pow(y_nove-y_stare,2))
        sucet_vzd += vzd
        g.write(str(vzd) + '\n')
        x_stare = x_nove
        y_stare = y_nove
        # kontrolný výpis:
        den += 1
        print("Za {}. deň prešiel {} km.".format(den, round(vzd,1)))
    f.close()
    g.close()
    return sucet_vzd

def vratMaxDen():
    f = open("trasy.txt", "r")
    trasy = []
    for vzdStr in f.readlines():          # predpoklad: všetky namerané údaje sa zmestili do RAM
        trasy.append(float(vzdStr.strip()))
    f.close()
    return trasy.index(max(trasy)) + 1
```

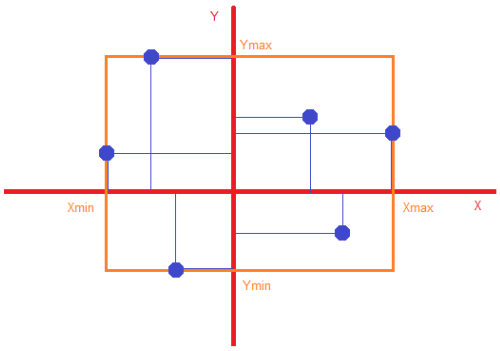
```
#====================================
nazov_suboru = "vlk.txt"
print("Celkovo prešiel {:.1f} km.".format(vratPrejdenuVzdialenost(nazov_suboru)))
print("Najdlhšiu trasu prešiel {}. deň.".format(vratMaxDen()))
```

Pre vyššie uvedené dáta súboru vlk.txt dostávame výstup:
Za 1. deň prešiel 5.0 km.
Za 2. deň prešiel 1.4 km.
Za 3. deň prešiel 3.0 km.
Za 4. deň prešiel 4.5 km.
Za 5. deň prešiel 10.8 km.
Celkovo prešiel 24.7 km.
Najdlhšiu trasu prešiel 5. deň.

Úloha F.9: Pozemok s bodovými objektmi chceme ohradiť ohradou v tvare obdĺžnika so stranami rovnobežnými s osami x a y. Súradnice objektov sú dvojice celých čísel uložené v súbore objekty.txt (môžete premenovať súbor vlk.txt). Vytvorte program, ktorý vypíše minimálnu dĺžku pletiva potrebného na oplotenie objektov.
Pripomíname, že podobný príklad je riešený v študijnom texte Štruktúrované údajové typy Príklad S.4.

Riešenie:

Z obrázka vpravo je zrejmé, že musíme nájsť najmenšiu x-ovú a y-ovú súradnicu a najväčšiu x-ovú a y-ovú súradnicu zo všetkých zadaných súradníc. Rozdiel $x_{max} - x_{min}$ je veľkosť jednej strany hľadaného obdĺžnika a rozdiel $y_{max} - y_{min}$ je veľkosť druhej strany hľadaného obdĺžnika. Výpočet minimálnej dĺžky pletiva je výpočet obvodu hľadaného obdĺžnika ($o = 2(x_{max} - x_{min} + y_{max} - y_{min})$).
Program sme doplnili aj o vypísanie „min a max“ súradníc a výpočet obsahu hľadaného obdĺžnika.




```
import sys
def vratMinMaxSuradnice(nazov_suboru):
    try:
        f = open(nazov_suboru,"r")
        minXstr, minYstr = f.readline().split()
        minX = float(minXstr)
        minY = float(minYstr)
        maxX, maxY = minX, minY
        xyStr = f.readline().split()
        while xyStr:
            x = float(xyStr[0])
            y = float(xyStr[1])
            if x < minX: minX = x
            if x > maxX: maxX = x
            if y < minY: minY = y
            if y > maxY: maxY = y
            xyStr = f.readline().split()
        f.close()
        return minX, minY, maxX, maxY
    except FileNotFoundError:
        print("Chyba pri čítaní dát!")
        sys.exit()

def vratMinObvod(minMaxSur):
    return 2*((minMaxSur[2] - minMaxSur[0])+(minMaxSur[3] - minMaxSur[1]))
```

