# 31.2.2 Algoritmai

## Ką reikia mokėti

* Užrašyti loginius reiškinius;
* Juos panaudoti masyvo (ar struktūrų (klasių) masyvo (sąrašo) rikiavimo, reikšmių paieškos, reikšmių šalinimo, reikšmių papildymo veiksmuose;
* Mokėti standartinius kalbos ir savo realizacijos rikiavimo, paieškos algoritmus;

## Praktinės užduotys

### 1 užduotis

Laikyto egzamino rezultatai yra pateikiami skaičių eilute. Parašykite programą, kuri pateiktų išlaikiusių studentų skaičių (gavo ne mažesnį įvertinimą nei 5), neišlaikiusių studentų skaičių (gavo mažesnį įvertinimą nei 3) ir skaičių studentų, kurie turi teisę perlaikyti egzaminą (gavo 3 arba 4).

| Pradiniai duomenys | Rezultatai |
| --- | --- |
| 6 8 3 4 7 2 5 4 | Išlaikė: 4 |
|  | Gali perlaikyti: 3 |
|  | Neišlaikė: 1 |

#### Vertinimas

| Nr. | Kas vertinama | Balai |
| --- | --- | --- |
| 1. | Įvesties eilutės nuskaitymas ir masyvo sudarymas | 1 |
| 2. | Logikos įgyvendinimas | 4 |
| 3. | Rezultatų spausdinimas | 1 |
|  | Viso: | 6 |

*Pastaba: Logiką įmanoma įgyvendinti ir nepritaikius loginio operatoriaus. Tai irgi skaitosi geras sprendimas.*

#### Teisingas atsakymas

C++ sprendimas:

#include <iostream>

#include <sstream>

using namespace std;

int main() {

const int max\_dydis = 20;

int skaiciai[max\_dydis];   
 int dydis = 0;   
 int islaike = 0;  
 int perlaikyti = 0;  
 int neislaike = 0;  
 string ivestis;

cout << "Įveskite pažymius: ";

getline(cin, ivestis);

stringstream ss(ivestis);

int pazymys;  
  
 while (ss >> pazymys && dydis < max\_dydis) {  
 skaiciai[dydis] = pazymys;  
 dydis++;  
 }  
  
 for (int i = 0; i < dydis; i++) {  
 if (skaiciai[i] >= 5) {  
 islaike++;  
 } else if (skaiciai[i] == 3 || skaiciai[i] == 4) {  
 perlaikyti++;  
 } else {  
 neislaike++;  
 }  
 }  
  
 cout << "Išlaikė: " << islaike << endl;  
 cout << "Gali perlaikyti: " << perlaikyti << endl;  
 cout << "Neišlaikė: " << neislaike << endl;  
  
 return 0;  
}

Python sprendimas:

ivestis = input("Įveskite pažymius atskirtus tarpais: ")  
pazymiai = list(map(int, ivestis.split()))  
islaike = 0  
perlaikyti = 0  
neislaike = 0  
  
for pazymys in pazymiai:  
 if pazymys >= 5:  
 islaike += 1  
 elif pazymys == 3 or pazymys == 4:  
 perlaikyti += 1  
 else:  
 neislaike += 1  
  
print(f"Išlaikė: {islaike}")  
print(f"Gali perlaikyti: {perlaikyti}")  
print(f"Neišlaikė: {neislaike}")

#### Taškų skaičius

6

#### Sudėtingumo lygis

2

#### Ugdomos kompetencijos

Pažinimo

### 2.1 užduotis

CSV formato faile (**duomenys.csv**) yra įvykusių rungtynių statistikos sąrašas: žaidėjo vardas, komanda, pelnytų taškų skaičius (PTS), žaidimo metu atliktų metimų kiekis (FGA) ir mestų baudų kiekis (FTA). Tikrasis žaidėjo taiklumas (TS) apskaičiuojamas pagal formulę: TS = PTS / (2 \* (FGA + 0.44 \* FTA)). Parašykite programą, kuri nuskaitytų duomenis, kiekvienam žaidėjui suskaičiuotų tikrojo taiklumo parametrą, surikiuotų duomenis pagal komandą ir taiklumą bei juos išvestų csv formatu į failą **rezultatai.csv** (formatas: žaidėjas, komanda, PTS, FGA, FTA, TS).

duomenys.csv:

