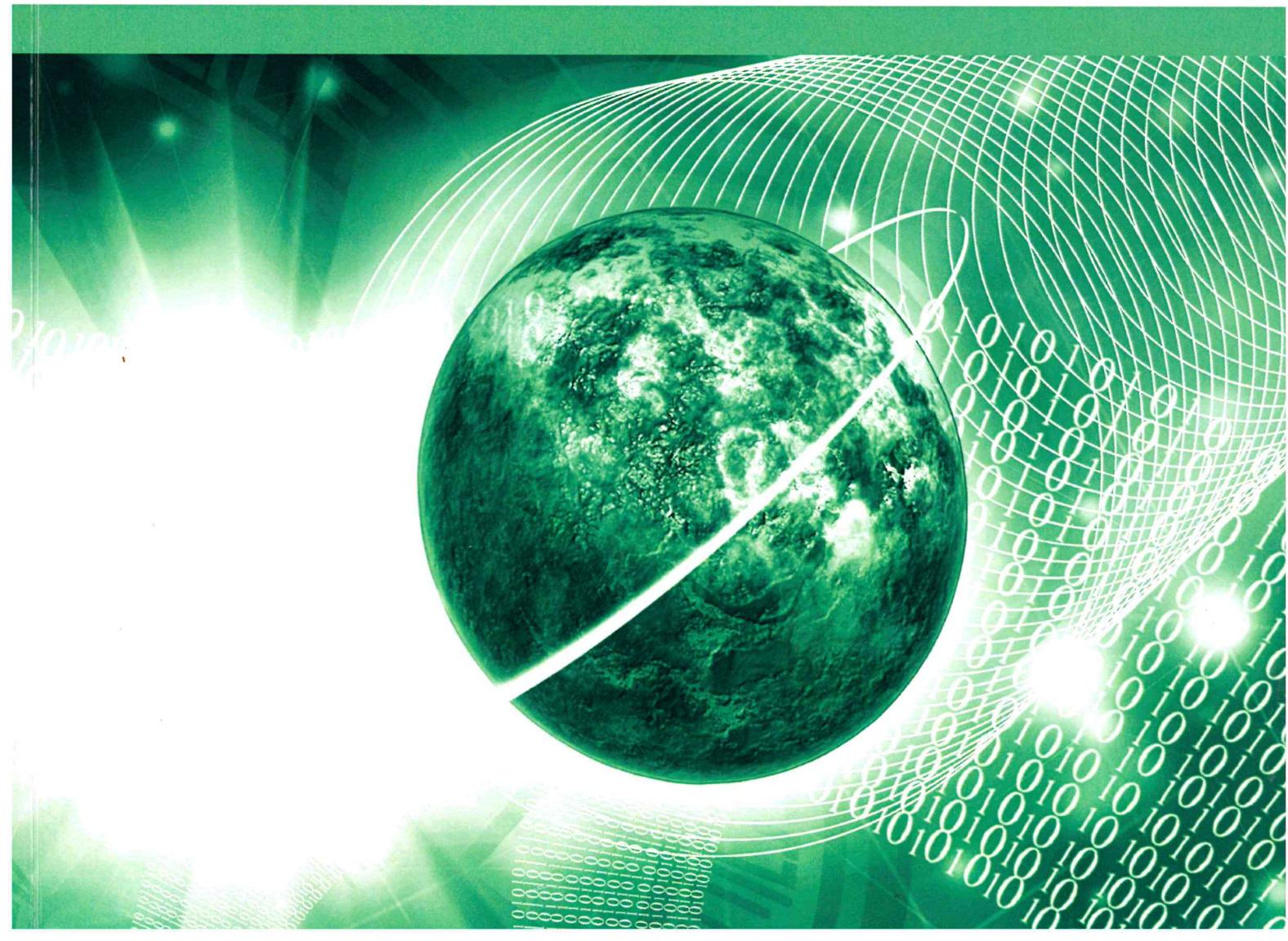


Jonas Blonskis  
Vytautas Bukšnaitis  
Vacius Jusas  
Romas Marcinkevičius  
Tomas Stonys

# PROGRAMAVIMO ĮVADAS



## Turinys

Pratarmė.....	4
1. Programavimo elementai .....	5
1.1. Ciklo sakiny.....	5
1.2. Sąlygos sakiny .....	7
1.3. Matematinis reiškinys .....	8
1.4. Sudėtingesnis matematinis reiškinys.....	9
1.5. Funkcijos .....	10
1.6. Simboliai ir simbolų eilutės .....	12
1.7. Kontroliniai klausimai.....	15
2. Klasė .....	18
2.1. Viena klasė.....	18
2.2. Dvi klasės.....	23
2.3. Kontroliniai klausimai.....	27
2.4. Užduotys .....	28
3. Objektų rinkinys .....	33
3.1. Pažintis su objektų rinkiniu.....	33
3.2. Du objektų rinkiniai .....	36
3.3. Naujo objektų rinkinio formavimas .....	38
3.4. Keletas pavadinimo žodžių .....	43
3.5. Objektų charakteristikų radimas .....	45
3.6. Kontroliniai klausimai.....	47
3.7. Užduotys .....	48
4. Konteinerinė klasė .....	53
4.1. Konteinerinės klasės sąsajos metodai .....	53
4.2. Operatorių užklojimas konteineryje .....	57
4.3. Kontroliniai klausimai.....	61
4.4. Užduotys .....	62
5. Teksto analizė ir redagavimas .....	67
5.1. Eilutės simbolų analizė .....	67
5.2. Teksto eilučių šalinimas.....	69
5.3. Teksto eilučių dalių šalinimas .....	71
5.4. Žodžių išskyrimas eilutėje.....	72
5.5. Eilučių redagavimas .....	74
5.6. Analizės failo sukūrimas .....	76
5.7. Kontroliniai klausimai.....	79
5.8. Užduotys .....	80
6. Susieti rinkiniai.....	83
6.1. Veiksmai su sveikujų skaičių dvimačiu masyvu.....	83
6.2. Veiksmai su sveikujų skaičių dvimačio masyvo eilutėmis ir stulpeliais .....	86
6.3. Dvimatis objektų masyvas .....	87
6.4. Veiksmai su objektais dvimačio masyvo eilutėse ir stulpeliuose .....	91
6.5. Objektų masyvas ir dvimatis sveikujų skaičių masyvas .....	93
6.6. Kontroliniai klausimai.....	100
6.7. Užduotys .....	101
7. Priedai.....	105

## Pratarmė

Išmokti programuoti galima tik rašant programas. Kiekviena didesnės apimties teorinė medžiaga yra išsisavinama parašant programą pagal individualią užduotį\*. Tačiau prieš tai tikslingo praktiškai patikrinti paskaitose pateiktą medžiagą. Šis pratimų rinkinys tam ir yra skirtas. Pratimai suskirstyti į 5 skyrius, kuriuose yra siejamas su studento individualia užduotimi. Studentui siūloma atlkti visus pateiktus pratimus, išitikinti, kad pakanka teorinių ir praktinių žinių, reikalingų individualiai užduočiai atlkti. Tam yra paruošiamos apklausos testai.

- Susipažinsite su baziniais ir struktūriniais duomenų tipais, vartotojo duomenų tipų kūrimo elementais.
- Rasite nesudėtingus algoritmus ir jų taikymą darbui su duomenimis konkrečiose duomenų struktūrose:
- sumos, sandaugos, kieko, aritmetinio vidurkio skaičiavimas;
- didžiausios (mažiausios) reikšmės paieška;
- duomenų rikiavimas;
- duomenų apdorojimo veiksmai: šalinimas, išterpimas, naujų rinkinių formavimas, duomenų filtracija;
- duomenų skaitymas iš failo, surašymas į failą;

Susipažinsite, kaip algoritmai taikomi darbui su duomenimis skirtingose duomenų struktūrose, kaip jie modifikuojami.

Programų pavyzdžiai pateikiami C++ programavimo kalba. Rasite naudojamus pavyzdžiuose programavimo kalbos elementus ir kaip jie taikomi konkretiems tikslams.

Susipažinsite su objektinio programavimo priemonėmis – klasėmis ir objektais.

Susipažinsite su programų rašymu žingsniais ir derinimu konkrečioje programavimo aplinkoje.

Susipažinsite su programų išbandymui reikalingų duomenų paruošimu.

Susipažinsite, kaip minimaliai dokumentuojama programa.

Pateikti pratimai sugrupuoti pagal naudojamas duomenų struktūras ir naudojamus algoritmus sudėtingumo seka.

Prieš pradedant dirbti reikia susipažinti su programavimo aplinka. Darbo su kai kuriomis aplinkomis susipažinimo scenarijai pateikiami prieduose.

Skyrelio „Programavimo elementai“ pratimų rinkinys skirtas pažinčiai su programavimo kalba, klase ir objektu. Sudėtingesnės programavimo kalbos priemonės pateikiamos kituose skyreliuose, kai tap atsiranda poreikis. Detaliau galima susipažinti vadovelyje ir programavimo kalbos mokomojoje knygoje. Skyrelio „Klasė“ pratimai supažindina su klase, kaip duomenų tipu ir objektu, kaip tos klasės kintamuoju.

Skyrelio „Objektų rinkinys“ pratimų rinkinys skirtas pažinčiai su objektų rinkiniu, duomenų srautais išvedimui bei spausdinimui, nesudėtingais veiksmai.

Skyrelio „Konteinerinė klasė“ pratimų rinkinys skirtas pažinčiai su objektų konteineriu (klase, skirta saugoti ir atlkti veiksmus su objektų rinkiniu).

Skyrelio „Teksto analizė ir redagavimas“ pratimų rinkinys skirtas pažinčiai su teksto analize ir redagavimu.

Skyrelio „Susieti rinkiniai“ pratimų rinkinys skirtas pažinčiai su dvimačių objektų rinkiniu.

## 1. Programavimo elementai

Susipažinsite su programavimo kalbos C++ pagrindinėmis konstrukcijomis:

- pagrindine funkcija `main()`;
- baziniai duomenų tipais (`int, double, bool, char`);
- duomenų spausdinimu ekrane;
- duomenų išvedimu klaviatūra;
- sąlygos sakiniai;
- ciklo sakiniai;
- funkcijomis, kaip programos struktūrizavimo priemone;
- simbolinių eilutė `string`.

### 1.1. Ciklo sakiny

- Ciklo sakiny `for`.
- `main()` funkcija.
- Duomenų spausdinimas ekrane.
- `int` tipo kintamieji.

#### 1 Pirmas žingsnis.

- Sukurkite projektą `main`, kurį išsaugosite D diske. I projektą iutraukite `.cpp` failą, kurį pavadinkite `main`.
- Iveskite programos, išvedančios į ekraną skaičius nuo 1 iki 10 ir jų kvadratus, tekštą:

```
// -----
// Ciklas for
#include <iostream>
using namespace std;
// -----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    cout << " Skaičiai nuo 1 iki 10 ir juų kvadratai: " << endl;
    for (int i = 1; i < 11; i++)
        cout << i << " " << i * i << endl;
    cout << endl;
    return 0;
}
// -----
```

- Suskaičiuokite laukiamus rezultatus.
- Kompiliuokite ir įvykdykite programą.
- Patirkinkite, ar gauti rezultatai atitinka laukiamus rezultatus.

#### 2 Antras žingsnis.

- Analizuodami programos `main.cpp` tekštą nustatykite, kuri programos vieta nusako skaičių intervalo kitimą. Šis sakiny vadinas ciklo sakiniu.
- Pakeiskite programos tekštą taip, kad būtų išvedami skaičiai nuo 5 iki 15 ir jų kvadratai.
- Suskaičiuokite laukiamus rezultatus.
- Įvykdykite pakeistą programą.
- Patirkinkite, ar gauti rezultatai atitinka laukiamus rezultatus.

#### 3 Trečias žingsnis.

- Pakeiskite programos tekštą taip, kad būtų išvedami skaičiai nuo 5 iki 15, jų kvadratai ir jų kubai.
- Suskaičiuokite laukiamus rezultatus.

\* nesudėtinga projektinė užduotis arba sudėtingesnė komandiniam darbui skirta projektinė užduotis, rekomenduojama jau turintiems pradinę sampratą apie programavimą.

- Įvykdykite pakeistą programą.
- Patikrinkite, ar gauti rezultatai atitinka laukiamus rezultatus.

#### Q Ketvirtas žingsnis.

- Susipažindami su programavimo aplinka (žr. Priedas 1), panaudojote kintamuosius *a* ir *b* su pradinėmis reikšmėmis tokiaiame pavyzdje:

```
//-----
// Dviejų kintamųjų reikšmių sumos radimas
#include <iostream>
using namespace std;
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    int a = 3;           // pirmasis kintamasis
    int b = 8;           // antrasis kintamasis
    int suma = a + b;    // skaičių sumai saugoti
    cout << a << " + " << b << " = ";
    cout << suma;
    cout << endl;
    return 0;
}
```

- Remdamiesi pavyzdžiu, programoje *main.cpp* paskelbkite du kintamuosius, nurodančius ciklo kitimo režius. Panaudokite šiuos kintamuosius ciklo sakinyje.
- Suskaičiuokite laukiamus rezultatus.
- Įvykdykite pakeistą programą ir patikrinkite, ar gauti rezultatai atitinka laukiamus rezultatus.

#### Q Penktas žingsnis.

- Į reikiama programos vietą įterpkite eilutę, kuri visų rezultatų pabaigoje išvestų, kiek kartų buvo skaičiuota. Atspausdinkite ir paaiškinamajį tekštą.
- Įvykdykite pakeistą programą ir patikrinkite rezultatus.

#### Q Šeistas žingsnis.

- Modifikuokite programą taip, kad ciklo režių reikšmės būtų įvedamos dialogu. Remkitės pavyzdžiu:

```
//-----
// Įvedamų kintamųjų reikšmių sumos radimas
#include <iostream>
using namespace std;
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    int a;           // pirmasis kintamasis
    int b;           // antrasis kintamasis
    int suma;        // kintamasis skaičių sumai saugoti
    cout << "Įveskite a, b reikšmes:" << endl;
    cin >> a >> b;
    suma = a + b;
    cout << "a + b = " << suma << endl;
    return 0;
}
```

Įvedimo lange spausdinkite paaiškinamajį tekštą, iš kurio būtų aišku, ką reikia įvesti.

- Įvykdykite pakeistą programą ir patikrinkite rezultatus.

## 1.2. Salygos sakiny

- Salygos sakiny *if*.

#### Q Pirmas žingsnis.

- Sukurkite projektą *SalygosSakinys*.
- Įveskite programos, kuri spausdina žvaigždutes, tekštą:

```
//-----
// Salygos sakiny
// Simbolių paskirstymas į eilutes
#include <iostream>
using namespace std;
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    for (int i = 1; i < 51; i++) {
        cout << '*';
        if (i % 5 == 0)
            cout << endl;
    }
    cout << endl;
    return 0;
}
```

- Atkreipkite dėmesį, kad patikrinimo salyga „ar lygu” užrašoma dviem simboliais *==*.
- Taip pat atkreipkite dėmesį į riestinius skliaustus, kurie nurodo ciklo viduje esančius sakinius. Jeigu tokį sakinių bloką sudaro tik vienas sakiny, tuomet riestiniai skliaustai nebūtini.
- Apskaičiuokite laukiamus rezultatus, įvykdykite programą ir pasitikrinkite, ar gauti rezultatai sutampa su apskaičiuotais.

#### Q Antras žingsnis.

- Pakeiskite programą taip, kad būtų galima nurodyti spausdinamą simbolį. Taip pat pakeiskite salygos sakinių. Tai leis atsisakyti riestinių skliaustų:

```
//-----
// Salygos sakiny
// Simbolių paskirstymas į eilutes
#include <iostream>
using namespace std;
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    char simbolis;
    cout << "Nurodykite simbolį:" << endl;
    cin >> simbolis;
    for (int i = 1; i < 51; i++)
        if (i % 5 == 0)
            cout << simbolis << endl;
        else
            cout << simbolis;
    cout << endl;
    return 0;
}
```

- Kompiliuokite ir įvykdykite programą. Įveskite jums patinkantį simbolį.
- Patikrinkite, ar gauti rezultatai atitinka laukiamus rezultatus.

**OTrečias žingsnis.**

- Papildykite programą, kad būtų galima nurodyti bendrą spausdinamų simbolių kiekį ir vienos eilutės simbolių kiekį.
- Kompiliuokite ir įvykdykite programą.
- Patikrinkite, ar gauti rezultatai atitinka laukiamus rezultatus.

**Savarankiško darbo užduotis.**

Kai kuriais atvejais tą patį simbolių spausdinimo rezultatą galima pasiekti nesinaudojant sąlygos sakiniu, bet panaudojant vienas iš kitų idėtus 2 ciklo sakinius (ciklą cikle). Parenkite tokią programą. Ciklų kintamieji turi būti skirtini.

**1.3. Matematinis reiškinys**

- Sąlygos sakiny.
- Matematinės funkcijos.

**Užduotis.**

Suskaičiuoti funkcijos reikšmę:  $f(x) = \sqrt{z-1}$

- D diske sukurate projektą.
- Parašykite programą:

```
-----  
// Funkcijos reikšmės skaičiavimas  
#include <iostream>  
#include <cmath> // matematinė funkcių rinkinys  
using namespace std;  
-----  
int main()  
{  
    setlocale(LC_ALL, "Lithuanian");  
    double fx;  
    int z;  
    cout << "Įveskite z reikšmę:" << endl;  
    cin >> z;  
    if (z - 1 >= 0) {  
        fx = pow(z - 1, 0.5);  
        cout << "z = " << z << " f(x) = " << fx << endl;  
    }  
    else  
        cout << "z = " << z << " f-ja neegzistuoja" << endl;  
    return 0;  
}
```

Atkreipkite dėmesį, kad sąlygos sakinyje panaudojome riestinius skliaustus. Jie būtini, nes sąlygos sakiniu šakoje yra daugiau kaip vienas programos sakiny.

- Įvykdykite programą ir patikrinkite rezultatus.

**Užduotis savarankiškam darbui.**

Suskaičiuoti funkcijos reikšmę:  $f(x, y) = \frac{y^2 - 2yx + x^2}{x^3 - y}$ .

**1.4. Sudėtingesnis matematinis reiškinys**

- Sąlygos sakiny.
- Matematinės funkcijos.

**Užduotis.**

Suskaičiuoti funkcijos reikšmę:  $f(x) = \begin{cases} ae^{-x}, & x \leq 0, \\ 5ax - 7, & 0 < x < 1, \\ \sqrt{x+1}. & \end{cases}$

```
-----  
// Sudėtingas if sakiny  
#include <iostream>  
#include <cmath> // Matematinė funkcių rinkinys  
using namespace std;  
-----  
int main()  
{  
    setlocale(LC_ALL, "Lithuanian");  
    double fx;  
    int a;  
    double x; // x negali būti int dėl exp() funkcijos  
    cout << "Įveskite a reikšmę:" << endl;  
    cin >> a;  
    cout << "Įveskite x reikšmę:" << endl;  
    cin >> x;  
    if (x <= 0)  
        fx = a * exp(-x);  
    else  
        if (x < 1)  
            fx = 5 * a * x - 7;  
        else  
            fx = pow(x + 1, 0.5);  
    cout << "Reikšmė a = " << a << ", reikšmė x = " << x  
         << ", fx = " << fx << endl;  
    return 0;  
}
```

- Įvykdykite programą ir patikrinkite rezultatus.

