# 31.2.1 Duomenų struktūrų naudojimas

## Ką reikia mokėti

* Mokėti naudotis (programuoti) standartinėmis struktūromis;
* Mokėti aptarti abstrakčiųjų duomenų struktūrų pavyzdžius (dinaminis masyvas ir pan.);
* Mokėti, kokius veiksmus galima atlikti su aptartomis duomenų struktūromis;
* Mokėti naudoti sudėtines duomenų struktūras (struktūrų/klasių masyvas, masyvai struktūroje/klasėje);

Šia diagnostinę užduotį sudaro 4 praktinės užduotys.   
IV gimnazijų klasėjeoOrientuojamasi į praktinį programavimą, nes ir VBE II dalyje - tik praktinės užduotys.

## Praktinės užduotys

### 1 užduotis

Parašykite programą, kuri sužymėtų skaičius į lyginius ir nelyginius intervale 1 <= n <= 10, sudėtų juos į sąrašą arba masyvą ir atspausdintų. Skaičiaus aprašymui naudokite struktūrą (arba klasę) pavadinimu *Skaičius*, kuri turėtų du narius: reikšmę ir lygiškumo požymį.

| Pradiniai duomenys | Rezultatai |
| --- | --- |
| - | Skaičius: 1 yra nelyginis. |
|  | Skaičius: 2 yra lyginis. |
|  | Skaičius: 3 yra nelyginis. |
|  | Skaičius: 4 yra lyginis. |
|  | Skaičius: 5 yra nelyginis. |
|  | Skaičius: 6 yra lyginis. |
|  | Skaičius: 7 yra nelyginis. |
|  | Skaičius: 8 yra lyginis. |
|  | Skaičius: 9 yra nelyginis. |
|  | Skaičius: 10 yra lyginis. |

#### Vertinimas

| Nr. | Kas vertinama | Balai |
| --- | --- | --- |
| 1. | Skaičiui aprašyti sudaryta tinkama struktūra | 2 |
| 2. | Skaičių struktūros užpildymas ir surašymas į masyvą arba sąrašą | 2 |
| 3. | Skaičių informacijos spausdinimas | 2 |
|  | Viso: | 6 |

#### Galimas teisingas atsakymas

C++ sprendimas:

#include <iostream>  
using namespace std;  
  
struct Skaičius {  
 int reikšmė;   
 bool yraLyginis;  
};  
  
int main() {  
 Skaičius skaičiai[10];  
  
 for (int i = 0; i < 10; i++) {  
 skaičiai[i].reikšmė = i + 1;  
 skaičiai[i].yraLyginis = (skaičiai[i].reikšmė % 2 == 0);  
 }  
  
 for (int i = 0; i < 10; i++) {  
 cout << "Skaičius: " << skaičiai[i].reikšmė;  
 if (skaičiai[i].yraLyginis) {  
 cout << " yra lyginis." << endl;  
 } else {  
 cout << " yra nelyginis." << endl;  
 }  
 }  
  
 return 0;  
}

Python sprendimas:

1 variantas:

skaičiai = []

for i in range(10):

skaičius = {

"reikšmė": i + 1,

"yra\_lyginis": (i + 1) % 2 == 0

}

skaičiai.append(skaičius)

for skaičius in skaičiai:

print(f"Skaičius: {skaičius['reikšmė']}", end="")

if skaičius["yra\_lyginis"]:

print(" yra lyginis.")

else:

print(" yra nelyginis.")

2 sprendimo variantas (jame naudojama „class“ – ne bendrosios informatikos programos objektas)

class Skaičius:  
 def \_\_init\_\_(self, reikšmė):  
 self.reikšmė = reikšmė  
 self.yra\_lyginis = (reikšmė % 2 == 0)  
  
skaičiai = [Skaičius(i + 1) for i in range(10)]  
  
for skaičius in skaičiai:  
 print(f"Skaičius: {skaičius.reikšmė}", end="")  
 if skaičius.yra\_lyginis:  
 print(" yra lyginis.")  
 else:  
 print(" yra nelyginis.")