LeBron James,Los Angeles Lakers,25,18,7  
Klay Thompson,Golden State Warriors,18,14,2  
Austin Reaves,Los Angeles Lakers,15,10,5  
D'Angelo Russell,Los Angeles Lakers,17,12,4  
Rui Hachimura,Los Angeles Lakers,12,8,3  
Stephen Curry,Golden State Warriors,30,20,8  
Anthony Davis,Los Angeles Lakers,22,16,6  
Draymond Green,Golden State Warriors,8,6,3  
Andrew Wiggins,Golden State Warriors,16,11,4  
Chris Paul,Golden State Warriors,9,7,2

rezultatai.csv:

Stephen Curry,Golden State Warriors,30,20,8,0.64  
Andrew Wiggins,Golden State Warriors,16,11,4,0.63  
Klay Thompson,Golden State Warriors,18,14,2,0.60  
Chris Paul,Golden State Warriors,9,7,2,0.57  
Draymond Green,Golden State Warriors,8,6,3,0.55  
Rui Hachimura,Los Angeles Lakers,12,8,3,0.64  
D'Angelo Russell,Los Angeles Lakers,17,12,4,0.62  
Austin Reaves,Los Angeles Lakers,15,10,5,0.61  
LeBron James,Los Angeles Lakers,25,18,7,0.59  
Anthony Davis,Los Angeles Lakers,22,16,6,0.59

#### Vertinimas

| Nr. | Kas vertinama | Balai |
| --- | --- | --- |
| 1. | Žaidėjui aprašyti sudaryta tinkama struktūra | 1 |
| 2. | Duomenų nuskaitymas į masyvą arba sąrašą | 2 |
| 3. | Tikrojo taiklumo skaičiavimas ir priskyrimas struktūrai | 2 |
| 4. | Rūšiavimas pagal du laukus | 2 |
| 5. | Rezultatų išvedimas į failą | 1 |
|  | Viso: | 8 |

#### Teisingas atsakymas

C++ sprendimas:

#include <iostream>  
#include <fstream>  
#include <sstream>  
#include <string>  
#include <vector>  
#include <algorithm>  
#include <iomanip>  
  
using namespace std;  
  
struct Zaidejas {  
 string vardas;  
 string komanda;  
 int pts;  
 int fga;  
 int fta;  
 double ts;  
};  
  
double skaiciuoti\_ts(int pts, int fga, int fta) {  
 if (fga + 0.44 \* fta == 0) {  
 return 0; // Išvengiame dalybos iš nulio  
 }  
 return (double)(pts) / (2 \* (fga + 0.44 \* fta));  
}  
  
vector<Zaidejas> nuskaityti\_zaidejus(const string& failo\_pavadinimas) {  
 vector<Zaidejas> zaidejai;  
 ifstream failas(failo\_pavadinimas);  
  
 if (!failas.is\_open()) {  
 cerr << "Nepavyko atidaryti failo: " << failo\_pavadinimas << endl;  
 return zaidejai;  
 }  
  
 string eilute;  
 while (getline(failas, eilute)) {  
 stringstream ss(eilute);  
 string vardas, komanda, pts\_str, fga\_str, fta\_str;  
 getline(ss, vardas, ',');  
 getline(ss, komanda, ',');  
 getline(ss, pts\_str, ',');  
 getline(ss, fga\_str, ',');  
 getline(ss, fta\_str, ',');  
  
 int pts = stoi(pts\_str);  
 int fga = stoi(fga\_str);  
 int fta = stoi(fta\_str);

Zaidejas zaidejas;  
 zaidejas.vardas = vardas;  
 zaidejas.komanda = komanda;  
 zaidejas.pts = pts;  
 zaidejas.fga = fga;  
 zaidejas.fta = fta;  
 zaidejas.ts = skaiciuoti\_ts(pts, fga, fta);  
  
 zaidejai.push\_back(zaidejas);  
 }  
  
 failas.close();  
 return zaidejai;  
}  
  
void isvesti\_zaidejus(vector<Zaidejas>& zaidejai, const string& failo\_pavadinimas) {  
 sort(zaidejai.begin(), zaidejai.end(), [](const Zaidejas& a, const Zaidejas& b) {  
 if (a.komanda == b.komanda) {  
 return a.ts > b.ts;  
 }  
 return a.komanda < b.komanda;  
 });  
  