**Savarankiško darbo užduotis.**

Suskaičiuoti funkcijos reikšmę:  $f(x) = \begin{cases} f_1(x), & \text{kai 1 sąlyga} \\ f_2(x), & \text{kai 2 sąlyga} \\ f_3(x), & \text{kitais atvejais} \end{cases}$

Funkcijos variantą pasirinkite pagal kompiuterio numerį. Kompiuterio numerį dalinkite iš 12, ir gauta liekaną rodo varianto numerį.

Nr.	$f_1(x)$	$f_2(x)$	$f_3(x)$	1 sąlyga	2 sąlyga
0.	$e^{-x}$	$2x^2 + 1$	$1/x^2$	$-2 \leq x \leq 0$	$0 < x < 2$
1.	$e^{-x}$	$5x - 7$	$\sqrt{x+1}$	$-3 \leq x \leq 0$	$0 < x \leq 1$
2.	$\cos x$	$1/(x+5)^3$	$\sqrt{x^2 + 1}$	$-4 \leq x \leq 0$	$0 < x \leq 4$
3.	$x+2x^2$	$x^2 + 4$	$\sqrt{2x^2 + 3}$	$-5 \leq x \leq 0$	$0 < x < 3$
4.	$\cos x^2$	$4x^2 + 3$	$\sqrt{x^2 + x + 4}$	$-1 \leq x \leq 0$	$0 < x < 1$
5.	$1/(x+5)$	$x+1$	$\sqrt{x^2 + x + 1}$	$-1 \leq x < 0$	$0 \leq x < 1$
6.	$\sin^2 x$	$(x-1)^2$	$x^2 + x + 1$	$-1 \leq x < 0$	$0 \leq x < 1$
7.	$3.2x^2$	$\sin^2(x+1)$	$3x^2 - 1$	$-2 \leq x < 0$	$0 \leq x \leq 1$

Programavimo išvadas					
8.	$1/(x-5)$	$(x+3)^2$	$\sqrt{x+5}$	$-3 \leq x \leq 0$	$0 < x \leq 5$
9.	$1/x$	$\cos x$	$2x+4$	$-4 \leq x < -2$	$-2 \leq x \leq 2$
10.	$1/x^2$	$\sin(x+1)$	$\sqrt{2x}$	$-5 \leq x < 0$	$0 \leq x \leq 2$
11.	$1/(x-3)$	$x^2+6$	$2x+3$	$-6 \leq x \leq -1$	$-1 < x \leq 3$

Pastaba. cos, sin funkcijų argumentas yra išreikštasis radianais.

#### Savarankiško darbo užduotis.

Sukurkite mini skaičiuotuvą darbui su realiais skaičiais. Skaičiuotuvas atlieka 4 operacijas: sudėtis (+), atimtis (-), daugyba (\*), dalyba (/). Operacijas žymėkite atitinkamais simboliais. Įveskite porą skaičių. Įveskite operacijos simbolį. Galima įvesti bet kokį simbolį. Jei įvestas simbolis neatitinka nei vieno operacijos ženklo, turi būti rodomas pranešimas ERROR (arba KLAIDA). Jei simbolis atitinka kurią nors operaciją, suskaičiuokite bei įveskite visus įvestus duomenis ir operacijos rezultatą. Sąlygos sakinyje operacijos ženklą nurodykite tarp apostrofų (viengubų kabučių).

## 1.5. Funkcijos

- Funkcija, negražinanti reikšmęs.
- Funkcija, grąžinanti reikšmę.

#### ¶Pirmas žingsnis.

Funkcija, negražinanti reikšmęs

- 1.2 pratimo antrojo žingsnio programą pertvarkykite taip, kad skaičiavimai būtų atliekami funkcijoje:

```
-----  
// Funkcija, negražinanti reikšmęs  
#include <iostream>  
using namespace std;  
-----  
void Spausdinti(char ch); // funkcijos prototipas  
-----  
int main()  
{  
    setlocale(LC_ALL, "Lithuanian");  
    char simbolis;  
    cout << "Nurodykite simbolį:" << endl;  
    cin >> simbolis;  
    Spausdinti(simbolis); // kreipinys į funkcija  
    return 0;  
}  
-----  
// Ekrane spausdinamas 51 simbolis po 5 eilutėje  
// ch - parametras reiškmė yra spausdinamas simbolis  
void Spausdinti(char ch)  
{  
    for (int i = 1; i < 51; i++)  
        if (i % 5 == 0)  
            cout << ch << endl;  
        else cout << ch;  
    cout << endl;  
}
```

- Funkcijos parametrai ir jų reikšmės susiejami pagal vietą parametrų sąraše, bet ne pagal vardą: funkcijos antraštėje parametras pavadintas ch, o kreipinyje – simbolis.
- Įvykdykite programą ir patikrinkite rezultatus.

#### ¶Antras žingsnis.

Funkcija, grąžinanti reikšmę

- 1.1 pratimo ketvirtio žingsnio programą pertvarkykite taip, kad skaičiavimus atliktumėte funkcijoje:

```
-----  
// Funkcija, grąžinanti reikšmę  
#include <iostream>
```

#### Programavimo išvadas

```
using namespace std;  
-----  
int Suma(int sk1, int sk2); // funkcijos prototipas  
-----  
int main()  
{  
    setlocale(LC_ALL, "Lithuanian");  
    int a = 3; // pirmasis kintamasis  
    int b = 8; // antrasis kintamasis  
    cout << a << " + " << b << " = ";  
    cout << Suma(a, b); // kreipinys į funkcija  
    cout << endl << endl;  
    return 0;  
}  
-----  
// Gražina dvielę skaičių sk1 ir sk2 sumą  
int Suma(int sk1, int sk2)  
{  
    int suma = sk1 + sk2;  
    return suma;  
}  
-----
```

Funkcijos antraštė su kabliataškiu gale vadinama funkcijos prototipu. Funkcijos prototipas užrašomas virš pagrindinės programos. Funkcijos antraštę sudaro grąžinamos reikšmės tipas, funkcijos vardas ir parametrai. Jei funkcija nieko negražina, grąžinamos reikšmės tipas – void. Jei funkcija gražina reikšmę, funkcijos kameine turi būti bent vienas return sakyns. Funkcijos realizacija užrašoma po pagrindine programa (funkcija main()). Pagrindinėje programoje turi būti kreipinys į funkciją.

- Funkcija gali būti ir tokia:

```
-----  
// Gražina dvielę skaičių sk1 ir sk2 sumą  
int Suma(int sk1, int sk2)  
{  
    return sk1 + sk2;  
}  
-----
```

#### ¶Trečias žingsnis. Funkcijos apibrėžtumas ir universalumas

- 1.3 programą pertvarkykite taip, kad skaičiavimai būtų atliekami funkcijoje:

```
-----  
// Funkcija suteikia universalumo  
#include <iostream>  
#include <cmath> // matematinė funkcių rinkinys  
using namespace std;  
-----  
double FReikšmė(int sk1, int sk2, double laipsnis);  
-----  
int main()  
{  
    setlocale(LC_ALL, "Lithuanian");  
    double fx;  
    int z;  
    cout << "Įveskite z reikšmę:" << endl;  
    cin >> z;  
    if (z - 1 >= 0) {  
        fx = FReikšmė(z, 1, 0.5);  
        cout << "z = " << z << ", f(x) = " << fx << endl;  
    }  
    else  
        cout << "z = " << z << " f-ja neegzistuoja" << endl;  
    return 0;  
}  
-----
```



```
// Gražina argumentų sk1 ir sk2 skirtuma, pakelta laipsniu laipsnis
double FReikšmė(int sk1, int sk2, double laipsnis)
{
    return pow(sk1 - sk2, laipsnis);
}-----
```

Funkcija visada privalo atlikti vieną apibrėžtą veiksmą. I vieną funkciją negalima jungti reikšmės skaičiavimo ir reikšmės spausdinimo. Jei visos naudojamos konstantos perkeliamos į parametru sąrašą, funkcija tampa universalesnė.

#### Savarankiško darbo užduotis.

Pertvarkykite visas anksčiau sukurtas programas taip, kad skaičiavimai būtų atliekami funkcijose.

## 1.6. Simboliai ir simbolių eilutės

- Simbolinis duomenų tipas `char`.
- Simbolių eilutė `string`.

#### 1 Pirmas žingsnis. Simbolinis duomenų tipas `char`.

- Parašykite programą, kuri atspausdintų mažąsias lotyniškos abécélės raides.

```
-----  
// Raidės
#include <iostream>
using namespace std;
-----  
int main()
{
    for (char ch = 'a'; ch <= 'z'; ch++)
        cout << ch;
    cout << endl;
    return 0;
}-----
```

- Ivykdykite programą ir patikrinkite rezultatus.

#### 2 Antras žingsnis. Raidės ir jų kodai.

- Papildykite ankstesnę programą taip, kad būtų spausdinamos raidės ir jų kodai.

```
-----  
// Raidės ir jų kodai
#include <iostream>
using namespace std;
-----  
int main()
{
    for (char ch = 'a'; ch <= 'z'; ch++)
        cout << ch << " - " << (int)ch << endl;
    cout << endl;
    return 0;
}-----
```

- Ivykdykite programą ir patikrinkite rezultatus.

#### 3 Trečias žingsnis. Nurodyto intervalo raidės ir jų kodai

- Papildykite programą taip, kad būtų spausdinamos nurodyto simbolių intervalo [pr, pb] raidės ir jų kodai.

```
-----  
// Nurodyto intervalo raidės ir jų kodai
#include <iostream>
using namespace std;
-----
```

```
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    char pr, pb;
    cout << "Įveskite simbolių intervalo pradžią: ";
    cin >> pr;
    cout << "Įveskite simbolių intervalo pabaiga: ";
    cin >> pb;
    for (char ch = pr; ch <= pb; ch++)
        cout << ch << " - " << (int)ch << endl;
    cout << endl;
    return 0;
}-----
```

- Ivykdykite programą ir patikrinkite rezultatus.

#### 4 Ketvirtas žingsnis. Simbolių eilutė `string`.

- Parašykite programą, kuri pasisveikina su prisistačiusiu vartotoju.

```
-----  
// Sveikinimas
#include <iostream>
#include <string>
using namespace std;
-----  
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    string vardas;
    cout << "Koks Jūsų vardas?" << endl;
    cin >> vardas;
    cout << "Sveika(s), " << vardas << "!" << endl;
    return 0;
}-----
```

- Ivykdykite programą ir patikrinkite rezultatus.

#### 5 Penktas žingsnis. Savaitės dienos.

- Parašykite programą, kuri atpažįsta ir apibūdina savaitės dienas.

```
-----  
// Savaitės dienos
#include <iostream>
#include <string>           // darbo su eilutėmis priemonės
using namespace std;
-----  
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    string diena;
    cout << "Kokia šiandien savaitės diena?" << endl;
    cin >> diena;
    if (diena == "pirmadienis")
        cout << "Pirmadieniai visada būna sunkūs." << endl;
    else
        if (diena == "antradienis")
            cout << "Antradieniais reikia daug dirbt." << endl;
        else
            if (diena == "trečiadienis")
                cout << "Trečiadieniais dažniausiai būna saulėta." << endl;
            else
                if (diena == "ketvirtadienis")
                    cout << "Ketvirtadieniais reikia rūpintis savo sveikata." << endl;
                else
```



```

if (diena == "penktadienis")
    cout << "Penktadieniais jau kvepia savaitgaliu." << endl;
else
    if (diena == "šeštadienis")
        cout << "Šeštadieniais galima šventes švēsti." << endl;
    else
        if (diena == "sekmadienis")
            cout << "Sekmadieniais reikia eiti į bažnyčią." << endl;
        else
            cout << "Tokios savaitės dienos pas mus nebūna." << endl;
    return 0;
}
//-----

```

- Įvykdykite programą ir patikrinkite rezultatus.

### 〇Šeštas žingsnis. Savaitės dienų atpažinimas.

- Savaitės dienų atpažinimo programą papildysime, kad savaitės dienos būtų atpažįstamos net ir tuo atveju, jei dienos pavadinimas buvo įvestas didžiosiomis ir mažosiomis raidėmis (pvz., PirmaDIenis).
- Parašysime funkciją, kuri visas eilutės mažiasias raides pakeičia didžiosiomis.

```

//-----
// Savaitės dienos
#include <iostream>
#include <string>
using namespace std;
//-----
string ToUpper(string str);
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    string diena;
    cout << "Kokia šiandien savaitės diena?" << endl;
    cin >> diena;
    diena = ToUpper(diena);
    if (diena == ToUpper("pirmadienis"))
        cout << "Pirmadieniai visada būna sunkūs." << endl;
    else
        if (diena == ToUpper("antradienis"))
            cout << "Antradieniais reikia daug dirbtini." << endl;
        else
            if (diena == ToUpper("trečiadienis"))
                cout << "Trečiadieniais dažniausiai būna saulėta." << endl;
            else
                if (diena == ToUpper("ketvirtadienis"))
                    cout << "Ketvirtadieniais reikia rūpintis savo sveikata." << endl;
                else
                    if (diena == ToUpper("penktadienis"))
                        cout << "Penktadieniais jau kvepia savaitgaliu." << endl;
                    else
                        if (diena == ToUpper("šeštadienis"))
                            cout << "Šeštadieniais galima šventes švēsti." << endl;
                        else
                            if (diena == ToUpper("sekmadienis"))
                                cout << "Sekmadieniais reikia eiti į bažnyčią." << endl;
                            else
                                cout << "Tokios savaitės dienos " << diena
                                    << " pas mus nebūna." << endl;
    return 0;
}
//-----
// Eilutės str raidės keičiamos didžiosiomis
string ToUpper(string str)
{

```

```

int sz = str.length(); // simbolių skaičius eilutėje
for (int pos = 0; pos < sz; ++pos) {
    str[pos] = toupper(str[pos]);
}
return str;
}
//-----

```

Raidės simbolių lentelėje yra surašyti abéceliškai. Pirmausiai – visos didžiosios, o toliau visos mažosios. Galima nenaudoti toupper() funkcijos. Mažiasias raides pakeisti didžiosiomis galima taip:

```

//-----
string ToUpper(string str)
{
    int sz = str.length(); // simbolių skaičius eilutėje
    for (int pos = 0; pos < sz; pos++) {
        if (str[pos] >= 'a' && str[pos] <= 'z') // ar raidė?
            str[pos] += 'A' - 'a'; // pridedamas atstumas tarp raidžių
    }
    return str;
}
//-----

```

## 1.7. Kontroliniai klausimai

1. Kokia bus kintamojo i reikšmę, atlikus šiuos veiksmus:

```

for (int i = 5; i < 3; i++)
    i -= 2;

```

- 5
- 3
- 0
- Neapibrėžta

2. Kokia bus kintamojo i reikšmę, atlikus šiuos veiksmus:

```

for (int i = 5; i < 8; i += 3)
    i -= 2;

```

- 7
- 8
- 3
- Neapibrėžta

3. Kokia bus kintamojo i reikšmę, atlikus šiuos veiksmus:

```

for (int i = 4; i < 10; i += 2)
    if (i % 2 == 1)
        i += 3;
    else
        i -= 1;

```

- 8
- 10
- 5
- Neapibrėžta

4. Kokia bus kintamojo k reikšmę, atlikus šiuos veiksmus:

```

int i = 5678;
int k = ((i / 1000) * 10 + i % 10) * 10 + (i / 10) + (i % 10);

```

- a. 575
- b. 1155
- c. 580
- d. 5678

5. Kokia bus kintamojo  $k$  reikšmė, atlikus šiuos veiksmus:

```
int Skaičius(int a, int b)
{
    return a + b;
}
int i = 57;
int k = Skaičius(i/3, i % 3) - Skaičius(i / 2, 12);
```

- a. 19
- b. 21
- c. -21
- d. 40

6. Kokia bus kintamojo  $k$  reikšmė, atlikus šiuos veiksmus:

```
int Skaičius(int a, int b)
{
    if (a > b) return a;
    return b;
}
int i = 57;
int k = Skaičius(i/3, i % 3) - Skaičius(i / 4, 10);
```

- a. 29
- b. 33
- c. 9
- d. Funkcija klaidinga

7. Kokia bus kintamojo  $k$  reikšmė, atlikus šiuos veiksmus:

```
int Skaičius(int a, int b)
{
    if (a > b) return a - b;
    return b - a;
}
int k = Skaičius(Skaičius(13, 8), Skaičius(57 / 4, 20));
```

- a. 11
- b. 20
- c. k skaičiavimo reiškinys klaidingas
- d. Funkcija klaidinga

8. Kas bus matoma ekrane, atlikus šiuos veiksmus:

```
int Eil(string a, string b)
{
    if (a > b) return 1;
    if (a == b) return 0;
    return 2;
}
cout << Eil("Namas", "Nama") << Eil(" Du", "Du") << endl;
```

- a. 12
- b. 10
- c. 20
- d. 21

- 9. Kaip nustatyti, ar du double tipo skaičiai lygūs?
- 10. Kokie veiksmai galimi su simboliais ir simbolių eilutėmis

## 2. Klasė

Susipažinsite su:

- klasės duomenų tipu ir objektu – klasės tipo kintamuoju;
- objekto laukų (vidinių kintamųjų) reikšmių naudojimu ir keitimu;
- duomenų įvedimui iš tekstinio failo;
- funkcijomis, grąžinančiomis rezultatus pervardą;
- funkcijomis, grąžinančiomis rezultatus per parametrus.

### 2.1 Viena klasė

- Viena klasė, keli objektai.
- Objekto laukų reikšmių keitimasis ir naudojimas taikant objekto klasės metodus.
- Reikšmių skaičiavimai, paieška.

#### 1 užduotis. Plytos ir namas.

Gamykla gamina kelių skirtinį dydžių platas. Žinomas kiekvieno tipo platu aukštis, plotis ir ilgis. Parašykite programą, kuri suskaičiuotų, kiek reikia kiekvieno tipo platu, norint išmūryti 4 vienodas namos sienas. Langų ir durų nėra. Sienos mūrijamos tik iš vieno tipo platu.

#### Pradiniai duomenys ir rezultatai.

Pradiniai duomenys			Rezultatai
	Ilgis	Plotis	Aukštis
Pirmas platu tipas	250 mm	120 mm	88 mm
Antras platu tipas	240 mm	115 mm	71 mm
Trečias platu tipas	240 mm	115 mm	61 mm
Namo viena siena	12 m	0,23 m	3 m

#### • Programos kūrimo eiga.

- Sukuriama klasė vienos platos duomenims saugoti.
- Pagrindinėje funkcijoje `main()` skelbiami trys objektai, skirti kiekvieno platos tipo duomenims saugoti.
- Namo sienos ilgis, plotis ir aukštis perskaičiuojami platomis. Gauti skaičiai sudauginami. Gautos sienos tūris platomis ir bus ieškomas platu skaičius. Visi rezultatai pateikiama sveikaisiais skaičiais.
- Skaičiuojami kiekvieno tipo platu kiekiai namo sienoms mūryti.