#### Taškų skaičius

6

#### Sudėtingumo lygis

3

#### Ugdomos kompetencijos

Pažinimo

### 2 užduotis

CSV formato faile yra mokinių sąrašas. Jame parašytas kiekvieno mokinio vardas, pavardė, klasė ir 5 pažymiai. Parašykite programą, kuri nuskaitytų kiekvieno mokinio duomenis, kiekvienam mokiniui suskaičiuotų jo pažymių vidurkį ir visą informaciją (kartu su suskaičiuotu vidurkiu) sudėtų į mokinių sąrašą (masyvą) bei atspausdintų mokinių vardus ir vidurkius (spausdinkite 2 skaitmenis po kablelio). Mokinio informacijai saugoti (įtraukiant vidurkį) turi būti sudaryta klasė/struktūra.

| duomenys.csv | Rezultatai |
| --- | --- |
| Jonas,Jonaitis,10,8,7,9,6,10 | Jonas Jonaitis: 8.00 |
| Ona,Petrauskaite,10,10,9,10,8,7 | Ona Petrauskaite: 8.80 |
| Petras,Jankauskas,9,7,6,8,5,9 | Petras Jankauskas: 7.00 |
| Raminta,Balsyte,10,9,10,9,8,10 | Raminta Balsyte: 9.20 |
| Rokas,Janonis,10,6,7,5,8,7 | Rokas Janonis: 6.60 |

duomenys.csv:

Jonas,Jonaitis,10,8,7,9,6,10  
Ona,Petrauskaite,10,10,9,10,8,7  
Petras,Jankauskas,9,7,6,8,5,9  
Raminta,Balsyte,10,9,10,9,8,10  
Rokas,Janonis,10,6,7,5,8,7

Rezultatai:

Jonas Jonaitis: 8.00  
Ona Petrauskaite: 8.80  
Petras Jankauskas: 7.00  
Raminta Balsyte: 9.20  
Rokas Janonis: 6.60

#### Vertinimas

| Nr. | Kas vertinama | Balai |
| --- | --- | --- |
| 1. | Mokiniui aprašyti sudaryta tinkama struktūra | 2 |
| 2. | Duomenų nuskaitymas į masyvą arba sąrašą | 2 |
| 3. | Mokinio vidurkio skaičiavimas ir priskyrimas struktūrai | 2 |
| 4. | Mokinių sąrašo spausdinimas | 1 |
|  | Viso: | 7 |

#### Teisingas atsakymas

C++ sprendimas:

#include <iostream>  
#include <fstream>  
#include <sstream>  
#include <vector>  
#include <string>  
#include <iomanip>  
using namespace std;  
  
struct Mokinys {  
 string vardas;  
 string pavarde;  
 string klase;  
 vector<int> pazymiai;  
 double vidurkis;  
};  
  
double skaiciuotiVidurki(const vector<int>& pazymiai) {  
 int suma = 0;  
 for (int pazymys : pazymiai) {  
 suma += pazymys;  
 }  
 return double(suma) / pazymiai.size();  
}  
  
int main() {  
 ifstream csvfile("duomenys.csv");  
 if (!csvfile.is\_open()) {  
 cerr << "Nepavyko atidaryti failo!" << endl;  
 return 1;  
 }  
 string eilute;  
 vector<Mokinys> mokiniai;  
  
 while (getline(csvfile, eilute)) {  
 stringstream sstream(eilute);  
 string reiksme;  
 vector<string> duomenys;  
  
 while (getline(sstream, reiksme, ',')) {  
 duomenys.push\_back(reiksme);  
 }  
  
 Mokinys mokinys;  
 mokinys.vardas = duomenys[0];  
 mokinys.pavarde = duomenys[1];  
 mokinys.klase = duomenys[2];  
  
 for (int i = 3; i < duomenys.size(); ++i) {  
 mokinys.pazymiai.push\_back(stoi(duomenys[i]));  
 }  
  