 ofstream failas(failo\_pavadinimas);  
 if (!failas.is\_open()) {  
 cerr << "Nepavyko sukurti failo: " << failo\_pavadinimas << endl;  
 return;  
 }  
  
 for (int i = 0; i < zaidejai.size(); ++i) {  
 failas << zaidejai[i].vardas << ","  
 << zaidejai[i].komanda << ","  
 << zaidejai[i].pts << ","  
 << zaidejai[i].fga << ","  
 << zaidejai[i].fta << ","  
 << fixed << setprecision(2) << zaidejai[i].ts << "\n";  
 }  
  
 failas.close();  
}  
  
int main() {  
 string duomenys = "duomenys.csv";  
 string rezultatai = "rezultatai.csv";  
  
 vector<Zaidejas> zaidejai = nuskaityti\_zaidejus(duomenys);  
 isvesti\_zaidejus(zaidejai, rezultatai);  
  
 return 0;  
}

Python sprendimas:

def skaiciuoti\_ts(pts, fga, fta):

vardiklis = fga + 0.44 \* fta

if vardiklis == 0:

return 0 # Išvengiame dalybos iš nulio

return pts / (2 \* vardiklis)

def nuskaityti\_zaidejus(failo\_pavadinimas):

zaidejai = []

with open(failo\_pavadinimas, 'r') as failas:

eilutes = failas.readlines()

for eilute in eilutes:

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

vardas, komanda, pts, fga, fta = duomenys[0], duomenys[1], int(duomenys[2]), int(duomenys[3]), int(duomenys[4])

zaidejas = {

"vardas": vardas,

"komanda": komanda,

"pts": pts,

"fga": fga,

"fta": fta,

"ts": skaiciuoti\_ts(pts, fga, fta)

}

zaidejai.append(zaidejas)

return zaidejai

def isvesti\_zaidejus(zaidejai, failo\_pavadinimas):

zaidejai.sort(key=lambda p: (p["komanda"], -p["ts"]))

with open(failo\_pavadinimas, 'w') as failas:

for zaidejas in zaidejai:

failas.write(

f"{zaidejas['vardas']},{zaidejas['komanda']},{zaidejas['pts']},"

f"{zaidejas['fga']},{zaidejas['fta']},{zaidejas['ts']:.2f}\n"

)

duomenys = 'duomenys22.csv'

rezultatai = 'rezultatai.csv'

zaidejai = nuskaityti\_zaidejus(duomenys)

isvesti\_zaidejus(zaidejai, rezultatai)

Python sprendimas (variantas, kuriame panaudojamos **klasės**):

class Zaidejas:

def \_\_init\_\_(self, vardas, komanda, pts, fga, fta):

self.vardas = vardas

self.komanda = komanda

self.pts = int(pts)

self.fga = int(fga)

self.fta = int(fta)

self.ts = self.skaiciuoti\_ts()

def skaiciuoti\_ts(self):

if (self.fga + 0.44 \* self.fta) == 0:

return 0 # Išvengiame dalybos iš nulio

return self.pts / (2 \* (self.fga + 0.44 \* self.fta))

def nuskaityti\_zaidejus(failo\_pavadinimas):

zaidejai = []

with open(failo\_pavadinimas, 'r') as failas:

eilutes = failas.readlines()

for eilute in eilutes:

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

vardas, komanda, pts, fga, fta = duomenys[0], duomenys[1], duomenys[2], duomenys[3], duomenys[4]

zaidejas = Zaidejas(vardas, komanda, pts, fga, fta)

zaidejai.append(zaidejas)

return zaidejai

def isvesti\_zaidejus(zaidejai, failo\_pavadinimas):

zaidejai.sort(key=lambda p: (p.komanda, -p.ts))

with open(failo\_pavadinimas, 'w') as failas:

for zaidejas in zaidejai:

failas.write(f"{zaidejas.vardas},{zaidejas.komanda},{zaidejas.pts},{zaidejas.fga},{zaidejas.fta},{zaidejas.ts:.2f}\n")

duomenys = 'duomenys.csv'

rezultatai = 'rezultatai.csv'

zaidejai = nuskaityti\_zaidejus(duomenys)

isvesti\_zaidejus(zaidejai, rezultatai)