#### OPirmas žingsnis.

- Sukurkite klasę `Plyta`, skirtą platos duomenims saugoti. Klasės aprašą užrašykite programos failo pradžioje, prieš `main()` funkciją.

```
//-----
class Plyta
{
private:
    int ilgis, plotis, aukštis; // milimetrais
public:
    void Dėti(int ilgioReikšmė, int pločioReikšmė, int aukščioReikšmė);
    int ImtiIlgį() { return ilgis; }
    int ImtiPlotį() { return plotis; }
    int ImtiAukštį() { return aukštis; }
};
//-----
    • Žemiau parašykite klasės metodą Dėti():
//-----
// Keičia klasės kintamuju reikšmes
// ilgioReikšmė - nauja ilgio reikšmė
// pločioReikšmė - nauja pločio reikšmė
```

// aukščioReikšmė - nauja aukščio reikšmė

void Plyta::Dėti(int ilgioReikšmė, int pločioReikšmė, int aukščioReikšmė)

```
{
    ilgis = ilgioReikšmė;
    plotis = pločioReikšmė;
    aukštis = aukščioReikšmė;
}
```

-----

- Programos failo pradžioje užrašykite sakinus, kurie reikalingi išvedimui į ekraną:

```
#include <iostream>
using namespace std;
```

- Už metodo `Dėti()` parašykite `main()` funkciją, kurioje būtų vieno tipo platu objektas. Spausdinkite objekto duomenis.

```
-----
```

```
int main()
```

```
{
```

```
    setlocale(LC_ALL, "Lithuanian");
```

```
    Plyta p1;
```

```
    p1.Dėti(250, 120, 88);
```

```
    cout << "Plytos aukštis: " << p1.ImtiAukštį() << endl;
```

```
    cout << "Plytos plotis: " << p1.ImtiPlotį() << endl;
```

```
    cout << "Plytos ilgis: " << p1.ImtiIlgį() << endl;
```

```
    cout << "Programa baigė darba\n";
```

```
    return 0;
}
```

```
-----
```

- Patikrinkite, kaip programa dirba. Ekrane turite matyti:

```
Plytos aukštis: 88
```

```
Plytos plotis: 120
```

```
Plytos ilgis: 250
```

```
Programa baigė darba
```

#### ○Antras žingsnis.

- Papildykite programą kintamaisiais, skirtais namo sienos aukščiui, ilgiui ir pločiui metrais saugoti:

```
double sienosIlgis = 12.0,
      sienosPlotis = 0.23,
      sienosAukštis = 3.0;
```

- Papildykite programą veiksmais, skaičiuojančiais, kiek platu reikia namo sienoms sumūryti:

```
int k1; // pirmo tipo platu skaičius
k1 = sienosIlgis * 1000 / p1.ImtiIlgį() *
      sienosPlotis * 1000 / p1.ImtiPlotį() *
      sienosAukštis * 1000 / p1.ImtiAukštį();
cout << "1-o tipo platu reikia: " << 4 * k1 << endl;
```

- Išbandykite, kaip veikia programa. Ekrane turėtumėte matyti:

```
Plytos aukštis: 88
```

```
Plytos plotis: 120
```

```
Plytos ilgis: 250
```

```
1-o tipo platu reikia: 12544
```

```
Programa baigė darba
```

#### ○Trečias žingsnis.

- Papildykite programą objektu, skirtu antro tipo platos duomenims saugoti ir spausdinti:

```
Plyta p2;
```

```
p2.Dėti(240, 115, 71);
```

```
cout << "Plytos aukštis: " << p2.ImtiAukštį() << endl;
```

```
cout << "Plytos plotis: " << p2.ImtiPlotį() << endl;
```

```
cout << "Plytos ilgis: " << p2.ImtiIlgį() << endl;
```

- Išbandykite, kaip veikia programa. Ekrane turėtumėte matyti:

```
Plytos aukštis: 88
```

```
Plytos plotis: 120
```

```
Plytos ilgis: 250
```

1-o tipo plytų reikia: 12544

Plytos aukštis: 71

Plytos plotis: 115

Plytos ilgis: 240

Programa baigė darbą

- Papildykite programą veiksmais, skaičiuojančiais kiek plytų reikia namo sienoms mūryti iš antro tipo plytų:

```
int k2; // antro tipo plytu skaičius
k2 = sienosIlgis * 1000 / p2.ImtiIlgij() *
      sienosPlotis * 1000 / p2.ImtiPlotij() *
      sienosAukstis * 1000 / p2.ImtiAukstij();
cout << "2-o tipo plytų reikia: " << 4 * k2 << endl;
```

- Išbandykite, kaip veikia programa. Ekrane turėtumėte matyti:

Plytos aukštis: 88

Plytos plotis: 120

Plytos ilgis: 250

1-o tipo plytų reikia: 12544

Plytos aukštis: 71

Plytos plotis: 115

Plytos ilgis: 240

2-o tipo plytų reikia: 16900

Programa baigė darbą

#### 4 Ketvirtas žingsnis.

- Parašykite funkciją, kuri suskaičiuotų, kiek plytų reikia vienai namo sienai mūryti:

```
int VienaSiena(Plyta p, double sienosPlotis, double sienosIlgis,
                 double sienosAukstis);
    • Funkcijos tekstas gali būti tokis:
//-----
// Skaičiuoja ir grąžina, kiek plytų reikia vienai sienai išmūryti
// p - objektas, kuriame yra vienos plytos duomenys
// sienosPlotis - sienos pločio reikšmė
// sienosIlgis - sienos ilgio reikšmė
// sienosAukstis - sienos aukščio reikšmė
int VienaSiena(Plyta p, double sienosPlotis, double sienosIlgis,
                 double sienosAukstis)
{
    return sienosIlgis * 1000 / p.ImtiIlgij() *
           sienosPlotis * 1000 / p.ImtiPlotij() *
           sienosAukstis * 1000 / p.ImtiAukstij();
//-----
```

- Pakeiskite plytų skaičiaus skaičiavimus kreipiniais į funkciją. Pavyzdžiu, k1 kintamajį, jo reikšmės skaičiavimą ir gauto rezultato spausdinimą galima užrašyti taip:

```
cout << "1-o tipo plytų reikia: "
      << 4 * VienaSiena(p1, sienosPlotis, sienosIlgis, sienosAukstis) << endl;
    • Išbandykite, kaip veikia programa. Ekrane turėtumėte matyti tuos pačius rezultatus.
    • Tokius keitimus padarykite ir su antro tipo plytomis. Išbandykite, kaip veikia programa. Ekrane turėtumėte matyti tuos pačius rezultatus.
```

#### 5 Penktas žingsnis.

- Papildykite programą skaičiavimais su trečio tipo plytomis.
- Išbandykite programą. Ekrane turite matyti, kad trečio tipo plytų reikia 19672.

#### 6 Šeštas žingsnis.

- Parašykite funkciją plytos duomenims ekrane spausdinti:

```
//-----
// Išveda į ekrana objekto p laukų reikšmes
void SpausdintiPlyta(Plyta p)
{
    cout << "Plytos aukštis: " << p.ImtiAukstij() << endl;
```

```
cout << "Plytos plotis: " << p.ImtiPlotij() << endl;
cout << "Plytos ilgis: " << p.ImtiIlgij() << endl;
}
```

- 
- Pakeiskite plytų duomenų spausdinimo sakinius kreipiniais į funkciją SpausdintiPlyta().
  - Išbandykite programą. Rezultatai turi nepasikeisti.

#### Programos patikrinimas.

- Pakeiskite namo sienų dydžius. Skaičiuotuvu suskaičiuokite, kiek kokio tipo plytų reikia. Patikrinkite, ar programas rezultatai sutampa su suskaičiuotais.
- Patikrinkite, kaip dirba programa, kai sienos dydžiai mažesni už plytą.

#### Programos papildymas.

Namas yra bokšto formos. Siena viena ir ji yra cilindro formos. Parašykite funkciją, kuri suskaičiuotų, kiek reikia plytų cilindro formos sienai mūryti. Tokią sieną aprašo trys dydžiai: cilindro skersmuo, aukštis ir sienos storis. Suskaičiuokite, kiek reikia kiekvieno tipo plytų bokštui mūryti.

#### Savarankiško darbo užduotis.

Statomas nestandartinės formos namas. Visos keturios sienos stačiakampės formos, tačiau skirtingo aukščio ir ilgio. Visų sienų plotis yra dviejų plytų. Yra dviejų tipų plytos. Išorinė sienos dalis mūrijama iš pirmo tipo plytų, o vidinė – iš antro. Parašykite programą, kuri suskaičiuotų, kiek kurio tipo plytų reikia namui sumūryti.

#### 2 užduotis. Lietuvos keliai.

Duoti trijų Lietuvos kelių duomenys:

- 1) pavadinimas: "Kaunas - Vilnius", ilgis: 105,0 km, leistinas greitis: 110 km/val.;
- 2) pavadinimas: "Kaunas - Alytus", ilgis: 65,6 km, leistinas greitis: 90 km/val.;
- 3) pavadinimas: "Vilnius - Panevėžys", ilgis: 136,0 km, leistinas greitis: 120 km/val.

Užrašykite klasę kelio duomenims (pavadinimas, ilgis, leistinas greitis) saugoti ir šios klasės objektus aukščiau nurodytų kelių duomenims saugoti. Parašykite programą, kuri rastų:

- per kiek laiko nuvažiuosime iš Alytaus į Panevėžį, jei važiuosime leistinu greičiu maršrute Alytus - Kaunas - Vilnius – Panevėžys;
- kuris kelias yra ilgiausias (rasti kelio pavadinimą).

#### Pradiniai duomenys ir rezultatai.

Pradiniai duomenys
1) pavadinimas: "Kaunas - Vilnius", ilgis: 105,0 km, leistinas greitis: 110 km/val.;
2) pavadinimas: "Kaunas - Alytus", ilgis: 65,6 km, leistinas greitis: 90 km/val.;
3) pavadinimas: "Vilnius - Panevėžys", ilgis: 136,0 km, leistinas greitis: 120 km/val.

Rezultatai
Iš Alytaus į Panevėžį nuvažiuosime per 2.81677 val. Ilgiausias kelias: Vilnius - Panevėžys

#### Programos kūrimo eiga.

- Sukuriama klasė vieno kelio duomenims saugoti.
- Pagrindinėje funkcijoje main() skelbiama trys užrašytos klasės objektai, skirti trijų Lietuvos kelių duomenims saugoti.
- Taikant klasės sąsajos metodus, suteikiamos reikšmės objektų kintamiesiems.
- Randamas laikas, per kurį galima nuvažiuoti iš Alytaus į Panevėžį, važiuojant leistinu greičiu maršruto Alytus - Kaunas - Vilnius - Panevėžys.
- Randamas ilgiausio kelio pavadinimas.

#### 7 Pirmas žingsnis.

- Sukurkite klasę, skirtą kelio duomenims saugoti.

```
//-----
class Kelias
{
private:
    string pav; // kelio pavadinimas
    double ilgis; // kelio ilgis
```

```

int lgr; // leistinas greitis km/val
public:
    void DėtiPav(string pavadinimas) { pav = pavadinimas; }
    void DėtiIlgis(double ilg) { ilgis = ilg; }
    void DėtiLeistGreiti(int lg) { lgr = lg; }
    string ImtiPav() { return pav; }
    double ImtiIlgis() { return ilgis; }
    int ImtiLeistGreiti() { return lgr; }

};

//-----
• Parašykite main() funkciją, kurioje būtų aprašyti reikalingi objektai ir suteiktos reikšmės šių objektų kintamiesiems. Spausdinkite objektų duomenis.
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    // Duomenų priskyrimas ir spausdinimas
    Kelias k1, k2, k3; // objektai
    k1.DėtiPav("Kaunas - Vilnius");
    k1.DėtiIlgis(105.0);
    k1.DėtiLeistGreiti(110);
    k2.DėtiPav("Kaunas - Alytus");
    k2.DėtiIlgis(65.6);
    k2.DėtiLeistGreiti(90);
    k3.DėtiPav("Vilnius - Panevėžys");
    k3.DėtiIlgis(136.0);
    k3.DėtiLeistGreiti(120);
    cout << "Keliai (pavadinimas, ilgis, leistinas greitis):" << endl;
    cout << k1.ImtiPav() << ", " << k1.ImtiIlgis() << ", "
        << k1.ImtiLeistGreiti() << endl;
    cout << k2.ImtiPav() << ", " << k2.ImtiIlgis() << ", "
        << k2.ImtiLeistGreiti() << endl;
    cout << k3.ImtiPav() << ", " << k3.ImtiIlgis() << ", "
        << k3.ImtiLeistGreiti() << endl;
    cout << "-----\n";
    cout << "Programa baigė darbą\n";
    return 0;
}
//-----

```

- Patikrinkite programos veikimą. Ekrane turite matyti:
- Keliai (pavadinimas, ilgis, leistinas greitis):  
 Kaunas - Vilnius, 105, 110  
 Kaunas - Alytus, 65.6, 90  
 Vilnius - Panevėžys, 136, 120  
 Programa baigė darbą

### ¶ Antras žingsnis.

- Papildykite programą važiavimo laiko skaičiavimo ir spausdinimo veiksmais.

```

// Važiavimo laiko radimas
double laikas = k2.ImtiLeistGreiti() +
    k1.ImtiIlgis() / k1.ImtiLeistGreiti() +
    k3.ImtiIlgis() / k3.ImtiLeistGreiti();
cout << "Iš Alytaus į Panevėžį nuvažiuosime per " << laikas
    << " val." << endl;

```

- Patikrinkite programos veikimą. Ekrane turite matyti:

Keliai (pavadinimas, ilgis, leistinas greitis):  
 Kaunas - Vilnius, 105, 110  
 Kaunas - Alytus, 65.6, 90  
 Vilnius - Panevėžys, 136, 120  
 Iš Alytaus į Panevėžį nuvažiuosime per 2.81677 val.  
 Programa baigė darbą

### ¶ Trečias žingsnis.

- Papildykite programą ilgiausio kelio radimo ir spausdinimo veiksmais.
- ```

// Ilgiausio kelio radimas
string maxPav = k1.ImtiPav(); double maxIlgis = k1.ImtiIlgis();
if (k2.ImtiIlgis() > maxIlgis) {
    maxPav = k2.ImtiPav();
    maxIlgis = k2.ImtiIlgis();
}
if (k3.ImtiIlgis() > maxIlgis) {
    maxPav = k3.ImtiPav();
    maxIlgis = k3.ImtiIlgis();
}
cout << "Ilgiausias kelias: " << maxPav << endl;

```

- Patikrinkite programos veikimą. Ekrane turite matyti:

Keliai (pavadinimas, ilgis, leistinas greitis):  
 Kaunas - Vilnius, 105, 110  
 Kaunas - Alytus, 65.6, 90  
 Vilnius - Panevėžys, 136, 120  
 Iš Alytaus į Panevėžį nuvažiuosime per 2.81677 val.  
 Ilgiausias kelias: Vilnius - Panevėžys  
 Programa baigė darbą

### Programos papildymas.

Papildykite programą veiksmais, nustatančiais, kuriame kelyje leistinas greitis yra mažiausias.

### Savarankiško darbo užduotis.

Yra keturi draugai. Žinomi kiekvieno jų vardas, ūgis, svoris. Sukurkite klasę vieno asmens duomenims saugoti. Parašykite programą, kuri rastų vidutinį vieno vaikino svorį ir kuris vaikinas žemiausias.

## 2.2. Dvi klasės

- Skirtingų klasių objektai.
- Reikšmių įvedimas klaviatūra. Skaičiavimai.

### 1 užduotis.

Papildykite 2.1 skyrelio pirmojo pratimo (1 užduotis) programą klase namo sienos duomenims saugoti.

### ¶ Pirmas žingsnis.

Nukopijuokite nurodyto pratimo programą. Pašalinkite sakinius ir kintamuosius, skirtus darbui su konkrečia siena. Taip pat pašalinkite veiksmus su antro ir trečio tipo plytomis.

### ¶ Antras žingsnis.

- Sukurkite klasę, skirtą namo sienos duomenims saugoti. Jos aprašą užrašykite prieš main() funkciją.

```

//-----
class Siena
{
private:
    double ilgis, plotis, aukštis; // metrais
public:
    void Dėti(double ilgioReikšmė, double pločioReikšmė, double aukščioReikšmė);
    double ImtiIlgis() { return ilgis; }
    double ImtiPlotis() { return plotis; }
    double ImtiAukštis() { return aukštis; }
};
//-----

```

- Parašykite klasės metodą Dėti(), kurį užrašykite už klasės Siena aprašo, bet prieš main() funkciją.

```

//-----
// Keičia klasės kintamuujų reikšmes
// ilgioReikšmė - nauja ilgio reikšmė

```

```
// pločioReikšmė - nauja pločio reikšmė
// aukščioReikšmė - nauja aukščio reikšmė
void Siena::Dėti(double ilgioReikšmė, double pločioReikšmė,
                  double aukščioReikšmė)
{
    ilgis = ilgioReikšmė;
    plotis = pločioReikšmė;
    aukštis = aukščioReikšmė;
}
//-----
```

- Papildykite main() funkciją namo sienos objektu. Spausdinkite duomenis.

```
-----int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Plyta p1;
    p1.Dėti(250, 120, 88);
    SpausdintiPlyta(p1);
    Siena s1;
    s1.Dėti(12.0, 0.23, 3.0);
    cout << "Sienos aukštis: " << s1.ImtiAukštį() << endl;
    cout << "Sienos plotis: " << s1.ImtiPlotį() << endl;
    cout << "Sienos ilgis: " << s1.ImtiIlgį() << endl;
    cout << "Programa baigė darbą\n";
    return 0;
}
//-----
```

- Patikrinkite, kaip programa dirba. Ekrane turite matyti:

```
Plytos aukštis: 88
Plytos plotis: 120
Plytos ilgis: 250
Sienos aukštis: 3
Sienos plotis: 0.23
Sienos ilgis: 12
Programa baigė darbą
```

### ¶ Trečias žingsnis.

- Parašykite funkciją, kuri suskaičiuotų, kiek reikia plytų duotai sienai sumūryti:

```
-----// Suskaičiuoja ir gražina sienai sumūryti reikalingų plytu kiekj
// p - plytos duomenys
// s - sienos duomenys
int ReikiaPlytų(Plyta p, Siena s)
{
    return s.ImtiIlgį() * 1000 / p.ImtiIlgį() *
           s.ImtiPlotį() * 1000 / p.ImtiPlotį() *
           s.ImtiAukštį() * 1000 / p.ImtiAukštį();
}
//-----
```

- Papildykite programą sakiniu:

```
cout << "Reikia plytų: " << 4 * ReikiaPlytų(p1, s1) << endl;
```

- Patikrinkite, kaip dirba programa. Ar rezultatai teisingi? Kadangi veiksmuose yra sveikieji ir realieji skaičiai, o rezultatas – sveikasis skaičius, tuomet galima vienos plytos paklaida.

### ¶ Ketvirtas žingsnis.

- Papildykite programą objektais, skirtais antro ir trečio tipo plytų duomenims saugoti.
- Atlikite skaičiavimus. Patikrinkite, ar gaunami teisingi rezultatai.

### Programos papildymas.

- Namo ilgis ir plotis nevienodi. Jeigu namo plotą pažymėsime a, ilgį b, o aukštį c, tuomet bus dvi sienos, kurių dydis bus a \* c ir dvi sienos, kurių dydis bus b \* c. Sukurkite du objektus sienų duomenims saugoti.
- Papildykite programą skaičiavimais, kiek reikia kiekvieno tipo plytų tokio namo sienoms sumūryti.
- Patikrinkite, ar programa teisingai skaičiuoja.

### Savarankiško darbo užduotis.

Statoma stačiakampio formos pilis su apvaliais gynybiniais bokštais kampuose. Yra žinomi duomenys apie sienas: aukštis, plotis ir ilgis. Bokstai yra apvalūs ir yra žinomi jų aukščiai, skersmenys ir storai. Pilis mūrijama iš vieno tipo plytų. Parašykite programą, kuri suskaičiuotų, kiek plytų reikia piliai sumūryti.