 mokinys.vidurkis = skaiciuotiVidurki(mokinys.pazymiai);  
 mokiniai.push\_back(mokinys);  
 }  
  
 cout << fixed << setprecision(2);  
 for (int i = 0; i < mokiniai.size(); ++i) {  
 cout << mokiniai[i].vardas << " " << mokiniai[i].pavarde << ": " << mokiniai[i].vidurkis << endl;  
 }  
  
 return 0;  
}

Python sprendimas:

import csv  
  
def skaičiuoti\_vidurkį(pažymiai):  
 return sum(pažymiai) / len(pažymiai)  
  
with open('duomenys.csv', newline='') as csvfile:  
 skaitytuvas = csv.reader(csvfile)  
 mokiniai = []  
   
 for eilutė in skaitytuvas:  
 vardas = eilutė[0]  
 pavardė = eilutė[1]  
 klasė = eilutė[2]  
 pažymiai = list(map(int, eilutė[3:]))  
 vidurkis = skaičiuoti\_vidurkį(pažymiai)  
   
 mokiniai.append({  
 'vardas': vardas,  
 'pavardė': pavardė,  
 'klasė': klasė,  
 'pažymiai': pažymiai,  
 'vidurkis': vidurkis  
 })  
  
for mokinys in mokiniai:  
 print(f"{mokinys['vardas']} {mokinys['pavardė']}: {mokinys['vidurkis']:.2f}")

#### Taškų skaičius

7

#### Sudėtingumo lygis

4

#### Ugdomos kompetencijos

Pažinimo

### 3 užduotis

CSV formato faile yra automobilių ir jų savininkų sąrašas faile *automobiliai.csv*. Savininko informacija įtraukia asmens kodą, vardą, pavardę bei automobilį. Automobilis turi markės, modelio, leidimo metų ir automobilio amžiaus informaciją. Programa turi apskaičiuoti ir atnaujinti automobilio amžių, kai yra žinoma, kad einamieji metai yra 2024. Programa turi sukurti failą *rezultatai.csv* ir jame CSV formatu įrašyti automobilių amžiaus ir jų savininkų informaciją, kurių automobilių amžius viršija 10 metų.

Automobiliams ir jų savininkams saugoti turi būti sudaryto dvi atskiros klasės/struktūros. Savininko struktūra turi įtraukti automobilio struktūrą.

automobiliai.csv:

Toyota,Corolla,2010,0,37511060442,Jonas,Jonaitis  
Honda,Civic,2015,0,37456789012,Marija,Petrauskaite  
Ford,Focus,2008,0,38567890123,Petras,Petraitis  
BMW,X5,2005,0,45678901234,Rita,Kazlauskiene  
Audi,A4,2012,0,46789012345,Antanas,Kazlauskas

rezultatai.csv:

14,37511060442,Jonas,Jonaitis  
16,38567890123,Petras,Petraitis  
19,45678901234,Rita,Kazlauskiene  
12,46789012345,Antanas,Kazlauskas

#### Vertinimas

| Nr. | Kas vertinama | Balai |
| --- | --- | --- |
| 1. | Sukurtos dvi struktūros (viena automobiliui, kita savininkui) | 2 |
| 2. | Duomenų struktūra yra sudėtinė (viena struktūra įtraukia kitą) | 2 |
| 3. | Duomenų nuskaitymas | 1 |
| 4. | Automobilio amžiaus skaičiavimas ir priskyrimas struktūrai | 2 |
| 5. | Rezultatų išvedimas į failą | 1 |
|  | Viso: | 8 |

#### Teisingas atsakymas

C++ sprendimas:

#include <iostream>  
#include <fstream>  
#include <sstream>  
#include <vector>  
#include <string>  
  
using namespace std;  
  
struct Automobilis {  
 string marke;  
 string modelis;  
 int leidimoMetai;  
 int amzius;  
};  
  
struct Savininkas {  
 string asmensKodas;  
 string vardas;  
 string pavarde;  
 Automobilis automobilis;  
};  
  
int skaiciuotiAmziu(int leidimoMetai, int einamiejiMetai) {  
 return einamiejiMetai - leidimoMetai;  
}  
  
int main() {  
 int einamiejiMetai = 2024;  
  
 ifstream csvfile("automobiliai.csv");  
 if (!csvfile.is\_open()) {  
 cerr << "Nepavyko atidaryti failo automobiliai.csv" << endl;  
 return 1;  
 }  
  
 string eilute;  
 vector<Savininkas> savininkai;  
  
 while (getline(csvfile, eilute)) {  
 stringstream sstream(eilute);  
 string reiksme;  
 vector<string> duomenys;  
  
 while (getline(sstream, reiksme, ',')) {  
 duomenys.push\_back(reiksme);  
 }  
  
 Savininkas savininkas;  
 savininkas.automobilis.marke = duomenys[0];  
 savininkas.automobilis.modelis = duomenys[1];  
 savininkas.automobilis.leidimoMetai = stoi(duomenys[2]);  
 savininkas.automobilis.amzius = skaiciuotiAmziu(stoi(duomenys[2]), einamiejiMetai);  
 savininkas.asmensKodas = duomenys[4];  
 savininkas.vardas = duomenys[5];  
 savininkas.pavarde = duomenys[6];  
  
 savininkai.push\_back(savininkas);  
 }  
  
 csvfile.close();  
  
 ofstream rezultataiFile("rezultatai.csv");  
 if (!rezultataiFile.is\_open()) {  
 cerr << "Nepavyko sukurti failo rezultatai.csv" << endl;  
 return 1;  
 }  
  
 for (int i = 0; i < savininkai.size(); ++i) {  
 if (savininkai[i].automobilis.amzius > 10) {  
 rezultataiFile << savininkai[i].automobilis.amzius << ","  
 << savininkai[i].asmensKodas << ","  
 << savininkai[i].vardas << ","  
 << savininkai[i].pavarde << endl;  
 }  
 }  
  
 rezultataiFile.close();  
  
 return 0;  
}

Python sprendimas (1 variantas):

def skaiciuoti\_amziu(leidimo\_metai, einamieji\_metai):

return einamieji\_metai - leidimo\_metai

einamieji\_metai = 2024

savininkai = []

# Skaityti duomenis iš failo

with open("automobiliai.csv", "r") as csvfile:

for eilute in csvfile:

duomenys = eilute.strip().split(',')

# Sukuriame žodyną automobilio duomenims

automobilis = {

"marke": duomenys[0],

"modelis": duomenys[1],

"leidimo\_metai": int(duomenys[2]),

"amzius": skaiciuoti\_amziu(int(duomenys[2]), einamieji\_metai)

}

# Sukuriame žodyną savininko duomenims

savininkas = {

"asmens\_kodas": duomenys[4],

"vardas": duomenys[5],

"pavarde": duomenys[6],

"automobilis": automobilis

}

# Pridedame savininką į sąrašą

savininkai.append(savininkas)

# Rašome rezultatus į failą

with open("rezultatai.txt", "w") as rezultatai\_file:

for savininkas in savininkai:

if savininkas["automobilis"]["amzius"] > 10:

rezultatai\_file.write(

f"{savininkas['automobilis']['amzius']},{savininkas['asmens\_kodas']},"

f"{savininkas['vardas']},{savininkas['pavarde']}\n"

)