#### Taškų skaičius

8

#### Sudėtingumo lygis

4

#### Ugdomos kompetencijos

Pažinimo

### 2.2 užduotis

CSV formato faile (**duomenys.csv**) yra įvykusių rungtynių statistikos sąrašas: žaidėjo vardas, komanda, pelnytų taškų skaičius (PTS), žaidimo metu atliktų metimų kiekis (FGA) ir mestų baudų kiekis (FTA). Tikrasis žaidėjo taiklumas (TS) apskaičiuojamas pagal formulę: TS = PTS / (2 \* (FGA + 0.44 \* FTA)). Parašykite programą, kuri nuskaitytų duomenis, kiekvienam žaidėjui suskaičiuotų tikrojo taiklumo parametrą, iš sąrašo (masyvo) pašalintų žaidėjus, kurių TS nesiekia 0.6 bei išvestų galutinį sąrašą csv formatu į failą **rezultatai.csv** (formatas: žaidėjas, komanda, PTS, FGA, FTA, TS).

duomenys.csv:

LeBron James,Los Angeles Lakers,25,18,7  
Klay Thompson,Golden State Warriors,18,14,2  
Austin Reaves,Los Angeles Lakers,15,10,5  
D'Angelo Russell,Los Angeles Lakers,17,12,4  
Rui Hachimura,Los Angeles Lakers,12,8,3  
Stephen Curry,Golden State Warriors,30,20,8  
Anthony Davis,Los Angeles Lakers,22,16,6  
Draymond Green,Golden State Warriors,8,6,3  
Andrew Wiggins,Golden State Warriors,16,11,4  
Chris Paul,Golden State Warriors,9,7,2

rezultatai.csv:

Klay Thompson,Golden State Warriors,18,14,2,0.60  
Austin Reaves,Los Angeles Lakers,15,10,5,0.61  
D'Angelo Russell,Los Angeles Lakers,17,12,4,0.62  
Rui Hachimura,Los Angeles Lakers,12,8,3,0.64  
Stephen Curry,Golden State Warriors,30,20,8,0.64  
Andrew Wiggins,Golden State Warriors,16,11,4,0.63

#### Vertinimas

| Nr. | Kas vertinama | Balai |
| --- | --- | --- |
| 1. | Žaidėjui aprašyti sudaryta tinkama struktūra | 1 |
| 2. | Duomenų nuskaitymas į masyvą arba sąrašą | 2 |
| 3. | Tikrojo taiklumo skaičiavimas ir priskyrimas struktūrai | 2 |
| 4. | Žaidėjų šalinimas pagal tikrojo taiklumo slenkstį | 2 |
| 5. | Rezultatų išvedimas į failą | 1 |
|  | Viso: | 8 |

#### Teisingas atsakymas

C++ sprendimas:

#include <iostream>  
#include <fstream>  
#include <sstream>  
#include <string>  
#include <vector>  
#include <algorithm>  
#include <iomanip>  
using namespace std;  
  
struct Zaidejas {  
 string vardas;  
 string komanda;  
 int pts;  
 int fga;  
 int fta;  
 double ts;  
};  
  
double skaiciuoti\_ts(int pts, int fga, int fta) {  
 if (fga + 0.44 \* fta == 0) {  
 return 0; // Išvengiame dalybos iš nulio  
 }  
 return (double)(pts) / (2 \* (fga + 0.44 \* fta));  
}  
  
vector<Zaidejas> nuskaityti\_zaidejus(const string& failo\_pavadinimas) {  
 vector<Zaidejas> zaidejai;  
 ifstream failas(failo\_pavadinimas);  
  
 if (!failas.is\_open()) {  
 cerr << "Nepavyko atidaryti failo: " << failo\_pavadinimas << endl;  
 return zaidejai;  
 }  
  
 string eilute;  
 while (getline(failas, eilute)) {  
 stringstream ss(eilute);  
 string vardas, komanda, pts\_str, fga\_str, fta\_str;  
 getline(ss, vardas, ',');  
 getline(ss, komanda, ',');  
 getline(ss, pts\_str, ',');  
 getline(ss, fga\_str, ',');  
 getline(ss, fta\_str, ',');  
  
 int pts = stoi(pts\_str);  
 int fga = stoi(fga\_str);  
 int fta = stoi(fta\_str);  
  