```
def vratMinObsah(minMaxSur):
    return (minMaxSur[2] - minMaxSur[0])*(minMaxSur[3] - minMaxSur[1])

def main():
    nazov_suboru = input("Názov súboru (bez prípony): ") + ".txt"
    print("Štvorica (minX, minY, maxX, maxY):", vratMinMaxSuradnice(nazov_suboru))
    print("Obvod minimálneho obdĺžnika je {:.1f} j.".format(vratMinObvod(vratMinMaxSuradnice(nazov_suboru))))
    print("Obsah minimálneho obdĺžnika je {:.1f} j\u00b2.".format(vratMinObsah(vratMinMaxSuradnice(nazov_suboru))))

main()
```

Pre údaje zo súboru vlk.txt (úloha F.8) premenovaného na objekty.txt dostávame:

```
Názov súboru (bez prípony): objekty
Štvorica (minX, minY, maxX, maxY): (0.0, 0.0, 7.0, 9.0)
Obvod minimálneho obdĺžnika je 32.0 j.
Obsah minimálneho obdĺžnika je 63.0 j².
```

Úloha F.9: Vytvorte program simulujúci elektronický telefónny zoznam (záznam obsahuje: meno, adresa, tel.číslo) s ponukou: Vypísať zoznam, Pridať záznam, Vyhľadať záznam podľa mena (podľa prvých písmen mena), Odstrániť záznam a Opraviť záznam. Myslite aj na to, že pri prvom spustení programu môže súbor s údajmi celkom chýbať.

Riešenie študentky Natálie Holkovej:

```
import sys

def ponuka():
    while True:
        print("Vypísať zoznam ................. 0")
        print("Pridať záznam .................. 1")
        print("Vyhľadať záznam podľa mena ..... 2")
        print("Odstrániť záznam ............... 3")
        print("Opraviť záznam ................. 4")
        print("Koniec ......................... ?")
        odpoved = input("Tvoja voľba: ")
        if odpoved == "0":
            vypisZoznam()
        elif odpoved == "1":
            pridajZaznam()
        elif odpoved == "2":
            hladat = input("Meno: ")
            vyhladajPodlaMena(hladat)
        elif odpoved == "3":
            odstran = input("Meno: ")
            odstranZaznam(odstran)
        elif odpoved == "4":
            oprav = input("Meno: ")
            opravZaznam(oprav)
        else:
            break

def vypisZoznam():
    f = open(nazov_suboru, "r")
    print(f.read())
    f.close()
    print()
```

```
def pridajZaznam():
    f = open(nazov_suboru, "r+")
    if f.read() != "":
        f.write('\n')
    f.write(input("Meno: ") + "\n")
    f.write(input("Adresa: ") + "\n")
    f.write(input("Telefónne číslo: "))
    f.close()
    print()

def vyhladajPodlaMena(hladat):
    f = open(nazov_suboru, "r")
    nasiel = False
    meno = f.readline().strip()
    while meno:
        adresa = f.readline().strip()
        cislo = f.readline().strip()
        if meno[:len(hladat)] == hladat:        # porovnanie len prvých len(hladat) znakov z mena!
            print(meno)
            print(adresa)
            print(cislo)
            nasiel = True
        meno = f.readline().strip()
    f.close()
    if not nasiel:
        print("Meno začínajúce na", hladat, "sa v databáze nevyskytuje!")
    print()

def odstranZaznam(odstran):
    f = open(nazov_suboru, "r")
    riadky = f.readlines()
    f.close()

    i = 0
    while i < len(riadky):
        if riadky[i].strip() == odstran:
            riadky.pop(i)
            riadky.pop(i)
            riadky.pop(i)
        i += 3

    f = open(nazov_suboru, "w")
    for riadok in riadky:
        f.write(riadok)
    f.close()
    print()

def opravZaznam(oprav):
    f = open(nazov_suboru, "r")
    riadky = f.readlines()
    f.close()

    i = 0
    while i < len(riadky):
        if riadky[i].strip() == oprav:
            riadky[i] = input("Nové meno: ") + "\n"
            riadky[i+1] = input("Nová adresa: ") + "\n"
            riadky[i+2] = input("Nové telefónne číslo: ")
            if (i+2) != (len(riadky)-1):
                riakdy[i+2] += "\n"
```