Python sprendimas (2 variantas, kuriame naudojamos klasės)

class Automobilis:  
 def \_\_init\_\_(self, marke, modelis, leidimo\_metai, amzius=0):  
 self.marke = marke  
 self.modelis = modelis  
 self.leidimo\_metai = leidimo\_metai  
 self.amzius = amzius  
  
class Savininkas:  
 def \_\_init\_\_(self, asmens\_kodas, vardas, pavarde, automobilis):  
 self.asmens\_kodas = asmens\_kodas  
 self.vardas = vardas  
 self.pavarde = pavarde  
 self.automobilis = automobilis  
  
def skaiciuoti\_amziu(leidimo\_metai, einamieji\_metai):  
 return einamieji\_metai - leidimo\_metai  
  
einamieji\_metai = 2024  
savininkai = []  
  
with open("automobiliai.csv", "r") as csvfile:  
 for eilute in csvfile:  
 duomenys = eilute.strip().split(',')  
  
 automobilis = Automobilis(  
 marke=duomenys[0],  
 modelis=duomenys[1],  
 leidimo\_metai=int(duomenys[2]),  
 )  
 automobilis.amzius = skaiciuoti\_amziu(automobilis.leidimo\_metai, einamieji\_metai)  
  
 savininkas = Savininkas(  
 asmens\_kodas=duomenys[4],  
 vardas=duomenys[5],  
 pavarde=duomenys[6],  
 automobilis=automobilis  
 )  
 savininkai.append(savininkas)  
  
with open("rezultatai.txt", "w") as rezultatai\_file:  
 for savininkas in savininkai:  
 if savininkas.automobilis.amzius > 10:  
 rezultatai\_file.write(  
 f"{savininkas.automobilis.amzius},{savininkas.asmens\_kodas},"  
 f"{savininkas.vardas},{savininkas.pavarde}\n"  
 )

#### Taškų skaičius

8

#### Sudėtingumo lygis

4

#### Ugdomos kompetencijos

Pažinimo

### 4 užduotis

CSV formato faile yra automobilių ir jų savininkų sąrašas faile *automobiliai.csv*. Savininko informacija įtraukia asmens kodą, vardą ir pavardę ir turimus automobilius. Automobilis turi markės, modelio, leidimo metų ir automobilio amžiaus informaciją. Savininkas gali turėti kelis automobilius. Savininką identifikuoja asmens kodas. Programa turi apskaičiuoti ir atnaujinti automobilio amžių, kai yra žinoma, kad einamieji metai yra 2024. Programa turi sukurti failą *rezultatai.csv* ir jame CSV formatu įrašyti pagal pavardę ir vardą surikiuotą automobilių ir jų savininkų informaciją (automobilio amžius, savininko asmens kodas, automobilio markė ir modelis, vardas, pavardė), kurių automobilių amžius viršija 10 metų.

automobiliai.csv:

Toyota,Corolla,2010,0,37511060442,Jonas,Jonaitis  
Honda,Civic,2015,0,37456789012,Marija,Petrauskaite  
Ford,Focus,2008,0,38567890123,Petras,Petraitis  
BMW,X5,2005,0,45678901234,Rita,Kazlauskiene  
Audi,A4,2012,0,46789012345,Antanas,Kazlauskas  
Toyota,Camry,2005,0,37511060442,Jonas,Jonaitis  
Volkswagen,Passat,2007,0,38567890123,Petras,Petraitis  
Nissan,Altima,2011,0,45678901234,Rita,Kazlauskiene

rezultatai.csv:

14,37511060442,Toyota,Corolla,Jonas,Jonaitis  
19,37511060442,Toyota,Camry,Jonas,Jonaitis  
12,46789012345,Audi,A4,Antanas,Kazlauskas  
19,45678901234,BMW,X5,Rita,Kazlauskiene  
13,45678901234,Nissan,Altima,Rita,Kazlauskiene  
16,38567890123,Ford,Focus,Petras,Petraitis  
17,38567890123,Volkswagen,Passat,Petras,Petraitis

#### Vertinimas

| Nr. | Kas vertinama | Balai |
| --- | --- | --- |
| 1. | Sukurtos dvi struktūros (viena automobiliui, kita savininkui), savininko struktūroje įtrauktas automobilių masyvas/sąrašas | 3 |
| 3. | Duomenų nuskaitymas | 2 |
| 4. | Automobilio amžiaus skaičiavimas ir priskyrimas struktūrai | 2 |
| 4. | Savininkų masyvo/sąrašo rūšiavimas pagal pavardę ir vardą | 2 |
| 5. | Rezultatų išvedimas į failą | 1 |
|  |  | 10 |