 Zaidejas zaidejas;  
 zaidejas.vardas = vardas;  
 zaidejas.komanda = komanda;  
 zaidejas.pts = pts;  
 zaidejas.fga = fga;  
 zaidejas.fta = fta;  
 zaidejas.ts = skaiciuoti\_ts(pts, fga, fta);  
 zaidejai.push\_back(zaidejas);  
 }  
  
 failas.close();  
 return zaidejai;  
}  
  
void pasalinti\_zaidejus(vector<Zaidejas>& zaidejai, double slenkstis) {  
 for (int i = zaidejai.size() - 1; i >= 0; --i) {  
 if (zaidejai[i].ts < slenkstis) {  
 zaidejai.erase(zaidejai.begin() + i);  
 }  
 }  
}  
  
void isvesti\_zaidejus(vector<Zaidejas>& zaidejai, const string& failo\_pavadinimas) {  
 ofstream failas(failo\_pavadinimas);  
 if (!failas.is\_open()) {  
 cerr << "Nepavyko sukurti failo: " << failo\_pavadinimas << endl;  
 return;  
 }  
  
 for (int i = 0; i < zaidejai.size(); ++i) {  
 failas << zaidejai[i].vardas << ","  
 << zaidejai[i].komanda << ","  
 << zaidejai[i].pts << ","  
 << zaidejai[i].fga << ","  
 << zaidejai[i].fta << ","  
 << fixed << setprecision(2) << zaidejai[i].ts << "\n";  
 }  
 failas.close();  
}

int main() {  
 string duomenys = "duomenys.csv";  
 string rezultatai = "rezultatai.csv";  
 vector<Zaidejas> zaidejai = nuskaityti\_zaidejus(duomenys);  
 pasalinti\_zaidejus(zaidejai, 0.6);  
 isvesti\_zaidejus(zaidejai, rezultatai);  
  
 return 0;  
}

Python sprendimas (1 var.)

import csv

def skaiciuoti\_ts(pts, fga, fta):

vardiklis = fga + 0.44 \* fta

if vardiklis == 0: return 0

return pts / (2 \* vardiklis)

def nuskaityti\_zaidejus(failo\_pavadinimas):

zaidejai = []

with open(failo\_pavadinimas, 'r') as failas:

csv\_duomenys = csv.reader(failas)

for eilute in csv\_duomenys:

vardas, komanda, pts, fga, fta = eilute

zaidejas = {

"vardas": vardas,

"komanda": komanda,

"pts": int(pts),

"fga": int(fga),

"fta": int(fta),

"ts": skaiciuoti\_ts(int(pts), int(fga), int(fta))

}

zaidejai.append(zaidejas)

return zaidejai

def pasalinti\_zaidejus(zaidejai, slenkstis):

return [zaidejas for zaidejas in zaidejai if zaidejas["ts"] >= slenkstis]

def isvesti\_zaidejus(zaidejai, failo\_pavadinimas):

with open(failo\_pavadinimas, 'w', newline='') as failas:

csv\_failas = csv.writer(failas)

for zaidejas in zaidejai:

csv\_failas.writerow([zaidejas["vardas"], zaidejas["komanda"], zaidejas["pts"],

zaidejas["fga"], zaidejas["fta"], f"{zaidejas['ts']:.2f}"])

# Pagrindinė programa

duomenys = 'duomenys.csv'

rezultatai = 'rezultatai.csv'

zaidejai = nuskaityti\_zaidejus(duomenys)

zaidejai = pasalinti\_zaidejus(zaidejai, 0.6)

isvesti\_zaidejus(zaidejai, rezultatai)

Python sprendimas (panaudojant **klases**):

import csv  
class Zaidejas:  
 def \_\_init\_\_(self, vardas, komanda, pts, fga, fta):  
 self.vardas = vardas  
 self.komanda = komanda  
 self.pts = int(pts)  
 self.fga = int(fga)  
 self.fta = int(fta)  
 self.ts = self.skaiciuoti\_ts()  
  
 def skaiciuoti\_ts(self):  
 if (self.fga + 0.44 \* self.fta) == 0:  
 return 0 # Išvengiame dalybos iš nulio  
 return self.pts / (2 \* (self.fga + 0.44 \* self.fta))  
  
def nuskaityti\_zaidejus(failo\_pavadinimas):  
 zaidejai = []  
 with open(failo\_pavadinimas, 'r') as failas:  
 csv\_duomenys = csv.reader(failas)  
 for eilute in csv\_duomenys:  
 vardas, komanda, pts, fga, fta = eilute  
 zaidejas = Zaidejas(vardas, komanda, pts, fga, fta)  
 zaidejai.append(zaidejas)  
 return zaidejai  
  