### 2 užduotis. Kelionės kaina.

Duoti Lietuvos kelių duomenys:

- |                                        |                  |                                 |
|----------------------------------------|------------------|---------------------------------|
| 1) pavadinimas: "Kaunas - Vilnius",    | ilgis: 105,0 km, | leistinas greitis: 110 km/val.; |
| 2) pavadinimas: "Kaunas - Alytus",     | ilgis: 65,6 km,  | leistinas greitis: 90 km/val.;  |
| 3) pavadinimas: "Vilnius - Panevėžys", | ilgis: 136,0 km, | leistinas greitis: 120 km/val.  |

Duoti automobilių duomenys:

- |                                    |                          |                                |
|------------------------------------|--------------------------|--------------------------------|
| 1) pavadinimas: "Opel Meriva",     | degalų tipas: benzinas,  | degalų sąnaudos 100 km: 7,5 l; |
| 2) pavadinimas: "Volkswagen Golf", | degalų tipas: dyzelinas, | degalų sąnaudos 100 km: 6,3 l. |

Sukurkite dvi klasses ir šių klasių objektus aukščiau nurodytų kelių ir automobilių duomenims saugoti. Kiek kainuos degalai keliaujant maršrutu Alytus - Kaunas - Vilnius - Panevėžys kiekvienu automobiliui? Benzino ir dyzelino kainos kelionės dieną yra žinomas ir įvedamos klaviatūra.

### Pradiniai duomenys ir rezultatai.

| Pradiniai duomenys                                                                         |
|--------------------------------------------------------------------------------------------|
| Keliai:                                                                                    |
| 1) pavadinimas: "Kaunas - Vilnius", ilgis: 105,0 km, leistinas greitis: 110 km/val.;       |
| 2) pavadinimas: "Kaunas - Alytus", ilgis: 65,6 km, leistinas greitis: 90 km/val.;          |
| 3) pavadinimas: "Vilnius - Panevėžys", ilgis: 136,0 km, leistinas greitis: 120 km/val.     |
| Automobiliai:                                                                              |
| 1) pavadinimas: "Opel Meriva", degalų tipas: benzinas, degalų sąnaudos 100 km: 7,5 l;      |
| 2) pavadinimas: "Volkswagen Golf", degalų tipas: dyzelinas, degalų sąnaudos 100 km: 6,3 l. |
| 1 litro benzino kaina: 4,75 Lt, 1 litro dyzelino kaina: 4,45 Lt.                           |
| Rezultatai                                                                                 |
| Automobiliu Opel Meriva iš Alytaus į Panevėžį nuvažiuosime už 109.23 Lt.                   |
| Automobiliu Volkswagen Golf iš Alytaus į Panevėžį nuvažiuosime už 85.96 Lt.                |

### • Programos kūrimo eiga.

- Nukopijuojama 2.1 skyrelio 2 užduoties programa.
- Sukuriama nauja klasė automobilio duomenims saugoti.
- Pagrindinėje funkcijoje main() skelbiami du nauji automobilio klasės objektai.
- Suteikiamos reikšmės objektų kintamiesiems.
- Klaviatūra įvedamos benzino ir dyzelino kainos.
- Randama, kiek kainuos kelionė iš Alytaus į Panevėžį per Kauną ir Vilnių pirmuoju automobiliu.
- Randama, kiek kainuos kelionė iš Alytaus į Panevėžį per Kauną ir Vilnių antruoju automobiliu.

### ¶ Pirmas žingsnis.

- Nukopijuokite 2.1 skyrelio 2 užduoties programą. Pašalinkite iš programos veiksmus, surandančius laiką, per kurį galima nuvažiuoti iš Alytaus į Panevėžį, ir ilgiausio kelio pavadinimą.

### ¶ Antras žingsnis.

- Sukurkite klasę, skirtą automobilio duomenims saugoti.

```
-----class Auto
{
private:
```

```

    string pav;           // automobilio pavadinimas
    string degalai;     // degalų tipas
    double sanaudos;    // kuro sanaudos 100 km
public:
    void DėtiPav(string p) { pav = p; }
    void DėtiDegalus(string d) { degalai = d; }
    void DėtiSanaudas(double s) { sanaudos = s; }
    string ImtiPav() { return pav; }
    string ImtiDegalus() { return degalai; }
    double ImtiSanaudas() { return sanaudos; }
};

//-----

```

- Papildykite main() funkciją dvieju objektu, saugančiu automobiliu duomenis, aprašais. Suteikite reikšmę šių objektų kintamiesiems ir spausdinkite objektų duomenis.

```

Auto a1;           // automobilius aprašantys objektais
a1.DėtiPav("Opel Meriva");
a1.DėtiDegalus("benzinas");
a1.DėtiSanaudas(7.5);
a2.DėtiPav("Volkswagen Golf");
a2.DėtiDegalus("dyzelinas");
a2.DėtiSanaudas(6.3);
cout << "Automobiliai:" << endl;
cout << a1.ImtiPav() << ", " << a1.ImtiDegalus() << ", "
    << a1.ImtiSanaudas() << endl;
cout << a2.ImtiPav() << ", " << a2.ImtiDegalus() << ", "
    << a2.ImtiSanaudas() << endl;

```

- Patikrinkite programos veikimą. Ekrane turite matyti:

Keliai (pavadinimas, ilgis, leistinas greitis):  
 Kaunas - Vilnius, 105, 110  
 Kaunas - Alytus, 65.6, 90  
 Vilnius - Panevėžys, 136, 120  
 Automobiliai:  
 Opel Meriva, benzinas, 7.5  
 Volkswagen Golf, dyzelinas, 6.3  
 Programa baigė darbą

### ¶ Trečias žingsnis.

- Papildykite programą kintamaisiais, kuriuose būtų saugomos benzino ir dyzelino kainos. Įveskite šiu kintamuojų reikšmes klaviatūra.

```

// Degalų kainų įvedimas klaviatūra
double benzKaina, dyzKaina;
cout << "Kiek kainuoja 1 litras benzino? "; cin >> benzKaina;
cout << "Kiek kainuoja 1 litras dyzelino? "; cin >> dyzKaina;

```

- Patikrinkite programos veikimą, kai 1 litras benzino kainuoja 4,75 Lt, o 1 litras dyzelino – 4,45 Lt. Ekrane turite matyti:

Keliai (pavadinimas, ilgis, leistinas greitis):  
 Kaunas - Vilnius, 105, 110  
 Kaunas - Alytus, 65.6, 90  
 Vilnius - Panevėžys, 136, 120  
 Automobiliai:  
 Opel Meriva, benzinas, 7.5  
 Volkswagen Golf, dyzelinas, 6.3  
 Kiek kainuoja 1 litras benzino? 4.75  
 Kiek kainuoja 1 litras dyzelino? 4.45  
 Programa baigė darbą

### ¶ Ketvirtas žingsnis.

- Parašykite funkciją, kuri suskaičiuotų kelionės kainą:

```

//-----
// Suskaičiuoja ir gražina kelionės kaina
// atstumas - kelionės ilgis (km)
// sanaudos - kuro sanaudos 100 km
// litroKaina - vieno litro degalų kaina (Lt)

```

```

double KelionėsKaina(double atstumas, double sanaudos, double litroKaina)
{
    return (atstumas / 100) * sanaudos * litroKaina;
}
//-----


- Papildykite programą kintamaisiais ir veiksmais, reikalingais kelionės kainai skaičiuoti ir spausdinti.


// Kelionės išlaidų skaičiavimas
double kaina1, kaina2;
double atstumas = k2.ImtiIlgij() + k1.ImtiIlgij() + k3.ImtiIlgij();
if (a1.ImtiDegalus() == "benzinas")
    kaina1 = KelionėsKaina(atstumas, a1.ImtiSanaudas(), benzKaina);
else
    kaina1 = KelionėsKaina(atstumas, a1.ImtiSanaudas(), dyzKaina);
if (a2.ImtiDegalus() == "benzinas")
    kaina2 = KelionėsKaina(atstumas, a2.ImtiSanaudas(), benzKaina);
else
    kaina2 = KelionėsKaina(atstumas, a2.ImtiSanaudas(), dyzKaina);
cout << "Automobiliu " << a1.ImtiPav()
    << " iš Alytaus į Panevėžį nuvažiuosime už " << kaina1 << " Lt" << endl;
cout << "Automobiliu " << a2.ImtiPav()
    << " iš Alytaus į Panevėžį nuvažiuosime už " << kaina2 << " Lt" << endl;

```

- Patikrinkite programos veikimą. Ekrane turite matyti:

Keliai (pavadinimas, ilgis, leistinas greitis):  
 Kaunas - Vilnius, 105, 110  
 Kaunas - Alytus, 65.6, 90  
 Vilnius - Panevėžys, 136, 120  
 Automobiliai:  
 Opel Meriva, benzinas, 7.5  
 Volkswagen Golf, dyzelinas, 6.3  
 Kiek kainuoja 1 litras benzino? 4.75  
 Kiek kainuoja 1 litras dyzelino? 4.45  
 Automobiliu Opel Meriva iš Alytaus į Panevėžį nuvažiuosime už 109.226 Lt  
 Automobiliu Volkswagen Golf iš Alytaus į Panevėžį nuvažiuosime už 85.9553 Lt  
 Programa baigė darbą

### Programos papildymas.

Papildykite programą klase Degalai (tipas, 1 litro kaina) ir šios klasės objektais benzino ir dyzelino duomenims saugoti. Pertvarkykite kelionės iš Alytaus į Panevėžį kainos skaičiavimą.

### Savarankiško darbo užduotis.

Yra keturi draugai. Žinomi kiekvieno jų vardas, ūgis, svoris. Krepšinio komanda renka žaidėjus į dvi pozicijas, apie kurias žinoma: pavadinimas, ūgio intervalas. Parašykite programą, kuri pateiktų rekomendacijas, į kurią iš pozicijų gali pretenduoti kiekvienas iš draugų.

### 2.3. Kontroliniai klausimai

- Kiek kintamuojų galima aprašyti klasės viduje?
- Kaip vadinami programos kintamieji, kurių tipas – klasė?
- Kiek programoje gali būti kintamuojų, kurių tipas – klasė?
- Kurie klasės elementai (kintamieji, metodai) vadinami uždaraisiais, o kurie – atviraisiais?
- Ar skiriasi atvirųjų ir uždarųjų klasės elementų naudojimas klasės viduje?
- Kokiui būdu galima pakeisti uždarųjų klasės kintamuojų reikšmes klasės viduje?
- Kokiui būdu galima pakeisti uždarųjų klasės kintamuojų reikšmes klasės išorėje?
- Kuo skiriasi klasės metodo realizavimas klasės išorėje nuo realizavimo klasės viduje?
- Kaip programoje naudojami atvirieji klasės metodai?
- Kurių metodų pagalba sužinomas klasės uždarųjų kintamuojų reikšmės?

## 2.4. Užduotys

### U2–1. Sodas

- Sukurkite klasę **Medis**, kuri turėtų kintamuosius amžiu ir aukščiu saugoti. Sode auga trys skirtinges liepos. Raskite, koks aukščiausios liepos aukštis ir koks seniausios liepos amžius.
- Papildykite klasę **Medis** kintamuoju, skirtu medžio lajos skersmeniui saugoti. Sukurkite klasę **Sodas**, kuris turėtų kintamuosius sodo ilgiui ir pločiui saugoti. Kiek sode gali augti kiekvieno dydžio liepu?
- Papildykite klasę **Sodas** metodu **Dėti()**, kuris leistų keisti sodo dydį. Ar sodo plotą padidinus dvigubai tame tilps tiek didžiausios lajos liepu, kiek mažesniame sode tilpo mažiausios lajos liepu?

### U2–2. Liftas

- Sukurkite klasę **Studentas**, kuri turėtų kintamuosius amžiu ir ūgiui saugoti. Trys studentai nutarė treniruotis žaisti krepšini. Raskite, koks aukščiausio studento amžius ir koks jauniausio studento ūgis.
- Papildykite klasę **Studentas** kintamuoju, skirtu studento svorii saugoti. Sukurkite klasę **Liftas**, kuri turėtų kintamuosius lifto keliamosios galios reikšmei ir talpai saugoti. Per kelis kartus visi studentai pakils liftu į reikiama aukštą?
- Papildykite klasę **Liftas** metodais **Dėti()**, kurie leistų keisti lifto keliamają galią ir talpą. Ar visi studentai vienu metu bus pakelti į reikiama aukštą, jeigu lifto keliamoji galia bus padvigubinta? O jeigu talpa bus padvigubinta?

### U2–3. Studentas

- Sukurkite klasę **Cukrus**, kuri turėtų kintamuosius masei ir 1 kg kainai saugoti. Parduotuvėje yra trys skirtingu gamintojų cukraus maišai. Raskite, kokia sunkiausio maišo masė ir kiek tas cukraus maišas kainuoja. Kiek kainuoja lengviausio maišo vienas cukraus kilogramas?
- Pildykite klasę **Cukrus** kintamuoju, skirtu pagaminimo datai saugoti. Sukurkite klasę **Studentas**, kuri turėtų kintamuosius pinigų kiekiui litais ir centais saugoti. Kuriuos maišus gali nupirkти studentas, jeigu jam reikia bet kurio vieno.
- Papildykite klasę **Studentas** metodais **Dėti()**, kuris leistų keisti studento turimų pinigų kiekį. Kuriuos maišus studentas gali nupirkti, jeigu jo turimų pinigų suma padvigubėtų?

### U2–4. Studentas

- Sukurkite klasę **Pabaisa**, kuri turėtų kintamuosius ragų ir uodegų kiekiui saugoti. Susitiko gūdžiame miške trys pabaisos. Kiek iš viso jos turi ragų ir uodegų? Kiek ragų turi mažiausiai uodeguota pabaisa?
- Pildykite klasę **Pabaisa** kintamuoju, skirtu amžiu saugoti. Sukurkite klasę **Studentas**, kuri turėtų kintamuosius trofėjų skaičiui (atskirai ragams ir uodegom) saugoti. Kiek studentas turės trofėjų, jeigu sunaikins vyriausią pabaisą?
- Papildykite klasę **Studentas** metodais **Dėti()**, kurie leistų keisti studento turimų trofėjų skaičius. Du studentai sumedžiojo dvi pabaisas: pirmasis vyriausią pabaisą, o antrasis – jauniausią. Papildykite studentų turimų trofėjų skaičius. Kuris studentas turi daugiau uodegų ir kuris daugiau ragų savo kolekcijoje?

### U2–5. Patalpa

- Sukurkite klasę **Lenta**, kuri turėtų kintamuosius ilgiui ir pločiui saugoti. Sandėlyje yra trijų tipų lento. Raskite, kurio tipo lento yra ilgiausios ir kurios plačiausios.
- Papildykite klasę **Lenta** kintamuoju, skirtu lento storii saugoti. Sukurkite klasę **Patalpa**, kuri turėtų kintamuosius patalpos ilgiui ir pločiui saugoti. Salėje reikia sudėti grindis, kurių storis yra ne mažesnis kaip 4 cm. Salės išmatavimai žinomi. Kiek reikia tinkamo storio lento grindims patalpoje sudėti.
- Papildykite klasę **Patalpa** metodais **Dėti()**, kurie leistų keisti patalpos dydį – ilgi ir plotę. Kurių lento reikės mažiausiai padidinus patalpos ilgį x cm, o plotį y cm?

### U2–6. Dėžė

- Sukurkite klasę **Vinis**, kuri turėtų kintamuosius ilgiui ir storii saugoti. Parduotuvėje yra trijų tipų vynys. Raskite, kurio tipo vynys yra ilgiausios ir kurios storius.

- Papildykite klasę **Vinis** kintamuoju, skirtu vynes svoriui saugoti. Sukurkite klasę **Dėžė**, kuri turėtų kintamuosius dėžės ilgiui, pločiui ir aukščiu saugoti. Reikia sukti dėžę, kurioje vynys kalamos kas 10 cm, jeigu jų ilgis yra 8 cm arba kas 15 cm jeigu jų ilgis yra 10 cm. Kiek reikia tinkamo ilgio vinių tokiai dėžei sukti?
- Papildykite klasę **Dėžė** metodais **Dėti()**, kurie leistų keisti dėžės dydį – ilgi, plotę ir aukštę. Kurių vinių reikės daugiausiai padidinus dėžės aukštį x cm?

### U2–7. Salė

- Sukurkite klasę **Kėdė**, kuri turėtų kintamuosius jos užimamam plotui – projekcijai ant grindų saugoti. Sandėlyje yra trijų tipų kėdės. Raskite, kurio tipo kėdės užima didžiausią plotą.
- Papildykite klasę **Kėdė** kintamuoju, skirtu sėdimos plokštumos aukščiu nuo grindų saugoti. Sukurkite klasę **Salė**, kuri turėtų kintamuosius salės ilgiui ir pločiui saugoti. Salėje reikia sustatyti kėdes, kurių sėdimos plokštumos aukštis nuo grindų gali būti tarp 40 ir 50 cm, o atstumas tarp kėdžių eilių 30 cm. Salės išmatavimai žinomi. Kiek reikia tinkamo aukščio kėdžių, jeigu per salės vidurį ir kraštose reikia palikti 1 m pločio praėjimus?
- Papildykite klasę **Salė** metodais **Dėti()**, kurie leistų keisti salės dydį – ilgi ir plotę. Kurių kėdžių reikės mažiausiai sumažinus salės ilgį x cm?

### U2–8. Namas

- Sukurkite klasę **Ekovata** (apšiltinimo medžiaga), kuri turėtų kintamuosius ilgiui ir pločiui saugoti. Statybinių prekių parduotuvėje yra trijų tipų ekovata. Raskite, kurio tipo ekovata dengia didžiausią ir mažiausią plotą.
- Papildykite klasę **Ekovata** kintamuoju, skirtu vatos storii saugoti. Sukurkite klasę **Namas**, kuri turėtų kintamuosius namo ilgiui ir pločiui saugoti. Namą reikia apšiltinti ekovata. Apšiltinimo sluoksnio storis tarp 25 ir 30 cm. Kiek reikia kvadratinį metrų ir kokio storio ekovatos namui apšiltinti.
- Papildykite klasę **Namas** metodais **Dėti()**, kurie leistų keisti namo dydį – ilgi ir plotę. Kurios ekovatos reikės mažiausiai padidinus namo ilgį x cm, o plotį y cm?

### U2–9. Rąstas

- Sukurkite klasę **Lenta**, kuri turėtų kintamuosius ilgiui ir storii saugoti. Sandėlyje yra trijų tipų lento. Raskite, kurio tipo lento yra ilgiausios ir kurios storiasios.
- Papildykite klasę **Lenta** kintamuoju, skirtu lento medžio rūšiai saugoti (1-beržas, 2-eglė, 3-pušis). Sukurkite klasę **Rąstas** (medžio), kuri turėtų kintamuosius rąsto ilgiui ir skersmeniui saugoti. Kiek kurio tipo lento galima išpjauti iš vieno rąsto?
- Papildykite klasę **Rąstas** metodais **Dėti()**, kurie leistų keisti rąsto dydį – ilgi ir skersmenį. Kurių lento galima išpjauti daugiausiai padidinus rąsto skersmenį x cm?

### U2–10. Malūnas

- Sukurkite klasę **Grūdai**, kuri turėtų kintamuosius grūdų rūšiai, rupumui ir malimo nuostoliams (%) saugoti. Malūnas mala 3 skirtingu rūšių grūdus. Raskite, kurie malami grūdai rupiausi ir kurių grūdų malimo nuostoliai mažiausiai.
- Papildykite klasę **Grūdai** kintamuoju, skirtu miltų tankiui saugoti. Sukurkite klasę **Malūnas**, kuri turėtų kintamajį malūno pavadinimui saugoti ir 3 kintamuosius (kiekvienai grūdų rūšiai, tonomis), skirtus saugoti per mėnesį reikalingam gauti miltų kiekiui. Suskaičiuokite, kiek tonų kiekvienos rūšies grūdų turi būti pristatyta kas mėnesį į malūnā.
- Papildykite klasę **Malūnas** kintamuoju, skirtu saugoti bendram miltų talpyklų tūriui (litrais). Ar malūno per mėnesį primaltas miltų kiekis telpa talpyklose?