#### Teisingas atsakymas

C++ sprendimas:

#include <iostream>  
#include <fstream>  
#include <sstream>  
#include <vector>  
#include <string>  
#include <algorithm>  
  
using namespace std;  
  
struct Automobilis {  
 string marke;  
 string modelis;  
 int leidimoMetai;  
 int amzius;  
};  
  
struct Savininkas {  
 string asmensKodas;  
 string vardas;  
 string pavarde;  
 vector<Automobilis> automobiliai;  
};  
  
int skaiciuotiAmziu(int leidimoMetai, int einamiejiMetai) {  
 return einamiejiMetai - leidimoMetai;  
}  
  
int main() {  
 int einamiejiMetai = 2024;  
 ifstream csvfile("automobiliai.csv");  
 if (!csvfile.is\_open()) {  
 cerr << "Nepavyko atidaryti failo automobiliai.csv" << endl;  
 return 1;  
 }  
  
 string eilute;  
 vector<Savininkas> savininkai;  
  
 while (getline(csvfile, eilute)) {  
 stringstream sstream(eilute);  
 string reiksme;  
 vector<string> duomenys;  
  
 while (getline(sstream, reiksme, ',')) {  
 duomenys.push\_back(reiksme);  
 }  
  
 Automobilis automobilis;  
 automobilis.marke = duomenys[0];  
 automobilis.modelis = duomenys[1];  
 automobilis.leidimoMetai = stoi(duomenys[2]);  
 automobilis.amzius = skaiciuotiAmziu(stoi(duomenys[2]), einamiejiMetai);  
  
 string asmensKodas = duomenys[4];  
 bool found = false;  
  
 for (int i = 0; i < savininkai.size(); ++i) {  
 if (savininkai[i].asmensKodas == asmensKodas) {  
 savininkai[i].automobiliai.push\_back(automobilis);  
 found = true;  
 break;  
 }  
 }  
  
 if (!found) {  
 Savininkas savininkas;  
 savininkas.asmensKodas = asmensKodas;  
 savininkas.vardas = duomenys[5];  
 savininkas.pavarde = duomenys[6];  
 savininkas.automobiliai.push\_back(automobilis);  
 savininkai.push\_back(savininkas);  
 }  
 }  
  
 csvfile.close();  
  
 sort(savininkai.begin(), savininkai.end(), [](const Savininkas& a, const Savininkas& b) {  
 if (a.pavarde == b.pavarde) return a.vardas < b.vardas;  
 return a.pavarde < b.pavarde;  
 });  
  
 ofstream rezultataiFile("rezultatai.csv");  
 if (!rezultataiFile.is\_open()) {  
 cerr << "Nepavyko sukurti failo rezultatai.csv" << endl;  
 return 1;  
 }  
  
 for (int i = 0; i < savininkai.size(); ++i) {  
 for (int j = 0; j < savininkai[i].automobiliai.size(); ++j) {  
 if (savininkai[i].automobiliai[j].amzius > 10) {  
 rezultataiFile << savininkai[i].automobiliai[j].amzius << ","  
 << savininkai[i].asmensKodas << ","  
 << savininkai[i].automobiliai[j].marke << ","  
 << savininkai[i].automobiliai[j].modelis << ","  
 << savininkai[i].vardas << ","  
 << savininkai[i].pavarde << endl;  
 }  
 }  
 }  
  
 rezultataiFile.close();  
  
 return 0;  
}

Python sprendimas (1 variantas):

def skaiciuoti\_amziu(leidimo\_metai, einamieji\_metai):

return einamieji\_metai - leidimo\_metai

einamieji\_metai = 2024

savininkai = []

with open("automobiliai.csv", "r") as csvfile:

for eilute in csvfile:

duomenys = eilute.strip().split(',')