def pasalinti\_zaidejus(zaidejai, slenkstis):  
 zaidejai[:] = [zaidejas for zaidejas in zaidejai if zaidejas.ts >= slenkstis]  
  
def isvesti\_zaidejus(zaidejai, failo\_pavadinimas):  
 with open(failo\_pavadinimas, 'w', newline='') as failas:  
 csv\_failas = csv.writer(failas)  
 for zaidejas in zaidejai:  
 csv\_failas.writerow([zaidejas.vardas, zaidejas.komanda, zaidejas.pts,  
 zaidejas.fga, zaidejas.fta, f"{zaidejas.ts:.2f}"])  
  
duomenys = 'duomenys.csv'  
rezultatai = 'rezultatai.csv'  
zaidejai = nuskaityti\_zaidejus(duomenys)  
pasalinti\_zaidejus(zaidejai, 0.6)  
isvesti\_zaidejus(zaidejai, rezultatai)

#### Taškų skaičius

8

#### Sudėtingumo lygis

3

#### Ugdomos kompetencijos

Pažinimo

### 2.3 užduotis

CSV formato faile (**duomenys.csv**) yra įvykusių rungtynių statistikos sąrašas: žaidėjo vardas, komanda, pelnytų taškų skaičius (PTS), žaidimo metu atliktų metimų kiekis (FGA) ir mestų baudų kiekis (FTA). Tikrasis žaidėjo taiklumas (TS) apskaičiuojamas pagal formulę: TS = PTS / (2 \* (FGA + 0.44 \* FTA)). Parašykite programą, kuri:

* nuskaitytų duomenis,
* kiekvienam žaidėjui suskaičiuotų tikrojo taiklumo parametrą,
* sukurtų du naujus sąrašus (po vieną rungtynių komandai),
* juos surikiuotų tikrojo taiklumo mažėjimo tvarka
* rezultatą išvestų į csv tipo failus **sarasas1.csv** ir **sarasas2.csv** (formatas: žaidėjas, komanda, PTS, FGA, FTA, TS).

duomenys.csv:

LeBron James,Los Angeles Lakers,25,18,7  
Klay Thompson,Golden State Warriors,18,14,2  
Austin Reaves,Los Angeles Lakers,15,10,5  
D'Angelo Russell,Los Angeles Lakers,17,12,4  
Rui Hachimura,Los Angeles Lakers,12,8,3  
Stephen Curry,Golden State Warriors,30,20,8  
Anthony Davis,Los Angeles Lakers,22,16,6  
Draymond Green,Golden State Warriors,8,6,3  
Andrew Wiggins,Golden State Warriors,16,11,4  
Chris Paul,Golden State Warriors,9,7,2

sarasas1.csv:

Rui Hachimura,Los Angeles Lakers,12,8,3,0.64  
D'Angelo Russell,Los Angeles Lakers,17,12,4,0.62  
Austin Reaves,Los Angeles Lakers,15,10,5,0.61  
LeBron James,Los Angeles Lakers,25,18,7,0.59  
Anthony Davis,Los Angeles Lakers,22,16,6,0.59

sarasas2.csv:

Stephen Curry,Golden State Warriors,30,20,8,0.64  
Andrew Wiggins,Golden State Warriors,16,11,4,0.63  
Klay Thompson,Golden State Warriors,18,14,2,0.60  
Chris Paul,Golden State Warriors,9,7,2,0.57  
Draymond Green,Golden State Warriors,8,6,3,0.55

#### Vertinimas

| Nr. | Kas vertinama | Balai |
| --- | --- | --- |
| 1. | Žaidėjui aprašyti sudaryta tinkama struktūra | 1 |
| 2. | Duomenų nuskaitymas į masyvą arba sąrašą | 2 |
| 3. | Tikrojo taiklumo skaičiavimas ir priskyrimas struktūrai | 2 |
| 4. | Atskirų komandų sąrašų sudarymas | 2 |
| 4. | Rūšiavimas | 2 |
| 5. | Rezultatų išvedimas į failą | 1 |
|  | Viso: | 10 |

#### Teisingas atsakymas

C++ sprendimas:

#include <iostream>  
#include <fstream>  
#include <sstream>  
#include <string>  
#include <vector>  
#include <algorithm>  
#include <iomanip>  
using namespace std;  
  
struct Zaidejas {  
 string vardas;  
 string komanda;  
 int pts;  
 int fga;  
 int fta;  
 double ts;  
};  
  
double skaiciuoti\_ts(int pts, int fga, int fta) {  
 if (fga + 0.44 \* fta == 0) {  
 return 0;  
 }  
 return (double)(pts) / (2 \* (fga + 0.44 \* fta));  
}  
  
vector<Zaidejas> nuskaityti\_zaidejus(const string& ivesties\_failas) {  
 vector<Zaidejas> zaidejai;  
 ifstream failas(ivesties\_failas);  
  
 if (!failas.is\_open()) {  
 cerr << "Nepavyko atidaryti failo: " << ivesties\_failas << endl;  
 return zaidejai;  
 }  
  
 string eilute;  
 while (getline(failas, eilute)) {  
 stringstream ss(eilute);  
 string vardas, komanda, pts\_str, fga\_str, fta\_str;  
  
 getline(ss, vardas, ',');  
 getline(ss, komanda, ',');  
 getline(ss, pts\_str, ',');  
 getline(ss, fga\_str, ',');  
 getline(ss, fta\_str, ',');  
  
 int pts = stoi(pts\_str);  
 int fga = stoi(fga\_str);  
 int fta = stoi(fta\_str);  
  
 Zaidejas zaidejas;  
 zaidejas.vardas = vardas;  
 zaidejas.komanda = komanda;  
 zaidejas.pts = pts;  
 zaidejas.fga = fga;  
 zaidejas.fta = fta;  
 zaidejas.ts = skaiciuoti\_ts(pts, fga, fta);  
  
 zaidejai.push\_back(zaidejas);  
 }  
  
 failas.close();  
 return zaidejai;  
}  
  
void apdoroti\_komandas(const vector<Zaidejas>& zaidejai, vector<Zaidejas>& komanda1, vector<Zaidejas>& komanda2) {  
 string pirmoji\_komanda, antroji\_komanda;  
   
 for (size\_t i = 0; i < zaidejai.size(); ++i) {  
 const Zaidejas& zaidejas = zaidejai[i];  
   
 if (pirmoji\_komanda.empty()) {  
 pirmoji\_komanda = zaidejas.komanda;  
 } else if (antroji\_komanda.empty() && zaidejas.komanda != pirmoji\_komanda) {  
 antroji\_komanda = zaidejas.komanda;  
 }  
   
 if (zaidejas.komanda == pirmoji\_komanda) {  
 komanda1.push\_back(zaidejas);  
 } else {  
 komanda2.push\_back(zaidejas);  
 }  
 }  
  
 sort(komanda1.begin(), komanda1.end(), [](const Zaidejas& a, const Zaidejas& b) {  
 return a.ts > b.ts;  
 });  
 sort(komanda2.begin(), komanda2.end(), [](const Zaidejas& a, const Zaidejas& b) {  
 return a.ts > b.ts;  
 });  
}  
  
void isvesti\_i\_csv(const vector<Zaidejas>& komanda, const string& isvesties\_failas) {  
 ofstream failas(isvesties\_failas);  
 if (!failas.is\_open()) {  
 cerr << "Nepavyko sukurti failo: " << isvesties\_failas << endl;  
 return;  
 }  
  
 for (int i = 0; i < komanda.size(); ++i) {  
 const Zaidejas& zaidejas = komanda[i];  
 failas << zaidejas.vardas << ","  
 << zaidejas.komanda << ","  
 << zaidejas.pts << ","  
 << zaidejas.fga << ","  
 << zaidejas.fta << ","  
 << fixed << setprecision(2) << zaidejas.ts << "\n";  
 }  
 failas.close();  
}  
  
int main() {  
 string ivesties\_failas = "duomenys.txt";  
 vector<Zaidejas> zaidejai = nuskaityti\_zaidejus(ivesties\_failas);  
 vector<Zaidejas> komanda1, komanda2;  
  
 apdoroti\_komandas(zaidejai, komanda1, komanda2);  
 isvesti\_i\_csv(komanda1, "sarasas1.txt");  
 isvesti\_i\_csv(komanda2, "sarasas2.txt");  
  
 return 0;  
}

Python sprendimas (1 var.)