### U2–11. Kepykla

- Sukurkite klasę **Duona**, kuri turėtų kintamuosius duonos rūšiai, miltų pavadinimui, rupumui, kepaliuko svoriui ir kepimo prieaugiui (%) saugoti. Kepykla kepa 3 skirtingu rūšių duoną. Raskite, kuri kepama duona mažiausiai rupi ir kurios duonos kepimo prieaugis didžiausias.
- Papildykite klasę **Duona** kintamuoju, skirtu duonos kepaliuko užimamam ant lentynos plotui saugoti. Sukurkite klasę **Kepykla**, kuri turėtų kintamajį kepyklos pavadinimui saugoti ir 3 kintamuosius (kiekvienai duonos rūšiai), skirtus saugoti per naktį reikalingam iškepti duonos kepaliukų kiekiui. Suskaičiuokite, kiek ploto lentynų kiekvienos rūšies duonai sudėti reikės.

- Papildykite klasę Kepykla kintamuoju, kuriame būtų saugoma informacija apie tai, kiek iš viso miltų talpyklose kepykla gali turėti miltų atsargą (kilogramais). Ar kepyklai per naktį reikalingas miltų kiekis telpa talpyklose?

#### U2–12. Kepykłė.

- Sukurkite klasę Morengas, kuri turėtų kintamuosius, skirtus rūšies pavadinimui, dešimčiai vienetų morengų pagaminti reikalingų kiaušinių kiekui, vieno morengo užimamam pločiui bei ilgiui, ir kiek vienetų morengų telpa orkaitėje vienu metu kepti saugoti. Kepykłė kepa 3 skirtingų rūsių morengus. Raskite, kuriai rūšiai iškepti reikia mažiausiai kiaušinių ir kurios rūšies morengų užimamas plotas didžiausias.
- Papildykite klasę Morengas kintamuoju, skirtu saugoti laikui, reikalingu morengui kepti orkaitėje (minutėmis). Sukurkite klasę Kepykłelė, kuri turėtų kintamajį kepykłelės pavadinimui saugoti ir 3 kintamuosius (kiekvienai morengų rūšiai), skirtus saugoti per naktį reikalingam iškepti morengų kiekui. Suskaičiuokite, kiek laiko sanaudų kiekvienos rūšies morengams iškepti reikės.
- Papildykite klasę Kepykłelė kintamuoju, kuriame būtų saugomas orkaičių skaičius. Ar kepykłelė per naktį (10 darbo valandų) spēs iškepti reikiama morengų kiekį?

#### U2–13. Siuvyklė.

- Sukurkite klasę Sijonas, kuri turėtų kintamuosius modelio pavadinimui, medžiagos pavadinimui, pasiuvimui reikalingos medžiagos ilgiui bei pločiui ir atraižų kiekui (procentais, %) saugoti. Siuvyklė siuva 3 skirtingų modelių sijonus. Raskite, kuriam modeliui pasiūti reikia daugiausiai medžiagos ir kurio modelio atraižų lieka mažiausiai.
- Papildykite klasę Sijonas kintamuoju, skirtu sijonui pasiūti reikalingoms laiko sanaudoms (minutėmis) saugoti. Sukurkite klasę Siuvyklė, kuri turėtų kintamajį siuvyklos pavadinimui saugoti ir 3 kintamuosius (kiekvienai sijonų rūšiai), skirtus saugoti per savaitę reikalingam pasiūti sijonų kiekui. Suskaičiuokite, kiek reikės laiko sanaudų kiekvienos rūšies sijonams pasiūti.
- Papildykite klasę Siuvyklė kintamuoju, skirtu siuvėjų skaičiui saugoti. Ar siuvyklė per savaitę (5 darbo dienos, po 8 darbo valandas) spēs pasiūti reikiama sijonų kiekį?

#### U2–14. Vartojojimas.

- Sukurkite klasę Vanduo, kuri turėtų kintamuosius vandens telkinio gyliai ir vandens telkinio tūriui saugoti. Atliekami trijų skirtinės vandens telkininių tyrimai. Rasti mažiausią telkinio gylį ir didžiausią vandens tūrį.
- Papildykite klasę Vanduo kintamuoju, skirtu vandens telkinio plotui saugoti. Sukurkite klasę Vartojojimas, kuri turėtų kintamuosius vieno žmogaus per parą suvartojoamo vandens kiekui ir žmonių, vartojančių vandenį, skaičiui saugoti. Ar kiekvieno iš telkininių vandens užtektų visiems žmonėms trims paroms?
- Papildykite klasę Vartojojimas metodu Dėti(), kuris leistų keisti vandenį vartojančių žmonių skaičių ir vieno žmogaus per parą suvartojojamą vandens kiekį. Ar vandenį vartojančių žmonių skaičių padidinus dvigubai, jiems užteks sekliausio telkinio vandens vienai parai? Ar užteks didžiausio tūrio telkinio vandens dviem parom, jei vieno žmogaus per parą suvartojojamas vandens kiekis padvigubėja?

#### U2–15. Lentyna.

- Sukurkite klasę Knyga, kuri turėtų kintamuosius knygos puslapių skaičiui ir knygos puslapio storui saugoti. Knygynė pasirinktos trys knygos. Rasti knygą, kurios mažiausias puslapio storis ir kiek puslapių turi storiausią knygą.
- Papildykite klasę Knyga kintamuoju, skirtu knygos aukščiui saugoti. Sukurkite klasę Lentyna, kuri turėtų kintamuosius lentynos aukščiui ir ilgiui saugoti. Ar tos trys knygos visos tilps lentynoje?
- Papildykite klasę Lentyna metodu Dėti(), kuris leistų keisti lentynos aukštį ir ilgį. Ar tilps visos tos knygos lentynoje, jeigu lentynos aukštis bus padidintas du kartus? Jeigu lentynos ilgis bus padidintas du kartus?

#### U2–16. Smaguris.

- Sukurkite klasę Saldainiai, kuri turėtų kintamuosius saldainių maišelio masei ir vieno saldainių maišelio kainai saugoti. Parduotuvėje yra trijų rūsių saldainiai. Raskite, kiek sveria sunkiausias maišelis ir kiek jis kainuoja. Kiek kainuoja 1 maišelis saldainių, kurio masė yra mažiausia?

- Papildykite klasę Saldainiai kintamuoju, skirtu pardavimo kiekiui saugoti. Sukurkite klasę Smaguris, kuri turėtų kintamuosius pinigų kiekiui litais ir centais saugoti. Kurių saldainių bent vieną maišelį gali pirkti smaguris, jeigu jam reikia bet kurios vienos rūšies saldainių.
- Papildykite klasę Smaguris metodu Dėti(), kuris leistų keisti smagurio turimų pinigų kiekį. Kuriuos saldainius smaguris galėtų nupirkti, jeigu jo turimų pinigų suma padvigubėtu?

#### U2–17. Traukinys.

- Sukurkite klasę Automobilis, kuri turėtų kintamuosius keleivių, telpančių į automobilį, skaičiui ir kuro, sunaudojamo 100 km, kiekiui saugoti. Kelionei pasirinkti trys automobiliai. Raskite, kiek kainuoja pigiausia kelionė (mažiausiai kuro sunaudoja) ir keli keleiviai telpa didžiausiam automobiliu.
- Papildykite klasę Automobilis kintamuoju, skirtu kelionės atstumui saugoti. Sukurkite klasę Traukinys, kuri turėtų kintamuosius nuvažiuotam atstumui ir bilieto kainai saugoti. Kurie automobiliai nuvažiuos didesnį atstumą, negu traukiniu nuvažiuotas atstumas?
- Papildykite klasę Traukinys metodu Dėti(), kuris leistų keisti traukinio nuvažiuotą atstumą ir bilieto kainą. Ar apsimoka mažiausio automobilio keleiviams važiuoti traukiniu, jei nuvažiuotas atstumas bus padidintas du kartus? Jeigu bilieto kaina bus padidinta du kartus? Jeigu traukiniu nuvažiuotas atstumas ir bilieto kaina bus padidinti du kartus?

#### U2–18. Monetas.

- Sukurkite klasę Moneta, kuri turėtų kintamuosius nominalui ir svoriui saugoti. Studentas gavo tris marsiečių monetas. Kuri moneta sunkiausia ir kokia visų monetų bendra piniginė vertė?
- Papildykite klasę Moneta kintamuoju, skirtu monetos storui saugoti. Sukurkite klasę Cilindras, kuri turėtų kintamuosius ilgiui ir skersmeniui saugoti. Kurioms monetoms saugoti tinkamai cilindrinė taupyklė? Kiek kiekvieno nominalo monetų atskirai galima sutalpinti į tokio tipo taupyklę? Kiek iš viso svertų tokį trijų taupyklų monetos?
- Papildykite klasę Cilindras metodais Dėti(), kuris leistų keisti cilindro ilgi ir skersmenį. Kiekvieno nominalo turime m<sub>1</sub>, m<sub>2</sub> ir m<sub>3</sub> monetų. Kokios turi būti talpyklos kiekvienam turimų monetų skaičiui saugoti?

#### U2–19. Malūnas.

- Sukurkite klasę Grūdai, kuri turėtų kintamuosius grūdų rūšiai, pirkimo kainai (Lt/t), malimo nuostoliams (procentais, %) saugoti. Malūnas mala 3 skirtinės rūsių grūdus. Raskite, kurių grūdų malimo nuostoliai mažiausiai ir kokia vidutinė 1 tonos grūdų pirkimo kaina.
- Papildykite klasę Grūdai kintamuoju, skirtu miltų pardavimo kainai saugoti. Sukurkite klasę Malūnas, kuri turėtų kintamajį malūno pavadinimui saugoti ir 3 kintamuosius (kiekvienai grūdų rūšiai, tonomis), skirtus saugoti per mėnesį sumalamą grūdų kiekui. Suskaičiuokite, kiek tonų kiekvienos rūšies grūdų turi būti pristatyta kas mėnesį į malūną, kiek pinigų už juos sumokama ir kiek pinigų gaunama pardavus miltus.
- Papildykite klasę Malūnas metodu Dėti(), kuris leistų keisti per mėnesį sumalamą grūdų kiekius. Kiek kartų padidės pinigų suma, gaunama už parduotus miltus, jei sumalamą grūdų kiekiai padidės 1,5 karto?

#### U2–20. Duona.

- Sukurkite klasę Duona, kuri turėtų kintamuosius duonos rūšiai, miltų pavadinimui, miltų kiekui viename kepaliuke (kg), kepaliuko svoriumi (kg), ir pardavimo kainai (Lt/kg) saugoti. Kepykla kepa 3 skirtinės rūsių duoną. Raskite vidutinį kepaliuko svorį ir kurios duonos kaina mažiausia.
- Papildykite klasę Duona kintamuoju, skirtu duonos kepaliuko užimamam ant lentynos plotui saugoti. Sukurkite klasę Kepykla, kuri turėtų kintamajį kepyklos pavadinimui saugoti ir 3 kintamuosius (kiekvienai duonos rūšiai), skirtus saugoti per naktį reikalingam iškepti duonos kepaliukų kiekui. Suskaičiuokite, kiek ploto lentynų kiekvienos rūšies duonai sudėti reikės ir kiek bus gauta pardavus duoną.
- Papildykite klasę Kepykla kintamuoju, kuriame būtų saugoma informacija apie tai, kiek iš viso miltų talpyklose kepykla gali turėti miltų atsargą (kilogramais). Ar kepyklai per naktį reikalingas miltų kiekis telpa talpyklose?

**U2–21. Siuvyklė.**

- Sukurkite klasę `Švarkas`, kuri turėtų kintamuosius modelio pavadinimui, medžiagos pavadinimui, jos kainai ( $\text{Lt}/\text{m}^2$ ), pasiuvimui reikalingos medžiagos ilgiui bei pločiui, atraižą kiekui (procenatais, %) ir pardavimo kainai saugoti. Siuvyklė siuva 3 skirtinį modelių švarkus. Raskite, kuriam modeliui pasiūti reikia mažiausiai medžiagos, kokia vidutinė švarko kaina.
- Papildykite klasę `Švarkas` kintamuoju, skirtu švarkui pasiūti reikalingoms laiko sąnaudoms (minutėmis) saugoti. Sukurkite klasę `Siuvyklė`, kuri turėtų kintamajį siuvyklos pavadinimui saugoti ir 3 kintamuosius (kiekvienai švarko rūšiai), skirtus saugoti per savaitę reikalingam pasiūti švarkų kiekui. Suskaičiuokite, kiek reikės laiko sąnaudų kiekvienos rūšies švarkams pasiūti, kiek bus sumokėta už medžiagą ir kiek pinigų bus gauta pardavus švarkus.
- Papildykite klasę `Siuvyklė` kintamuoju, skirtu siuvėjų skaičiui saugoti. Ar siuvyklė per savaitę (5 darbo dienos, po 8 darbo valandas) spēs pasiūti reikiama švarkų kiekį?

**U2–22. Studentas**

- Sukurkite klasę `Cukrus`, kuri turėtų kintamuosius 1 kg. kainai ir gamintojo pavadinimui saugoti. Parduotuvėje yra trys skirtinį gamintojų cukraus vienodo dydžio (50 kg) maišai. Raskite, kiek kainuoja brangiausio cukraus maišas ir kas gamintojas. Kokia vidutinė 1 maišo kaina?
- Papildykite klasę `Cukrus` kintamuoju, skirtu pagaminimo datai saugoti. Sukurkite klasę `Studentas`, kuri turėtų kintamuosius pinigų kiekui litais ir centais saugoti. Kuriuos maišus gali nupirkti studentas, jeigu jam reikia bet kurio vieno.
- Papildykite klasę `Studentas` metodais `Dėti()`, kuris leistų keisti studento turimų pinigų kiekį. Kuriuos maišus studentas gali nupirkti, jeigu jo turimų pinigų suma padvigubėtų?

**U2–23. Kepykla.**

- Sukurkite klasę `Duona`, kuri turėtų kintamuosius duonos miltų pavadinimui, kepaliuko svoriui ir kainai saugoti. Kepykla kepa duoną iš 3 skirtinų rūšių miltų. Raskite, kuri kepama duona mažiausiai sveria ir kurios duonos kaina didžiausia.
- Papildykite klasę `Duona` kintamuoju, skirtu duonos kepaliuko užimamam ant lentynos plotui saugoti. Sukurkite klasę `Kepykla`, kuri turėtų kintamajį kepyklos pavadinimui saugoti ir 3 kintamuosius (kiekvienai duonos miltų rūšiai), skirtus saugoti per pamainą reikalingam iškepti duonos kepaliukų kiekui. Suskaičiuokite, kiek ploto lentynų kiekvienos rūšies duonai sudėti reikės.
- Papildykite klasę `Kepykla` kintamuoju, kuriame būtų saugoma informacija apie tai, kiek kg duonos kepyklos automobilis gali vežti. Kiek kartų reikės važiuoti kepyklos automobiliui, kad išvežti iš parduotuves visą per pamainą iškeptą duoną?

**U2–24. Vandens telkiniai.**

- Sukurkite klasę `VandensTelkinys`, kuri turėtų kintamuosius vandens telkinio gylį, vandens telkinio tūriui ir pavadinimui saugoti. Atliekami trijų skirtinų vandens telkiniių tyrimai. Rasti mažiausią telkinio gylį ir bendrą visų telkiniių vandens tūri.
- Papildykite klasę `VandensTelkinys` kintamuoju, skirtu vandens telkinio plotui saugoti. Sukurkite klasę `Vartojimas`, kuri turėtų kintamuosius vieno žmogaus per parą suvartojoamo vandens kiekui ir žmonių, vartojančių vandenį, skaičiui saugoti. Ar visus telkinius atskirai gali vartoti tie žmonės?
- Papildykite klasę `Vartojimas` metodu `Dėti()`, kuris leistų keisti vandenį vartojančių žmonių skaičių ir vieno žmogaus per parą suvartojoamą vandens kiekį. Ar vandenį vartojančių žmonių skaičių padidinus dvigubai, jiems užteks sekliausio telkinio vandens vienai parai? Ar užteks didžiausio tūrio telkinio vandens trims paroms, jei vieno žmogaus per parą suvartojoamas vandens kiekis padvigubėja?

**3. Objektų rinkinys**

Susipažinsite su:

- masyvo struktūra ir masyvo tipo kintamuoju, kuriame bus saugomas objektų rinkinys;
- tekstiniuose failuose, kuriuose saugomi objektų duomenys ir į kuriuos surašomi skaičiavimų rezultatai;
- sumos, kiekio, aritmetinio vidurkio skaičiavimo algoritmai;
- didžiausios (mažiausios) reikšmės paieškos algoritmu;
- dviejų objektų rinkinių apjungimo į vieną algoritmu;

**3.1. Pažintis su objektų rinkiniu**

- Masyvas objektų rinkiniui saugoti.
- Sumos, kiekio ir aritmetinio vidurkio skaičiavimo algoritmu taikymas.
- Duomenų skaitymas iš tekstinio failo.

**Užduotis.** Dviratis.

Dviračių nuomos punkte yra įvairių modelių dviračiai. Žinomi kiekvieno dviračio pagaminimo metai ir kaina.

Parašykite programą, kuri:

- suskaičiuotų, kiek yra tinkamų naudoti dviračių (ne senesni kaip 10 metų) ir kokia jų piniginė vertė;
- kiek yra netinkamų naudoti dviračių ir kokia jų piniginė vertė;
- koks vidutinis tinkamų naudoti ir jau netinkamų naudoti dviračių amžius.

**Pradiniai duomenys ir rezultatai.**

| Pradiniai duomenys | Paaiškinimai                       |
|--------------------|------------------------------------|
| 15                 | Kritinis dviračio naudojimo laikas |
| 5                  | Dviračių skaičius                  |
| 1935 1259.58       | Dviračio pagaminimo metai ir kaina |
| 1988 300.45        | Dviračio pagaminimo metai ir kaina |
| 1995 200.46        | Dviračio pagaminimo metai ir kaina |
| 2008 985.25        | Dviračio pagaminimo metai ir kaina |
| 2012 3589.45       | Dviračio pagaminimo metai ir kaina |

| Skaičiavimų rezultatai                       |           |
|----------------------------------------------|-----------|
| Tinkami naudoti:                             | 2 4574.7  |
| Netinkami naudoti:                           | 3 1760.49 |
| Tinkamų naudoti dviračių vidutinis amžius:   | 2         |
| Netinkamų naudoti dviračių vidutinis amžius: | 39.33     |

**Programos kūrimo eiga.**

- Sukuriama klasė dviračio duomenims saugoti.
- Parašomi klasės metodai, skirti duomenims dėti ir imti.
- Paruošiamas duomenų failas.
- Paskelbiamas objektų rinkinys.
- Parašoma duomenų skaitymo iš failo funkcija, kuri suteikia rinkinio objektams reikšmes.
- Parašoma nurodyto amžiaus intervalo dviračių kiekio ir jų piniginės vertės skaičiavimo funkcija.
- Parašoma nurodyto amžiaus intervalo dviračių amžiaus vidurkio skaičiavimo funkcija.