# Sukuriame žodyną automobilio duomenims

automobilis = {

"marke": duomenys[0],

"modelis": duomenys[1],

"leidimo\_metai": int(duomenys[2]),

"amzius": skaiciuoti\_amziu(int(duomenys[2]), einamieji\_metai)

}

# Patikriname, ar savininkas jau egzistuoja sąraše

asmens\_kodas = duomenys[4]

found = False

for savininkas in savininkai:

if savininkas["asmens\_kodas"] == asmens\_kodas:

savininkas["automobiliai"].append(automobilis)

found = True

break

# Jei savininkas nerastas, sukuriame naują savininką

if not found:

savininkas = {

"asmens\_kodas": asmens\_kodas,

"vardas": duomenys[5],

"pavarde": duomenys[6],

"automobiliai": [automobilis]

}

savininkai.append(savininkas)

# Rikiuojame savininkus pagal pavardę ir vardą

sorted\_savininkai = sorted(

savininkai,

key=lambda s: (s["pavarde"], s["vardas"])

)

# Rašome rezultatus į failą

with open("rezultatai.csv", "w") as rezultatai\_file:

for savininkas in sorted\_savininkai:

for automobilis in savininkas["automobiliai"]:

if automobilis["amzius"] > 10:

rezultatai\_file.write(

f"{automobilis['amzius']},{savininkas['asmens\_kodas']},"

f"{automobilis['marke']},{automobilis['modelis']},"

f"{savininkas['vardas']},{savininkas['pavarde']}\n"

)

Python sprendimas (2 variantas, kuriame panaudojamos klasės)

from operator import attrgetter  
  
class Automobilis:  
 def \_\_init\_\_(self, marke, modelis, leidimo\_metai):  
 self.marke = marke  
 self.modelis = modelis  
 self.leidimo\_metai = leidimo\_metai  
 self.amzius = 0  
  
class Savininkas:  
 def \_\_init\_\_(self, asmens\_kodas, vardas, pavarde):  
 self.asmens\_kodas = asmens\_kodas  
 self.vardas = vardas  
 self.pavarde = pavarde  
 self.automobiliai = []  
  
def skaiciuoti\_amziu(leidimo\_metai, einamieji\_metai):  
 return einamieji\_metai - leidimo\_metai  
  
einamieji\_metai = 2024  
savininkai = []  
  
with open("automobiliai.csv", "r") as csvfile:  
 for eilute in csvfile:  
 duomenys = eilute.strip().split(',')  
  
 automobilis = Automobilis(  
 marke=duomenys[0],  
 modelis=duomenys[1],  
 leidimo\_metai=int(duomenys[2])  
 )  
 automobilis.amzius = skaiciuoti\_amziu(automobilis.leidimo\_metai, einamieji\_metai)  
  
 asmens\_kodas = duomenys[4]  
 found = False  
 for savininkas in savininkai:  
 if savininkas.asmens\_kodas == asmens\_kodas:  
 savininkas.automobiliai.append(automobilis)  
 found = True  
 break  
  
 if not found:  
 savininkas = Savininkas(  
 asmens\_kodas=asmens\_kodas,  
 vardas=duomenys[5],  
 pavarde=duomenys[6]  
 )  
 savininkas.automobiliai.append(automobilis)  
 savininkai.append(savininkas)  
  
sorted\_savininkai = sorted(savininkai, key=attrgetter('pavarde', 'vardas'))  
  
with open("rezultatai.txt", "w") as rezultatai\_file:  
 for savininkas in sorted\_savininkai:  
 for automobilis in savininkas.automobiliai:  
 if automobilis.amzius > 10:  
 rezultatai\_file.write(  
 f"{automobilis.amzius},{savininkas.asmens\_kodas},"  
 f"{automobilis.marke},{automobilis.modelis},"  
 f"{savininkas.vardas},{savininkas.pavarde}\n"  
 )

#### Taškų skaičius

10

#### Sudėtingumo lygis

4

#### Ugdomos kompetencijos

Pažinimo