```
        else:
            i += 3

    f = open(nazov_suboru, "w")
    for riadok in riadky:
        f.write(riadok)
    f.close()
    print()
#=======================================
nazov_suboru = "telzoz.txt"
try:
    f = open(nazov_suboru, "r")
except FileNotFoundError:
    print("\nSúbor nebol nájdený, vytváram prázdny!\n")
    f = open(nazov_suboru, "w")
f.close()
ponuka()
```

Úloha F.10: Novovytvorený ostrov (po podmorskej erupcii) je rozparcelovaný na rovnaké parcely. Nepredaná parcela má číslo nula (0), predaná má číslo vlastníka (1 až 99) zapísané v súbore parcely.txt. Vytvorte program, ktorý vypíše koľko parciel je ešte na predaj a koľko to je percent zo všetkých parciel. Program doplňte o výpis, koľko parciel vlastnia jednotliví vlastníci, a tieto údaje nech sa zapíšu do súboru vlastnici.txt, kde bude aj počet nepredaných parciel.

Príklad súboru parcely.txt:

```
0 0 0 0 0 0 0 0
0 5 5 5 5 3 3 0
0 0 0 0 0 3 3 0
0 4 4 3 0 1 2 0
0 4 4 2 0 2 2 0
0 4 4 2 2 2 2 0
0 2 2 2 3 3 9 0
0 0 0 0 0 0 0 0
```

34 núl – t.j. na predaj je ešte 34 parciel, %... 34/64*100 = 53%

Obsah súboru vlastníci.txt:

```
0 34
1 1
2 11
3 7
4 6
atď.
```

Riešenie s využitím typu slovník:

```
def vratPocetNaPredaj(nazov_suboru):
    f = open(nazov_suboru)
    pocetNul = 0
    pocetParciel = 0
    riadok = f.readline()
    while riadok:
        for znak in riadok:
            if znak == "0":
                pocetNul += 1
        pocetParciel += len(riadok.split())
        riadok = f.readline()
    f.close()
    try:
        pocetPercent = pocetNul/pocetParciel*100
    except DivisionToZero:
        pocetPercent = 0
    return pocetNul, pocetPercent

def vratPoctyVlastnikov(nazov_suboru):
    f = open(nazov_suboru)
```

```
    vlastnici = {}                                  # vytvorí prázdny slovník
    riadok = f.readline()
    while riadok:
        cisla = riadok.split()                      # premenné cislo aj cisla mohli zostať typu str!
        for cislo in cisla:
            if cislo in vlastnici:
                vlastnici[cislo] += 1
            else:
                vlastnici[cislo] = 1
        riadok = f.readline()
    f.close()
    return vlastnici

def zapisDoSuboru(citat, zapisovat):
    zaznamy = vratPoctyVlastnikov(citat)
    f = open(zapisovat,"w")
    for vlastnik in sorted(zaznamy):                # zapíše utriedené podľa čísel vlastníkov
        f.write(vlastnik + " " + str(zaznamy[vlastnik]) + "\n")
    f.close()                                       # chyba krásy, na konci súboru bude prázdny riadok
#============================================================
nazov_suboru = "parcely.txt"
pocetNepredanych, tjPercent = vratPocetNaPredaj(nazov_suboru)
print("Na predaj je ešte {} parciel, čo je {:.0f} %.".format(pocetNepredanych, tjPercent))
zaznamy = vratPoctyVlastnikov(nazov_suboru)
print("Vlastník číslo    Počet parciel")
for vlastnik in sorted(zaznamy):
    print("{:>8} {:16}".format(vlastnik, zaznamy[vlastnik]))
zapisDoSuboru(nazov_suboru, "vlastnici.txt")
```

Výstup pre príklad súboru parcely.txt:

Na predaj je ešte 34 parciel, čo je 53 %.

Vlastník číslo	Počet parciel
0	34
1	1
2	11
3	7
4	6
5	4
9	1

Úloha F.11: Program slovensko-anglický prekladový slovník uvedený v študijnom texte Štruktúrované údajové typy Príklad S.3 upravte tak, aby databáza slovníka bola uložená v textovom súbore a podľa potreby sa dopĺňala.

Riešenie

Program sme doplnili len o načítanie dát z textového súboru do údajového typu slovník použitého v programe:

```
def main():
    nazov_suboru = "slovnikSA.txt"
    try:
        f = open(nazov_suboru,"r")
    except:
        f = open(nazov_suboru,"w")                  # ak súbor neexistoval
        f.write("začiatok\n")
        f.write("begin")
        print("Inicializoval som súbor "+nazov_suboru+"!")
    f.close()
```

```
    slovnikSA = {}                                  # vytvorenie prázdneho slovníka v pamäti
    f = open(nazov_suboru,"r+")                     # umožní novú dvojicu slov hneď zapísať na koniec súb.
    sl_slovo = f.readline().strip()
    while sl_slovo:                                 # naplnenie slovníka dátami z textového súboru
        an_slovo = f.readline().strip()
        slovnikSA[sl_slovo] = an_slovo
        sl_slovo = f.readline().strip()
```

a doplnili o zvýraznené príkazy

```
    while True:
        print()
        print("Preklad slovensko -> anglický ..... 0")
        print("Preklad anglicko -> slovenský ..... +")
        print("Koniec ............................ Enter")
        odpoved = input("Tvoja voľba: ")
        if odpoved == "":
            f.close()                               # uzavretie súboru na voľbu Koniec
            break
        if odpoved == "0":
            sl_slovo = input("Preložiť slovenské slovo: ")
            if sl_slovo in slovnikSA:
                print(sl_slovo, " - ", slovnikSA[sl_slovo])
            else:
                print(sl_slovo, " -  ?")
                an_slovo = input("Preklad: ")
                slovnikSA[sl_slovo] = an_slovo
                f.write("\n"+sl_slovo+"\n")         # zapísanie novej dvojice slov do textového súboru
                f.write(an_slovo)
        elif odpoved == "+":
            hladat = input("Preložiť anglické slovo: ")
            naslo_sa = False
            for sl_slovo, an_slovo in slovnikSA.items():
                if an_slovo == hladat:
                    print(an_slovo, " - ", sl_slovo)
                    naslo_sa = True
                    break
            if not naslo_sa:
                print(hladat, " -  ?")
                sl_slovo = input("Preklad: ")
                slovnikSA[sl_slovo] = hladat
                f.write("\n"+sl_slovo+"\n")         # zapísanie novej dvojice slov do textového súboru
                f.write(hladat)
        else:
            print("Stlačený zlý kláves!")
main()
```

Úloha F.12: Peťko sa potrebuje zdokonaliť v sčitovaní a Paľko v násobení. Súbor príklady.txt obsahuje zápis príkladov na sčítanie a násobenie. Vytvorte program, ktorý zo súboru vytvorí dva súbory petko.txt a palko.txt a vytvorí aj súbory petko_vysledky.txt a palko_vysledky.txt, ktoré doplní výsledkami, t.j. + 17 61 78 alebo * 15 8 120 atď., aby si Peťko aj Paľko mohli porovnať svoje výsledky so správnymi.

Príklady súborov:	priklady.txt	petko.txt	palko.txt	petko_vysledky.txt	palko_vysledky.txt
	+ 17 61	+ 17 61	* 15 8	+ 17 61 78	* 15 8 120
	* 15 8	+ 14 29	* 2 45	+ 14 29 43	* 2 45 90
	* 2 45				
	+ 14 29				

Riešenie:

```
def vytvorPlusKrat(subor_zdroj, subor_plus, subor_krat):
    f = open(subor_zdroj)
    plus = open(subor_plus,"w")
    krat = open(subor_krat,"w")

    for riadok in f:
        if riadok[0] == "+":
            plus.write(riadok)
        else:
            krat.write(riadok)
    f.close()
    plus.close()
    krat.close()

def vypocitaj(subor_zdroj):
    f = open(subor_zdroj)
    g = open(subor_zdroj[:-4]+"_vysledky.txt","w")
    for riadok in f:
        udaje = riadok.split()
        if udaje[0] == "+":
            g.write(riadok.strip()+" "+str(int(udaje[1])+int(udaje[2]))+"\n")
        else:
            g.write(riadok.strip()+" "+str(int(udaje[1])*int(udaje[2]))+"\n")
    f.close()
    g.close()
#=============================================================================
subor_zdroj = "priklady.txt"
subor_plus = "petko.txt"
subor_krat = "palko.txt"
vytvorPlusKrat(subor_zdroj, subor_plus, subor_krat)
vypocitaj("petko.txt")
vypocitaj("palko.txt")
```