**1 Pirmas žingsnis.**

- Sukurkite klasę, skirtą dviračio duomenims saugoti. Jos aprašą rašykite virš `main()` funkcijos.

```
//-----
class Dviratis
{
    private:
        int metai;           // pagaminimo metai
        double kaina;        // piniginė vertė
    public:
```

```

void Dėti(int metai, double kaina);
int ImtiMetus() { return metai; }
double ImtiKaina() { return kaina; }
};

//-----
• Parašykite metodą Dėti():

// Suteikia klasės kintamiesiems naujas reikšmes,
// nurodomas parametrais metai ir kaina
void Dviratis::Dėti(int metai, double kaina)
{
    Dviratis::metai = metai;
    Dviratis::kaina = kaina;
};

//-----
• Patikrinkite, ar klasė aprašyta teisingai, ar teisingai dirba sukurti metodai. Tam parašykite main() funkcijoje darbo su klasės objektu imitaciją:

int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Dviratis d;
    d.Dėti(2012, 25.26);
    cout << d.ImtiMetus() << " " << d.ImtiKaina() << endl;
    cout << "Programa baigė darbą\n";
    return 0;
};

//-----
• Įvykdykite programą. Ekrane turėtumėte matyti dviračio duomenis:
2012 25.26
Programa baigė darbą

```

### ¶ Antras žingsnis.

- Paruoškite duomenų failą.
  - Parašykite rinkinio (masyvo) maksimalų galimą dydį apibrėžiančią konstantą:
- ```
const int Cn = 100;
```
- Parašykite duomenų failo vardą, kuris nurodomas konstanta:
- ```
const char CFd[] = "Duom.txt";
```
- Pašalinkite darbo su objektu d veiksmus.
  - Objektą d pakeiskite objektų rinkiniu D[Cn], kur Cn – konstanta, reiškianti didžiausią galimą turėti dviračių skaičių.
  - Parašykite du sveikojo tipo kintamuosius, skirtus rinkinio duomenų skaičiui ir dviračio naudojimo kritinio amžiaus reikšmėms saugoti.
  - Parašykite duomenų skaitymo iš failo funkciją:
- ```

//-----
// Skaitomi duomenys iš failo, kurio vardas nurodomas konstanta CFd
// D - objektų rinkinis dviračių duomenims saugoti
// n - dviračių skaičius
// m - kritinis dviračių naudojimo amžius
void Skaityti(Dviratis D[], int & n, int & m)
{
    int metai; double kaina;
    ifstream fd(CFd);
    fd >> m >> n;
    for (int i = 0; i < n; i++) {
        fd >> metai >> kaina;
        D[i].Dėti(metai, kaina);
    }
    fd.close();
};

//-----
```

Programavimo įvadas

```

}
//-----
• Papildykite main() funkciją kreipiniu į parašytą funkciją.

int main()
{
    setlocale(LC_ALL, "Lithuanian");
    cout << "***** Pradžia *****\n";
    Dviratis D[Cn]; // dviračių duomenys - objektais
    int n; // dviračių skaičius
    int am; // dviračio tinkamumo naudoti kritinis amžius
    Skaityti(D, n, m); // iš failo skaitomi dviračių duomenys
    cout << n << endl; // kontrolinis spausdinimas
    cout << "Programa baigė darbą\n";
    return 0;
};

//-----
• Įvykdykite programą. Jeigu ekrane matote skaičių 5, tai reiškia, kad galbūt duomenys teisingai perskaityti. Norėdami įsitikinti, ar taip tikrai yra, tikslinga kontrolinį spausdinimą papildyti sakiniu, kurie spausdintų ekrane visus dviračių rinkinio duomenis:
for (int i = 0; i < n; i++)
    cout << D[i].ImtiMetus() << " " << D[i].ImtiKaina() << endl;
• Įvykdykite programą. Vėliau kontrolinio duomenų spausdinimo sakiniai nebus reikalingi, todėl juos bus galima pašalinti.

¶ Trečias žingsnis.
• Parašykite funkciją, skaičiuojančią nurodyto amžiaus intervalo dviračių [amPr, amPb] kiekį ir jų piniginę vertę. Tokia funkcija universali, nes galima atlkti veiksmus skirtiniams amžių intervalams.

//-----
// D - dviračių duomenys
// n - dviračių skaičius
// amPr - dviračių priešinos amžiaus intervalo pradžia
// amPb - dviračių priešinos amžiaus intervalo pabaiga
// metai - metai, kurių atžvilgiu skaičiuojamas amžius
// kiek - dviračių skaičius duotame amžiaus intervale
// suma - duoto amžiaus intervalo dviračių piniginė vertė
void Pinigai(Dviratis D[], int n, int amPr, int amPb, int metai,
              int & kiek, double & suma)
{
    kiek = 0;
    suma = 0.0;
    int amžius;
    for (int i = 0; i < n; i++) {
        amžius = metai - D[i].ImtiMetus();
        if ((amPr <= amžius) && (amžius <= amPb))
        {
            suma += D[i].ImtiKaina();
            kiek++;
        }
    }
};

//-----
• Papildykite main() funkciją kreipiniu į funkciją Pinigai() ir gautų rezultatų spausdinimo sakiniu:
int kiekTinka;
double sumaTinka;
Pinigai(D, n, 0, m, 2012, kiekTinka, sumaTinka);
cout << "Tinkami naudoti: " << kiekTinka << " " << sumaTinka << endl;
• Įvykdykite programą ir pasitikrinkite gautus rezultatus.
• Papildykite programą veiksmais dėl netinkamų naudoti dviračių:
```

```

int kiekNetinka;
double sumaTinka, sumaNetinka;
// Netinkami naudoti dviračiai, kurių amžius didelis,
// t.y. intervalė nuo m iki begalybės (pvz., 1000 metų)
Pinigai(D, n, m + 1, 1000, 2012, kiekNetinka, sumaNetinka);
cout << "Netinkami naudoti: " << kiekNetinka << " " << sumaNetinka << endl;
• Išvykdykite programą ir pasitikrinkite gautus rezultatus.

```

#### ¶ Ketvirtas žingsnis.

- Parašykite funkciją, skaičiuojančią nurodyto dviračių amžiaus intervalo vidurkį:

```

//-----
// D - dviračių duomenys
// n - dviračių skaičius
// metai - metai, kurių atžvilgiu skaičiuojamas dviračio amžius
// amPr - dviračių paieškos amžiaus intervalo pradžia
// amPb - dviračių paieškos amžiaus intervalo pabaiga
double Vidurkis(Dviratis D[], int n, int metai, int amPr, int amPb)
{
    int kiek = 0, suma = 0;
    int amžius;
    for (int i = 0; i < n; i++) {
        amžius = metai - D[i].ImtiMetus();
        if ((amPr <= amžius) && (amžius <= amPb))
        {
            suma += amžius;
            kiek++;
        }
    }
    if (kiek > 0) return (double) suma / kiek;
    return 0.0;
}
//-----

```

- Papildykite funkciją main() likusiais skaičiavimais:

```

cout << "Tinkamų naudoti dviračių vidutinis amžius: "
      << Vidurkis(D, n, 2012, 0, am) << endl;
cout << "Netinkamų naudoti dviračių vidutinis amžius: "
      << Vidurkis(D, n, 2012, am + 1, 1000) << endl;

```

#### Programos papildymas.

- Parašykite main() funkcijoje sakinius, kurie suskaičiuotų ir išspausdintų visų dviračių piniginę vertę ir visų dviračių vidutinį amžių.
- Parašykite main() funkcijoje sakinius, kurie suskaičiuotų ir išspausdintų dviračių, kurie buvo pagaminti nurodytais (pvz., 2012) metais, piniginę vertę ir visų dviračių vidutinį amžių.
- Parašykite main() funkcijoje sakinius, kurie suskaičiuotų ir išspausdintų dviračių, kurie buvo pagaminti 1000 metais (tokių nėra), piniginę vertę ir visų dviračių vidutinį amžių. Ar programa pateikia teisingus rezultatus?

#### Savarankiško darbo užduotis.

Žinoma, kiek kuris turistinės grupės narys turi pinigų (litai, centai). Duomenys tekstiniame faile. Parašykite programą, kuri suskaičiuotų, kiek grupė turi pinigų iš viso ir kiek vidutiniškai jų tenka kiekvienam nariui. Kiekvienas narys bendroms grupės išlaidoms skiria ketvirtadalį turimų pinigų. Kiek pinigų bus iš viso surinkta bendroms grupės išlaidoms?

### 3.2. Du objektų rinkiniai

- Duomenys dviejuose tekstiniuose failuose.
- Veiksmai su du objektų rinkiniais.
- Objektų rinkinių charakteristikų radimas.

#### Užduotis. Dviratis.

Dviejuose dviračių nuomas punktuose yra įvairių modelių dviračiai. Žinomi kiekvieno dviračio pagaminimo metai ir kaina.

Parašykite programą, kuri suskaičiuotų:

- kiek yra tinkamų naudoti dviračių (ne senesni kaip m metų) ir kokia jų piniginė vertė;
- kiek yra netinkamų naudoti dviračių ir kokia jų piniginė vertė;
- vidutinį tinkamų naudoti ir jau netinkamų naudoti dviračių amžių.

#### Pradiniai duomenys.

Pirmojo nuomas punkto duomenys	Paaiškinimai
Lék su vėjeliu	Nuomas punkto pavadinimas
15	Kritinis dviračio naudojimo laikas
5	Dviračių skaičius
1935 1259.58	Dviračio pagaminimo metai ir kaina
1988 300.45	Dviračio pagaminimo metai ir kaina
1995 200.46	Dviračio pagaminimo metai ir kaina
2008 985.25	Dviračio pagaminimo metai ir kaina
2012 3589.45	Dviračio pagaminimo metai ir kaina

Antrojo nuomas punkto duomenys	Paaiškinimai
Neskubėk ir būsi pirmas	Nuomas punkto pavadinimas
10	Kritinis dviračio naudojimo laikas
6	Dviračių skaičius
1955 1000.58	Dviračio pagaminimo metai ir kaina
1990 345.45	Dviračio pagaminimo metai ir kaina
1995 500.46	Dviračio pagaminimo metai ir kaina
2010 900.25	Dviračio pagaminimo metai ir kaina
2012 3589.45	Dviračio pagaminimo metai ir kaina
2012 1500.99	Dviračio pagaminimo metai ir kaina

#### Programos kūrimo eiga.

- Nukopijuojama ankstesnio pratimo programa.
- Paruošiamas antras duomenų failas.
- Programa papildoma dviem objektų rinkiniais ir veiksmais su jais.

#### ¶ Pirmas žingsnis.

- Nukopijuokite ankstesnio pratimo programą.
- Papildykite programą antru duomenų rinkiniu.
- Duomenų rinkinio pavadinimą D pakeiskite pavadinimu D1 bei kitus kintamuju pavadinimus. Papildykite programą kintamaisiais, skirtais nuomas punkto pavadinimui saugoti ir antruoju objektų rinkiniu:

```

// Pirmojo dviračių nuomas punkto
Dviratis D1[Cn]; // dviračių duomenys
int n1; // dviračių skaičius
int aml; // dviračio tinkamumo naudoti kritinis amžius
string pav1; // nuomas punkto pavadinimas
int kiekTinkal, kiekNetinkal;
double sumaTinkal, sumaNetinkal;
// Antrojo dviračių nuomas punkto
Dviratis D2[Cn]; // dviračių duomenys
int n2; // dviračių skaičius
int am2; // dviračio tinkamumo naudoti kritinis amžius
string pav2; // nuomas punkto pavadinimas
int kiekTinka2, kiekNetinka2;
double sumaTinka2, sumaNetinka2;

```

- Papildykite duomenų skaitymo funkciją parametrais, skirtais failo vardui perduoti ir grąžinti perskaitytam nuomas punkto pavadinimui:
- ```

//-----
// Skaitomi duomenys iš failo, kurio vardas nurodomas parametru Fv
// D - objektų rinkinys dviračių duomenims saugoti

```

```
// n - dviračių skaičius
// m - kritinis dviračių naudojimo amžius
// pav - nuomas punkto pavadinimas
void Skaityti(const char Fv[], Dviratis D[], int & n, int & am, string & pav)
{
    int metai; double kaina;
    ifstream fd(Fv);
    getline(fd, pav);
    fd >> am >> n;
    for (int i = 0; i < n; i++) {
        fd >> metai >> kaina;
        D[i].Dėti(metai, kaina);
    }
    fd.close();
}
-----
```

- Pakeiskite ir papildykite failų vardų konstantų sąrašą:

```
const char CFd1[] = "Duom1.txt";
const char CFd2[] = "Duom2.txt";
```

- Parašykite kreipinį į duomenų skaitymo funkciją pirmo nuomas punkto duomenims skaityti:  
Skaityti(CFd1, D1, n1, aml, pav1);
- Patikrinkite, ar pirmojo dviračių nuomas punkto duomenys perskaityti teisingai.

#### ¶ Antras žingsnis.

- Papildykite programą sakiniais, skirtais antrojo nuomas punkto duomenims skaityti. Patikrinkite, ar antrojo dviračių nuomas punkto duomenys perskaityti teisingai.

#### ¶ Trečias žingsnis.

- Papildykite programą sakiniais, skaičiuojančiais, kiek abiejuose nuomas punktuose yra tinkamų naudoti dviračių ir kokia jų piniginė vertė bei vidutinis amžius. Spausdinkite gautus rezultatus.
- Patikrinkite, ar gauti rezultatai teisingi.
- Papildykite programą sakiniais, skaičiuojančiais, kiek abiejuose nuomas punktuose yra netinkamų naudoti dviračių ir kokia jų piniginė vertė bei vidutinis amžius. Spausdinkite gautus rezultatus.
- Patikrinkite, ar gauti rezultatai teisingi.

#### ¶ Ketvirtas žingsnis.

- Parašykite sakinius, kuriais nustatoma, kuriame punkte yra daugiau tinkamų naudoti dviračių.
- Parašykite sakinius, kuriais nustatoma, kurio punkto turimų dviračių piniginė vertė yra didesnė.

#### Savarankiško darbo užduotis.

Barboros turimos valiutos surašyti faile B.txt, o Anupro – faile A.txt. Vienoje eilutėje – vienos šalies valiutos duomenys: pinigai (pvz., doleriai ir centai) ir kursas, perskaičiuojant į litus ir centus. Parašykite programą, kuri suskaičiuotų, kiek pinigų litais ir centais turi Barbora ir kiek Anupras atskirai ir kiek abu bendrai.

### 3.3. Naujo objektų rinkinio formavimas

- Didžiausios (mažiausios) reikšmės paieška.
- Naujo objektų rinkinio formavimas.

#### Užduotis. Dviratis.

Dvieluose dviračių nuomas punktuose yra įvairių modelių dviračiai. Žinomas kiekvieno dviračio modelis, kiekis, pagaminimo metai ir kaina.

Parašykite programą, kuri:

- surastų, kuriame dviračių nuomas punkte yra seniausias dviratis;
- surastų, kuriame dviračių nuomas punkte yra brangiausias dviratis ir kokia jo piniginė vertė;
- suformuotų dviračių modelių, kurie yra tuo pačiu metu abiejuose nuomas punktuose, rinkinį.

#### Pradiniai duomenys.

| Pirmaus nuomas punkto duomenys | Paaiškinimai                                        |                                                     |  |
|--------------------------------|-----------------------------------------------------|-----------------------------------------------------|--|
| Lék su vėjeliu                 | Nuomas punkto pavadinimas                           | Irašų skaičius                                      |  |
| 5                              | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| ORION 2 1935 1259.58           | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| POLARIS 1 1988 300.45          | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| INDUSTRIERAD 10 1995 200.46    | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| Bauer 25 2008 985.25           | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| HERREN 15 2012 3589.45         | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |

| Antrojo nuomas punkto duomenys | Paaiškinimai                                        |                                                     |  |
|--------------------------------|-----------------------------------------------------|-----------------------------------------------------|--|
| Neskubék ir būsi pirmas        | Nuomas punkto pavadinimas                           | Irašų skaičius                                      |  |
| 6                              | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| VOLTA 1 1955 1000.58           | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| POLARIS 2 1990 345.45          | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| BANANA 3 1995 500.46           | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| INDUSTRIERAD 10 2010 900.25    | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| ORION 15 2012 3589.45          | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |
| POLARIS 20 2012 1500.99        | Dviračio modelis, kiekis, pagaminimo metai ir kaina | Dviračio modelis, kiekis, pagaminimo metai ir kaina |  |

#### Programos kūrimo eiga.

- Ankstesnio pratimo programos teksto kopija modifikuojama darbui su dviračių rinkiniai, kai žinomi dviračių modelių pavadinimai.
- Parašoma objektų rinkinio duomenims spausdinti failo funkcija.
- Parašoma funkcija seniausiam dviračiui rasti nurodytame objektų rinkinyje.
- Parašoma funkcija brangiausiam dviračiui rasti nurodytame objektų rinkinyje.
- Parašoma funkcija, skirta rasti nurodyto modelio dviračiui nurodytame objektų rinkinyje.
- Parašomos funkcijos, skirtos formuoti bendram objektų sąrašui iš turimų dviejų.