def skaiciuoti\_ts(pts, fga, fta):

vardiklis = fga + 0.44 \* fta

if vardiklis == 0:

return 0 # Išvengiame dalybos iš nulio

return pts / (2 \* vardiklis)

def nuskaityti\_zaidejus(ivesties\_failas):

zaidejai = []

with open(ivesties\_failas, 'r') as failas:

for eilute in failas:

vardas, komanda, pts, fga, fta = eilute.strip().split(',')

zaidejas = {

"vardas": vardas,

"komanda": komanda,

"pts": int(pts),

"fga": int(fga),

"fta": int(fta),

"ts": skaiciuoti\_ts(int(pts), int(fga), int(fta))

}

zaidejai.append(zaidejas)

return zaidejai

def apdoroti\_komandas(zaidejai):

komanda1, komanda2 = [], []

komandos = set()

for zaidejas in zaidejai:

if len(komandos) < 2:

komandos.add(zaidejas["komanda"])

if zaidejas["komanda"] == list(komandos)[0]:

komanda1.append(zaidejas)

else:

komanda2.append(zaidejas)

komanda1.sort(key=lambda z: z["ts"], reverse=True)

komanda2.sort(key=lambda z: z["ts"], reverse=True)

return komanda1, komanda2

def isvesti\_i\_csv(komanda, isvesties\_failas):

with open(isvesties\_failas, 'w') as failas:

for zaidejas in komanda:

failas.write(f"{zaidejas['vardas']},{zaidejas['komanda']},{zaidejas['pts']},"

f"{zaidejas['fga']},{zaidejas['fta']},{zaidejas['ts']:.2f}\n")

# Pagrindinė programa

ivesties\_failas = "duomenys.csv"

zaidejai = nuskaityti\_zaidejus(ivesties\_failas)

komanda1, komanda2 = apdoroti\_komandas(zaidejai)

isvesti\_i\_csv(komanda1, "sarasas1.csv")

isvesti\_i\_csv(komanda2, "sarasas2.csv")

Python sprendimas (panaudojant **klases**):

class Zaidejas:  
 def \_\_init\_\_(self, vardas, komanda, pts, fga, fta):  
 self.vardas = vardas  
 self.komanda = komanda  
 self.pts = int(pts)  
 self.fga = int(fga)  
 self.fta = int(fta)  
 self.ts = 0.0  
 self.skaiciuoti\_ts()  
  
 def skaiciuoti\_ts(self):  
 if self.fga + 0.44 \* self.fta == 0:  
 self.ts = 0  
 else:  
 self.ts = self.pts / (2 \* (self.fga + 0.44 \* self.fta))  
  
def nuskaityti\_zaidejus(ivesties\_failas):  
 zaidejai = []  
 with open(ivesties\_failas, 'r') as failas:  
 for eilute in failas:  
 vardas, komanda, pts, fga, fta = eilute.strip().split(',')  
 zaidejas = Zaidejas(vardas, komanda, pts, fga, fta)  
 zaidejai.append(zaidejas)  
 return zaidejai  
  
def apdoroti\_komandas(zaidejai):  
 komanda1, komanda2 = [], []  
 komandos = set()  
  
 for zaidejas in zaidejai:  
 if len(komandos) < 2:  
 komandos.add(zaidejas.komanda)  
 if zaidejas.komanda == list(komandos)[0]:  
 komanda1.append(zaidejas)  
 else:  
 komanda2.append(zaidejas)  
  
 komanda1.sort(key=lambda z: z.ts, reverse=True)  
 komanda2.sort(key=lambda z: z.ts, reverse=True)  
 return komanda1, komanda2  
  
def isvesti\_i\_csv(komanda, isvesties\_failas):  
 with open(isvesties\_failas, 'w') as failas:  
 for zaidejas in komanda:  
 failas.write(f"{zaidejas.vardas},{zaidejas.komanda},{zaidejas.pts},{zaidejas.fga},{zaidejas.fta},{zaidejas.ts:.2f}\n")  
  
ivesties\_failas = "duomenys.csv"  
zaidejai = nuskaityti\_zaidejus(ivesties\_failas)  
komanda1, komanda2 = apdoroti\_komandas(zaidejai)  
isvesti\_i\_csv(komanda1, "sarasas1.csv")  
isvesti\_i\_csv(komanda2, "sarasas2.csv")

#### Taškų skaičius

10

#### Sudėtingumo lygis

4

#### Ugdomos kompetencijos

Pažinimo