#### ¶ Pirmas žingsnis.

- Paruoškite duomenų failus, suteikdami jiems vardus Nuoma1.txt ir Nuoma2.txt.
- Papildykite duomenų failus duomenimis pagal pateiktą pavyzdį.
- Nukopijuokite ankstesnio pratimo programos tekštą ir išsaugokite nauju vardu.
- Pašalinkite visas funkcijas, išskyrus duomenų skaitymo iš failo. Iš funkcijos main() pašalinkite kreipinius į tas funkcijas.
- Papildykite klasę Dviratis kintamaisiais, skirtais modelio pavadinimui ir dviračių kiekiui saugoti:

```
string pav;
int kiek;
• Klasės metodą Dėti() papildykite parametrais, skirtais dviračio modelio vardui ir kiekiui dėti.
Metodo antraštė:
void Dėti(string pav, int kiek, int metai, double kaina);
• Papildykite klasę metodais, grąžinančiais dviračio modelio pavadinimą ir jų kiekį:
string ImtiPav() { return pav; }
int ImtiKiek() { return kiek; }
• Įvykdykite programą. Jeigu nėra klaidų, programa sėkmingai baigia darbą nieko neskaičiuodama.
```

#### ¶ Antras žingsnis.

- Modifikuokite duomenų skaitymo iš failo funkciją:

```
-----
// Skaitomi duomenys iš failo, kurio vardas nurodomas parametru Fd
// D - objektų rinkinys dviračių duomenims saugoti
// n - dviračių skaičius
// pav - nuomas punkto pavadinimas
void Skaityti(const char Fd[], Dviratis D[], int & n, string & pav)
{
    string eil; int kiek; int metai; double kaina;
    ifstream fd(Fd);
```

```

getline(fd, pav);
fd >> n;
for (int i = 0; i < n; i++){
    fd >> eil >> kiek >> metai >> kaina;
    D[i].Dėti(eil, kiek, metai, kaina);
    fd.ignore(80, '\n');
}
fd.close();
}

-----
• Parašykite main() funkcijoje duomenų skaitymo sakinus, patikrinkite, ar gerai skaitomi duomenys.

const char CFd1[] = "Nuomal.txt"; // pirmo nuomos punkto dviračių sąrašas
const char CFd2[] = "Nuoma2.txt"; // antro nuomos punkto dviračių sąrašas
-----

int main()
{
    // Pirmojo dviračių nuomos punkto
    Dviratis D1[Cn]; // dviračių duomenys
    int n1; // dviračių skaičius
    string pav1; // nuomos punkto pavadinimas
    // Antrojo dviračių nuomos punkto
    Dviratis D2[Cn]; // dviračių duomenys
    int n2; // dviračių skaičius
    string pav2; // nuomos punkto pavadinimas

    Skaityti(CFd1, D1, n1, pav1);
    Skaityti(CFd2, D2, n2, pav2);
    return 0;
}

```

### ¶ Trečias žingsnis.

- Parašykite objektų rinkinio spausdinimo failo funkciją:

```

// Objektų rinkinio D duomenys spausdinami sraute fr
// n - objektų skaičius rinkinyje
// pav - dokumento pavadinimas
void Spausdinti(ofstream & fr, Dviratis D[], int n, string pav)
{
    fr << pav << endl;
    fr << "-----\n";
    fr << " Modelis Kiekis Data Kaina \n";
    fr << "-----\n";
    for (int i = 0; i < n; i++)
        fr << setw(15) << left << D[i].ImtiPav()
            << setw(4) << right << D[i].ImtiKiek()
            << setw(8) << D[i].ImtiMetus()
            << setw(10) << right << D[i].ImtiKaina() << endl;
    fr << "-----\n";
    fr << endl;
}

```

- Papildykitė failų vardų konstantų sąrašą rezultatų failo vardo CFrez konstanta.
- Parašykite main() funkcijoje rezultatų failo parengimo spausdinimui sakinį, o gale – failo uždarymo sakinį:

```

ofstream fr(CFrez);
fr.close();

```

- Papildykitė main() funkciją krepiniai į funkciją Spausdinti(), kuriais būtų išspausdinami nuomos punktų duomenys. Patikrinkite, ar rezultatų failo gavote tokias lenteles:

### Lėk su vėjeliu

| Modelis      | Kiekis | Data | Kaina   |
|--------------|--------|------|---------|
| ORION        | 2      | 1935 | 1259.58 |
| POLARIS      | 1      | 1988 | 300.45  |
| INDUSTRIERAD | 10     | 1995 | 200.46  |
| Bauer        | 25     | 2008 | 985.25  |
| HERREN       | 15     | 2012 | 3589.45 |

### Neskubėk ir būsi pirmas

| Modelis      | Kiekis | Data | Kaina   |
|--------------|--------|------|---------|
| VOLTA        | 1      | 1955 | 1000.58 |
| POLARIS      | 2      | 1990 | 345.45  |
| BANANA       | 3      | 1995 | 500.46  |
| INDUSTRIERAD | 10     | 2010 | 900.25  |
| ORION        | 15     | 2012 | 3589.45 |
| POLARIS      | 20     | 2012 | 1500.99 |

### ¶ Ketvirtas žingsnis.

- Sukurkite funkciją, kuri duotame objektų rinkinyje surastų seniausio dviračio indeksą:

```

// Gražina dviračio, kurio metų skaičius yra mažiausias, indeksą.
// D - objektų rinkinys
// n - objektų skaičius rinkinyje
int Seniausias(Dviratis D[], int n)
{
    int k = 0;
    for (int i = 0; i < n; i++)
        if (D[i].ImtiMetus() < D[k].ImtiMetus())
            k = i;
    return k;
}

```

- Papildykitė main() funkciją veiksmais, kuriais būtų surandamas nuomos punktas, kuriame yra seniausias dviratis:

```

if (D1[Seniausias(D1, n1)].ImtiMetus() < D2[Seniausias(D2, n2)].ImtiMetus())
    cout << "Seniausias dviratis yra nuomos punkte " << pav1 << endl;
else
    cout << "Seniausias dviratis yra nuomos punkte " << pav2 << endl;

```

### Galima parašyti kitaip:

```

int ind1 = Seniausias(D1, n1);
int ind2 = Seniausias(D2, n2);
if (D1[ind1].ImtiMetus() < D2[ind2].ImtiMetus())
    cout << "Seniausias dviratis yra nuomos punkte " << pav1 << endl;
else
    cout << "Seniausias dviratis yra nuomos punkte " << pav2 << endl;

```

- Patikrinkite, ar programa dirba teisingai.
- Pakeiskite duomenis taip, kad seniausias dviratis būtų kitame nuomos punkte. Patikrinkite, ar programa dirba teisingai?
- Pakeiskite duomenis taip, kad abiejuose punktuose yra vienodi seniausi dviračiai. Kurio punkto pavadinimą matote? Patikslinkite programą taip, kad šiuo atveju programa ekrane rodytų abiejų punktų pavadinimus.
- Patikrinkite, kaip programa dirba, kai punkte / punktuose yra keli seniausi dviračiai.

**① Penktas žingsnis.**

- Sukurkite funkciją, kuri nustatyti, ar objektų sąraše yra nurodyto modelio dviratis:

```
-----
// Gražina surasto dviračio masyve indeksą arba -1, jeigu dviračio masyve nėra
// D - objektų rinkinys
// n - objektų skaičius rinkinyje
// pav - ieškomo modelio pavadinimas
int YraModelis(Dviratis D[], int n, string pav)
{
    for (int i = 0; i < n; i++)
        if (D[i].ImtiPav() == pav) return i;
    return -1;
}
-----
```

- Papildykite klasę metodu

```
void PapildytiKiek(int k) { kiek += k; }
```

- Sukurkite funkciją, kuri dviračių modelių sąrašą papildytų dviračiais iš duoto objektų rinkinio:

```
-----
// Objektų rinkinių Dr papildo duomenimis iš objektų rinkinio D
// Jeigu objektų rinkinio D tokio pat modelio dviratis yra objektų rinkinyje Dr,
// tuomet didinamas kiekis, kitaip - papildomas nauju objektu
// n - objektų skaičius rinkinyje D
// nr - objektų skaičius rinkinyje Dr
void Formuoti(Dviratis D[], int n, Dviratis Dr[], int & nr)
{
    int k;
    for (int i = 0; i < n; i++) {
        k = YraModelis(Dr, nr, D[i].ImtiPav()); //
        if (k >= 0)
            Dr[k].PapildytiKiek(D[i].ImtiKiek()); // didinamas kiekis
        else {
            Dr[nr] = D[i]; // papildomas rinkinys
            nr++;
        }
    }
}
-----
```

- Papildykite main() funkciją veiksmais, kuriais formuojamas abiejuose nuomos punktuose esančiu dviračių modelių rinkinys.

```
Dviratis Dr[Cn]; int nr;
nr = 0;
Formuoti(D1, n1, dr, nr);
Formuoti(D2, n2, dr, nr);
Spausdinti(fr, Dr, nr, "Modelių sąrašas");
```

- Patikrinkite, ar rezultatų faile išspausdinta teisinga modelių lentelė:

Modelių sąrašas

| Modelis      | Kiekis | Data | Kaina   |
|--------------|--------|------|---------|
| ORION        | 17     | 1935 | 1259.58 |
| POLARIS      | 23     | 1988 | 300.45  |
| INDUSTRIERAD | 20     | 1995 | 200.46  |
| Bauer        | 25     | 2008 | 985.25  |
| HERREN       | 15     | 2012 | 3589.45 |
| VOLTA        | 1      | 1955 | 1000.58 |
| BANANA       | 3      | 1995 | 500.46  |

**Savarankiško darbo užduotis.**

- Sukurkite funkciją, kuri surastų duotame objektų rinkinyje brangiausią dvirati.
- Papildykite programą veiksmais, kuriais būtų randama, kuriame dviračių nuomos punkte yra brangiausias dviratis ir kokia jo piniginė vertė.

**3.4. Keletas pavadinimo žodžių**

- Objektų rinkinys.
- Reikšmių įvedimas iš failo.
- Reikšmių spausdinimas į failą lentele.
- Funkcijos.

**Užduotis.** Firmos automobilai.

Failo Duomenys.txt pirmoje eilutėje yra vienos UAB automobilių kiekis, kitose eilutėse – automobilių duomenys: pavadinimas, degalų tipas, degalų sąnaudos 100 km.

Sukurkite klasę automobilio duomenims saugoti. I šios klasės objektų masyvą perskaitykite duomenis iš failo Duomenys.txt. Masyvo objektų reikšmes spausdinkite lentele į failą Rezultatai.txt.

**Pradiniai duomenys ir rezultatai.**

| Pradiniai duomenys failo Duomenys.txt |                  |            |     |
|---------------------------------------|------------------|------------|-----|
| 8                                     | Škoda Fabia,     | benzinis,  | 7.1 |
|                                       | Opel Meriva,     | benzinis,  | 7.5 |
|                                       | Volkswagen Golf, | dyzelinis, | 6.3 |
|                                       | Opel Astra,      | dyzelinis, | 6.1 |
|                                       | Toyota Corolla,  | benzinis,  | 6.8 |
|                                       | Toyota Yaris,    | benzinis,  | 5.9 |
|                                       | Peugeot 207,     | dyzelinis, | 5.5 |
|                                       | Škoda Octavia,   | dyzelinis, | 7.3 |

| Rezultatai failo Rezultatai.txt |           |                     |  |
|---------------------------------|-----------|---------------------|--|
| Automobilių skaičius: 8         |           |                     |  |
| Automobilių sąrašas:            |           |                     |  |
| -----                           |           |                     |  |
| Pavadinimas                     | Degalai   | Sąnaudos (1/100 km) |  |
| Škoda Fabia                     | benzinis  | 7.10                |  |
| Opel Meriva                     | benzinis  | 7.50                |  |
| Volkswagen Golf                 | dyzelinis | 6.30                |  |
| Opel Astra                      | dyzelinis | 6.10                |  |
| Toyota Corolla                  | benzinis  | 6.80                |  |
| Toyota Yaris                    | benzinis  | 5.90                |  |
| Peugeot 207                     | dyzelinis | 5.50                |  |
| Škoda Octavia                   | dyzelinis | 7.30                |  |

**Programos kūrimo eiga.**

- Sukuriama klasė vieno automobilio duomenims saugoti. Parašomi klasės sasajos metodai.
- Pagrindinėje funkcijoje main() skelbiamas automobilio klasės objektų masyvas.
- Parašoma funkcija duomenims iš failo įvesti į objektų masyvą.
- Paruošiamas duomenų failas "Duomenys.txt".
- Parašoma funkcija objektų masyvo reikšmėms spausdinti lentele į failą.
- Pagrindinė funkcija papildoma kreipiniais į parašytas funkcijas.

**② Pirmas žingsnis.**

- Sukurkite klasę Auto ir jos sasajos metodus. Imti() ir Deti()

```
-----
class Auto
{
```

```

private:
    string pav;           // automobilio pavadinimas
    string degalai;       // degalų tipas
    double sanaudos;      // kuro sanaudos 100 km
public:
    void Dėti(string p, string d, double s)
    {
        pav = p;           // automobilio pavadinimas
        degalai = d;        // automobilio pavadinimas
        sanaudos = s;        // kuro sanaudos 100 km
    }
    string ImtiPav() { return pav; }
    string ImtiDegalus() { return degalai; }
    double ImtiSanaudas() { return sanaudos; }
};

//-----

```

- Pagrindinėje funkcijoje aprašykite klasės Auto objektų rinkinį A ir kintamąjį, kuriame bus saugomas UAB automobilių kiekis.

```

//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Auto A[100];           // automobilių duomenys
    int na;                 // automobilių kiekis
    cout << "Programa baigė darbą\n";
    return 0;
}
//-----

```

### ¶ Antras žingsnis.

- Prieš pagrindinę funkciją parašykite duomenų įvedimo funkcijos prototipą.

```

//-----
void Įvesti(string fv, Auto A[], int & kiek);
    • Po pagrindine funkcija parašykite duomenų įvedimo funkcijos tekštą.

// Iš failo fv įveda duomenis į objektų masyvą A(kiek).
void Įvesti(string fv, Auto A[], int & kiek)
{
    string pav, degalai;
    double sanaudos;
    ifstream fd(fv.c_str());
    fd >> kiek; fd.ignore();
    cout << "Pavadinimai:\n";
    for (int i = 0; i < kiek; i++) {
        getline(fd, pav, ','); fd >> ws;
        cout << pav << endl;
        getline(fd, degalai, ',');
        fd >> sanaudos;
        A[i].Dėti(pav, degalai, sanaudos);
        fd.ignore();
    }
    fd.close();
}
//-----

```

Naudodamiesi pavyzdžiu paruoškite duomenų failą Duomenys.txt.

### ¶ Trečias žingsnis.

- Prieš pagrindinę funkciją parašykite objektų masyvo reikšmių spausdinimo funkcijos prototipą.

```
void Spausdinti(string fv, Auto A[], int kiek);
```

- Po pagrindine funkcija parašykite objektų masyvo reikšmių spausdinimo funkcijos tekštą.

```
//-----
```

```

// Objektų masyvo A(kiek) reikšmes spausdina lentelė į failą fv
void Spausdinti(string fv, Auto A[], int kiek)
{
    ofstream fr(fv.c_str(), ios::app);
    fr.setf(ios::fixed); fr.setf(ios::left);
    fr << "Automobilių skaičius: " << kiek << endl;
    fr << "Automobilių sąrašas:\n";
    fr << "-----\n";
    fr << "| Pavadinimas | Degalai | Sanaudos (1/100 km)\n";
    fr << "-----\n";
    for (int i = 0; i < kiek; i++) {
        fr << "| " << setw(15) << A[i].ImtiPav() << " | " << setw(10)
            << A[i].ImtiDegalus() << " | " << setprecision(2)
            << A[i].ImtiSanaudas() << " | " << endl;
    }
    fr << "-----\n";
    fr.close();
}
//-----

```

### ¶ Ketvirtas žingsnis.

- Papildykitė pagrindinę funkciją rezultatų failo sukūrimo (išvalymo) veiksmais.

```
ofstream fr("Rezultatai.txt");
fr.close();
```

- Papildykitė pagrindinę funkciją kreipiniiais į įvedimo ir spausdinimo funkcijas.

```
Įvesti("Duomenys.txt", A, na);
Spausdinti("Rezultatai.txt", A, na);
```

- Ivykdykite programą. Rezultatų failo "Rezultatai.txt" matysite:

Automobilių skaičius: 8

Automobilių sąrašas:

| Pavadinimas     | Degalai | Sanaudos (1/100 km) |
|-----------------|---------|---------------------|
| Škoda Fabia     | benzin  | 7.10                |
| Opel Meriva     | benzin  | 7.50                |
| Volkswagen Golf | dyzelin | 6.30                |
| Opel Astra      | dyzelin | 6.10                |
| Toyota Corolla  | benzin  | 6.80                |
| Toyota Yaris    | benzin  | 5.90                |
| Peugeot 207     | dyzelin | 5.50                |
| Škoda Octavia   | dyzelin | 7.30                |

### Programos papildymas.

Pakeiskite programą taip, kad automobilių sąrašo lentelėje būtų spausdinami eilučių numeriai.

### Savarankiško darbo užduotis.

Yra žinomi n (5 < n < 100) Lietuvos kelių duomenys (pavadinimas, ilgis, leistinas greitis). Surašykite šiuos duomenis į failą. Parašykite programą, kuri įvestų duomenis iš failo į objektų masyvą ir spausdintų lentelę į kitą failą.

### 3.5. Objektų charakteristikų radimas

- Sumos, sandaugos, kiekio skaičiavimas.

#### Užduotis.

Failo "Duomenys.txt" pirmoje eilutėje yra vienos UAB automobilių kiekis, kitose eilutėse yra automobilių duomenys: pavadinimas, degalų tipas, degalų sanaudos 100 km.

Sukurkite klasę automobilio duomenims saugoti. I šios klasės objektų masyvą perskaitykite duomenis iš failo "Duomenys.txt".

Kiek degalų 100 km vidutiniškai sunaudoja vienas automobilis?

Duomenis ir rezultatus spausdinkite į failą "Rezultatai.txt".

## Pradiniai duomenys ir rezultatai.

| Pradiniai duomenys faile "Duomenys.txt" |            |     |
|-----------------------------------------|------------|-----|
| 8                                       |            |     |
| Škoda Fabia,                            | benzinas,  | 7.1 |
| Opel Meriva,                            | benzinas,  | 7.5 |
| Volkswagen Golf,                        | dyzelinas, | 6.3 |
| Opel Astra,                             | dyzelinas, | 6.1 |
| Toyota Corolla,                         | benzinas,  | 6.8 |
| Toyota Yaris,                           | benzinas,  | 5.9 |
| Peugeot 207,                            | dyzelinas, | 5.5 |
| Škoda Octavia,                          | dyzelinas, | 7.3 |

| Rezultatai faile "Rezultatai.txt" |           |                     |
|-----------------------------------|-----------|---------------------|
| Automobilių skaičius: 8           |           |                     |
| Automobilių sąrašas:              |           |                     |
| Pavadinimas                       | Degalai   | Sąnaudos (1/100 km) |
| Škoda Fabia                       | benzinas  | 7.10                |
| Opel Meriva                       | benzinas  | 7.50                |
| Volkswagen Golf                   | dyzelinas | 6.30                |
| Opel Astra                        | dyzelinas | 6.10                |
| Toyota Corolla                    | benzinas  | 6.80                |
| Toyota Yaris                      | benzinas  | 5.90                |
| Peugeot 207                       | dyzelinas | 5.50                |
| Škoda Octavia                     | dyzelinas | 7.30                |

Vidutinės degalų sąnaudos: 6.56 litro/100 km

### Programos kūrimo eiga.

- Ankstesnio pratimo programa kopijuojama į atskirą katalogą.
- Parašoma funkcija klasės Auto objektų masyvo lauko sąnaudos reikšmių vidurkiui skaičiuoti.
- Pagrindinė funkcija papildoma parašytos funkcijos grąžintos reikšmės spausdinimu.

### ¶ Pirmas žingsnis.

- Nukopijuokite ankstesnio pratimo programą kartu su klase Auto ir duomenų failu Duomenys.txt.

### ¶ Antras žingsnis.

- Prieš pagrindinę funkciją parašykite vidurkio skaičiavimo funkcijos prototipą.

```
double VidSąnaudos(Auto A[], int kiek);
//-----  
// Skaičiuoja ir gražina objektų masyvo A(kiek) lauko sąnaudos vidutinę reikšmę
double VidSąnaudos(Auto A[], int kiek)
{
    double sum = 0;
    for (int i=0; i<kiek; i++) {
        sum = sum + A[i].ImtiSąnaudas();
    }
    return sum / kiek;
} //-----
```

### ¶ Trečias žingsnis.

- Papildykite pagrindinę funkciją parašytos funkcijos grąžintos reikšmės spausdinimu.

```
fr.open("rezultatai.txt", ios::app);
fr.setf(ios::fixed); fr.setf(ios::left);
fr << "Vidutinės degalų sąnaudos: " << setprecision(2) << VidSąnaudos(A, na)
     << " litro/100 km" << endl;
fr.close();
```

- Ivykdykite programą. Rezultatų failą "Rezultatai.txt" matysite:

Automobilių skaičius: 8

Automobilių sąrašas:

| Pavadinimas     | Degalai   | Sąnaudos (1/100 km) |
|-----------------|-----------|---------------------|
| Škoda Fabia     | benzinas  | 7.10                |
| Opel Meriva     | benzinas  | 7.50                |
| Volkswagen Golf | dyzelinas | 6.30                |
| Opel Astra      | dyzelinas | 6.10                |
| Toyota Corolla  | benzinas  | 6.80                |
| Toyota Yaris    | benzinas  | 5.90                |
| Peugeot 207     | dyzelinas | 5.50                |
| Škoda Octavia   | dyzelinas | 7.30                |

Vidutinės degalų sąnaudos: 6.56 litro/100 km

### Programos papildymas.

Parašykite funkciją dyzelinį automobilių kiekiui rasti. Papildykite programą šio kieko radimui ir spausdinimui į rezultatų failą.

### Savarankiško darbo užduotis.

Modifikuokite paskutinio pratimo programą, kuri papildomai rastų:

- kiek degalų 100 km vidutiniškai sunaudoja vienas benziniinis automobilis;
- įvertintų situaciją, kad gali nebūti né vieno benziniinio automobilio.

## 3.6. Kontroliniai klausimai

- Kiek ir kokių kintamųjų reikia, norint aprašyti duomenų masyvą arba objektų rinkinį?
- Kuo skiriasi reikšmių sumos radimo algoritmas nuo aritmetinio vidurkio skaičiavimo algoritmo ir kas juose yra benda?
- Kokia turi būti pradinė kintamojo, skirto didžiausiai reikšmei masyve (rinkinyje) rasti, reikšmė?
- Kuo skiriasi mažiausios reikšmės radimo algoritmas nuo algoritmo, skirto rasti mažiausią reikšmę turintį objektą rinkinyje?
- Kaip surasti, kiek objektų rinkinyje yra objektų, turinčių vienodą didžiausią reikšmę?
- Duotas masyvo aprašas: double A[35]; Kuris masyvo elemento panaudojimo atveju yra teisingas?
  - double x = A[0]+3.5;
  - A[3.5] = 2.5;
  - A[0] = A[35]+2;
  - A[7] = "2.5";
- Duotas masyvo aprašas: double A[35]; Kintamajame S reikia sukaupti teigiamų masyvo A elementų sumą.  
Kokia turi būti kintamojo S pradinė reikšmė?
  - 0
  - 1
  - bet koks teigiamas skaičius
  - pradinės reikšmės priskirti nereikia
- Duotas masyvo aprašas: double A[35]; Reikia rasti masyvo A didžiausio elemento indeksą.  
Kokia turi būti kintamojo, kuriame saugosime ieškomą indeksą, pradinė reikšmė?
  - bet kurio elemento indeksas
  - bet kurio elemento reikšmė
  - bet koks teigiamas skaičius
  - bet koks neigiamas skaičius
- Sveikujų skaičių masyve A(n) yra teigiamų, neigiamų ir nulinii reikšmių. Didžiausios reikšmės paieškos algoritmas toks:
 

```
int did = A[0];
for (int i = 0; i < n; i++)
    if (A[i] > did) did = A[i];
```

Reikia surasti mažiausią teigiamą reikšmę.

Kokia turi būti paieškos rezultato kintamojo did pradinė reikšmė?

- a. bet kuri masyvo teigiamą reikšmę;
- b. bet kuri masyvo reikšmę, tačiau reikia palyginimo sąlygą papildyti tikrinimu, ar  $(A[i] > 0)$
- c. gali būti  $A[0]$ , tačiau reikia palyginimo sąlygą papildyti tikrinimu, ar  $(A[i] > 0)$
- d. bet kokia teigiamą reikšmę.

10. Ką matysime ekrane, atlikus tokius veiksbus:

```
int D[8] = {5, 4, -3, 8, 0, 0, 0, 0};  
int B[8] = {6, 5, -9, -4, 3, 0, 0, 0};  
cout << Suma(D, 0, 4, 2) << " " << Suma(B, 1, 5, 2) << endl;
```

- 2 1
- 14 1
- 2 0
- Programos fragmente yra klaidų.

### 3.7. Užduotys

#### U3–1. Sodas

- Sode auga daug skirtingų vaismedžių. Tekstiniame duomenų faile yra vaismedžio pavadinimas, jo aukštis ir amžius. Pirmoje failo eilutėje yra sodo savininko pavardė ir vardas. Sukurkite klasę Vaismedis, kuri turėtų kintamuosius pavadinimui, amžiu i ir aukščiu saugoti. Raskite, koks aukščiausio vaismedžio aukštis ir koks seniausio vaismedžio amžius.
- Papildykite programą veiksmais su dvieju sodais. Kiekvieno sodo duomenys saugomi atskiruose failuose. Kuriame sode auga aukščiausias vaismedis? Surašykite į atskirą rinkinį visus abiejų sodybų vaismedžius, kurių pavadinimas sutampa su aukščiausio vaismedžio pavadinimu

#### U3–2. Krepšinis

- Krepšinio mokykloje treniruotes lankančių sąrašas yra tekstiniame faile: būsimo krepšininko vardas ir pavardė, amžius ir ūgis. Pirmoje eilutėje yra krepšinio mokyklos pavadinimas. Sukurkite klasę Krepšininkas, kuri turėtų kintamuosius vardui su pavarde, amžiu bei ūgiu saugoti. Raskite, koks būsimų krepšininkų amžiaus vidurkis ir koks ūgio vidurkis.
- Papildykite programą veiksmais su dvieju krepšinio mokyklų duomenimis. Kiekvienos mokyklos duomenys saugomi atskiruose failuose. Kurioje mokykloje aukščiausias sportininkas? Surašykite į atskirą rinkinį visus abiejų mokyklų sportininkus, kurių ūgis didesnis už vidurkį.

#### U3–3. Studentas

- Gūdžiame miške gyvena pabaisos. Duomenys apie jas yra tekstiniame faile: pavadinimas, ragų skaičius, uodegų skaičius. Pirmoje eilutėje yra studento, kuris tyrinėja tas pabaisas, vardas ir pavardė. Antroje eilutėje yra miško pavadinimas. Sukurkite klasę Pabaisa, kuri turėtų kintamuosius pavadinimui, ragų ir uodegų kiekiu saugoti. Kiek iš viso miške yra ragų ir uodegų? Kiek ragų turi mažiausiai uodeguota pabaisa?
- Papildykite programą veiksmais su dvieju miškų duomenimis. Kiekvieno miško duomenys saugomi atskiruose failuose. Kurio studento miške yra pabaisa, turinti daugiausiai ragų? Surašykite į atskirą rinkinį visus abiejų miškų pabaisų, kurių ragų skaičius didesnis už turimų uodegų skaičių, duomenis.

#### U3–4. Monetos

- Monetų kolekcinko turimų monetų duomenys surašyti faile: monetos kilmės šalis, nominalas ir svoris. Pirmoje eilutėje yra kolekcinko vardas ir pavardė. Sukurkite klasę Moneta, kuri turėtų kintamuosius kilmės šalies pavadinimui, nominalui ir svorui saugoti. Kuri moneta sunkiausia ir kokia visų monetų bendra piniginė vertė?
- Papildykite programą veiksmais su dvieju kolekcijų duomenimis. Kiekvienos kolekcijos duomenys saugomi atskiruose failuose. Kurioje kolekcijoje yra didžiausio nominalo moneta? Kurių kolekcinkas turtingiausias? Surašykite į atskirą rinkinį visus abiejų kolekcijų monetų, kurių nominalas yra 1, duomenis.

#### U3–5. Automobilis

- Studentas analizuojia automobilių tinkamumą kelionėms. Faile pirmoje eilutėje yra studento vardas ir pavardė. Toliau yra duomenys apie automobilius: markė, keleivių, telpančių į automobilį, skaičius, ir kuro, sunaudojamo 100 km, kiekiui saugoti. Sukurkite klasę Automobilis, kuri turėtų

kintamuosius markei, keleivių, telpančių į automobilį, skaičiui ir kuro, sunaudojamo 100 km, kiekiui saugoti. Rasti, kiek kainuoja pigiausia kelionė (mažiausiai kuro sunaudoja) ir keli keleiviai telpa didžiausiam automobiliuje.

- Papildykite programą veiksmais su dvieju studentų automobilių rinkiniais. Kiekvieno studento automobilių duomenys saugomi atskiruose failuose. Kurio studento sąraše yra talpiausias automobilis? Surašykite į atskirą rinkinį visus abiejų studentų tyrinėtų automobilių, kurie sunaudoja 100 km kuro ne daugiau kaip talpiausias automobilis, kiekiui, duomenis.

#### U3–6. Knygos

- Studento perkamų knygų sąrašas yra faile: autorius, pavadinimas, kaina, puslapių skaičius. Pirmoje eilutėje yra studento vardas ir pavardė. Sukurkite klasę Knyga, kuri turėtų kintamuosius knygos pavadinimui, puslapių skaičiui ir knygos kainai saugoti. Kuri knyga brangiausia ir už kokią pinigų sumą studentas perka knygų?
- Papildykite programą veiksmais su dvieju studentų perkamomis knygomis. Kiekvieno studento perkamų knygų duomenys saugomi atskiruose failuose. Kurio studento sąraše yra brangiausia knyga? Surašykite į atskirą rinkinį abiejų studentų visų knygų, kurių puslapių skaičius neviršija surastos brangiausios knygos puslapių skaičių, duomenis.

#### U3–7. Salė

- Koncertų salei siūlomų kėdžių tipų sąrašas yra faile: pavadinimas, užimamas plotas (stačiakampio formos), kaina. Pirmoje eilutėje yra firmos, kuri siūlo kėdes, pavadinimas. Sukurkite klasę Kėdė, kuri turėtų kintamuosius pavadinimui, kainai ir užimamam plotui saugoti. Raskite, kurio tipo kėdė užima didžiausią plotą ir kurio tipo kėdė yra brangiausia.
- Papildykite programą veiksmais su dvieju firmų siūlomų kėdžių rinkiniais. Kiekvienos firmos kėdžių duomenys saugomi atskiruose failuose. Kuriame sąraše yra brangiausia kėdė? Surašykite į atskirą rinkinį abiejų sąrašų visus kėdžių, kurios yra vidutinio dydžio, duomenis. Vidutinė kėdė yra ta, kurios užimamas plotas skiriasi nuo turimų kėdžių užimamų plotų aritmetinio vidurkio ne daugiau, kaip 10 proc.

#### U3–8. Prekė

- Statomo namo sienų apšiltinimui siūlomų medžiagų sąrašas yra faile: pavadinimas,  $1 \text{ m}^2$  kaina, storis. Pirmoje eilutėje yra statybinių prekių parduotuvės, kuri siūlo prekes, pavadinimas. Sukurkite klasę Prekė, kuri turėtų kintamuosius pavadinimui, kainai ir storiiu saugoti. Raskite, kiek sąraše yra medžiagų, kurių kaina neviršija namo savininko nurodytos kainos ir storis yra ribose [a, b].
- Papildykite programą veiksmais su dvieju parduotuvių siūlomų medžiagų rinkiniais. Kiekvienos parduotuvės medžiagų duomenys saugomi atskiruose failuose. Kuriame sąraše yra brangiausia medžiaga? Surašykite į atskirą rinkinį visus abiejų sąrašų medžiagų, kurios tenkina namo savininko reikalavimus, duomenis.

#### U3–9. Lenta

- Sandėlyje esančių lentų tipų sąrašas yra faile: lento medžio pavadinimas, lento storis, ilgis, plotis ir kaina. Pirmoje eilutėje yra firmos, kuri prekiauja lentomis, pavadinimas. Sukurkite klasę Lenta, kuri turėtų kintamuosius lento duomenims saugoti. Raskite, kurio tipo lentoms bus išleista daugiausiai pinigų, jeigu koncertų salės grindų ilgis yra L, o plotis P.
- Papildykite programą veiksmais su dvieju firmų siūlomų lentų rinkiniais. Kiekvienos firmos lentų duomenys saugomi atskiruose failuose. Kuriame sąraše yra brangiausia lenta? Surašykite į atskirą rinkinį visus abiejų sąrašų lentų, kurios yra vidutinio dydžio, duomenis. Vidutinė lenta yra ta, kurios užimamas plotas skiriasi nuo turimų lentų užimamų plotų aritmetinio vidurkio ne daugiau, kaip  $\frac{P}{L}$  proc.

#### U3–10. Siuvykla

- Siuvyklos siuvamų kostiumų duomenys saugomi faile: modelio pavadinimas, medžiagos pavadinimas, pasiuvimui reikalingos medžiagos ilgis bei plotis ir atraižų kiekis (procentais, %). Pirmoje eilutėje yra siuvyklos, kuri siuva kostiumus, pavadinimas. Sukurkite klasę Kostiumas, kuri turėtų kintamuosius kostiumų duomenims saugoti. Kurio modelio kostiumui pasiūti reikia daugiausiai medžiagos ir kurio modelio kostiumui atraižų lieka mažiausiai?
- Papildykite programą veiksmais su dvieju siuvyklių siūlomų kostiumų rinkiniais. Kiekvienos siuvyklos duomenys saugomi atskiruose failuose. Kuriame sąraše yra kostiumas, kuriam reikia daugiausiai medžiagos? Surašykite į atskirą rinkinį visus abiejų sąrašų kostiumų modelius, kurie yra vidutinio dydžio, duomenis. Vidutinis kostiumas yra tas, kuriam pasiūti reikalingos medžiagos kiekis skiriasi nuo visų modelių reikalingos medžiagos aritmetinio vidurkio ne daugiau, kaip  $\frac{P}{M}$  proc.

**U3–11. Kepykla**

- Kepykloje kepamos duonas ir pyragų assortimentas yra faile: kepinio pavadinimas, miltų pavadinimas, kepinio rupumas, kepaliuko svoris ir kepimo prieaugis (procentais, %). Pirmoje eilutėje yra kepyklos pavadinimas. Sukurkite klasę Kepykla kepamos produkcijos duomenims saugoti. Kuris kepinys mažiausiai rupus ir kurio kepimo prieaugis didžiausias?
- Papildykite programą veiksmais su dviejų kepyklų kepinių rinkiniais. Kiekvienos kepyklos kepinių duomenys saugomi atskiruose failuose. Kuriame sąraše yra kepinys, kurio kepimo prieaugis didžiausias. Surašykite iš atskirų rinkinjų visus abiejų sąrašų kepinius, kurie yra vidutinio svorio, duomenis. Vidutinis kepaliukas tas, kurio svoris skiriiasi nuo visų kepinių kepaliukų aritmetinio vidurkio ne daugiau, kaip 10 proc.

**U3–12. Kepyklėlė**

- Kepyklėlėje kepamų pyragaičių duomenys yra faile: pavadinimas, dešimčiai vienetų pagaminti reikalingų kiaušinių kiekis, miltų kiekis ir kitų priedų kiekis. Pirmoje eilutėje yra kepyklėlės pavadinimas. Sukurkite klasę Pyragas pyragaičio duomenims saugoti. Raskite, kuriai pyragaičių rūšiai iškepti reikia mažiausiai kiaušinių ir kurios rūšies pyragaičiai sunkiausi (miltų ir kitų priedų kiekis).
- Papildykite programą veiksmais su dviejų kepyklėlių pyragaičių rinkiniais. Kiekvienos kepyklėlės pyragaičių duomenys saugomi atskiruose failuose. Kuriame sąraše yra pyragaitis, kuriam iškepti reikia daugiausia kiaušinių? Surašykite iš atskirų rinkinjų visus abiejų sąrašų pyragaičius, kurie yra vidutinio svorio (miltų ir kitų priedų kiekis), duomenis. Vidutinis pyragaičio svoris tas, kurio svoris skiriiasi nuo visų kepinių aritmetinio vidurkio ne daugiau, kaip 10 proc.

**U3–13. Nuotraukos**

- Nuotraukų kolekcijos duomenys yra faile: nuotraukos pavadinimas, tipas (spalvota ar nespalvota, stereo), aukštis ir plotis. Pirmoje eilutėje yra kolekcijos savininko pavardė ir vardas. Sukurkite klasę Nuotrauka nuotraukos duomenims saugoti. Raskite, kurio tipo yra didžiausia nuotrauka ir kiek kolekcijoje yra nurodyto tipo nuotraukų.
- Papildykite programą veiksmais su dviejų kolekcijų rinkiniais. Kiekvienos kolekcijos duomenys saugomi atskiruose failuose. Kurioje kolekcijoje yra didžiausia nuotrauka? Kurioje kolekcijoje yra daugiau nurodyto tipo nuotraukų? Surašykite iš atskirų rinkinjų visus abiejų sąrašų nespalvotų nuotraukų duomenis.

**U3–14. Transportas**

- Miesto viešojo transporto maršrutų duomenys yra faile: maršruto pavadinimas, ilgis kilometrais ir stotelų skaičius. Pirmoje eilutėje yra transporto įmonės pavadinimas. Sukurkite klasę Transportas vieno maršruto duomenims saugoti. Raskite visų maršrutų bendrą ilgi ir kiek stotelų turi ilgiausias maršrutas.
- Papildykite programą veiksmais su dviejų transporto įmonių (pvz., autobusų ir troleibusų) rinkiniais. Kiekvienos įmonės duomenys saugomi atskiruose failuose. Kurio įmonės maršrutų bendras ilgis didesnis ir kurios įmonės ilgiausias maršrutas turi daugiau stotelų? Surašykite iš atskirų rinkinjų abiejų sąrašų visus maršrutes, kuriuose yra ne mažiau, kaip n stotelės.

**U3–15. Miestai**

- Miestų duomenys yra faile: pavadinimas, plotas ir gyventojų skaičius. Pirmoje eilutėje yra valstybės pavadinimas. Sukurkite klasę Miestas vieno miesto duomenims saugoti. Raskite miestą, kurio gyventojų tankis didžiausias ir kiek valstybės miestuose gyvena žmonių.
- Papildykite programą veiksmais su dviejų valstybių miestų rinkiniais. Kiekvienos valstybės duomenys saugomi atskiruose failuose. Kurios valstybės miestų gyventojų skaičius didesnis ir kurioje valstybėje yra tankiausiai apgyvendintas miestas. Surašykite iš atskirų rinkinjų visus abiejų sąrašų miestus, kuriuose yra ne mažiau, kaip k gyventojų, duomenis.

**U3–16. Atmintukai**

- Atmintukų duomenys yra faile: markė, talpa ir kaina. Pirmoje eilutėje yra savininko pavardė ir vardas. Sukurkite klasę Atmintukas vieno atmintuko duomenims saugoti. Raskite kokia pigiausio atmintuko talpa ir už kokią pinigų sumą žmogus turi atmintuką.
- Papildykite programą veiksmais su dviejų žmonių atmintukų rinkiniais. Kiekvieno rinkinio duomenys saugomi atskiruose failuose. Pas kurį žmogų yra didesnės talpos pigiausias atmintukas ir kuris žmogus turi atmintuką už didesnę kainą. Surašykite iš atskirų rinkinjų abiejų sąrašų visus nurodytos markės atmintukų duomenis.

**U3–17. Fonoteka**

- Kūrinių duomenys yra faile: pavadinimas, autorius, grojimo trukmė ir failo dydis. Pirmoje eilutėje yra rinkinio savininko pavardė ir vardas. Sukurkite klasę Fonoteka vieno kūrinių duomenims saugoti. Kiek laiko reikia perkausyti visus turimus kūriniai? Kieno ir kokio kūrinių failas didžiausias?
- Papildykite programą veiksmais su dviejų savininkų rinkiniais. Kiekvieno rinkinio duomenys saugomi atskiruose failuose. Kuris savininkas ilgiau savo kūrinius gali klausyti ir pas kurį savininką yra didžiausios apimties kūrinių failas? Surašykite iš atskirų rinkinjų visus abiejų sąrašų kūrinius, kurių trukmė neviršija n minučių.

**U3–18. Dienraštis**

- Dienraščių duomenys yra faile: pavadinimas, tiražas ir kaina. Pirmoje eilutėje yra valstybės pavadinimas. Sukurkite klasę Dienraštis vieno dienraščio duomenims saugoti. Kurio dienraščio tiražo piniginė vertė didžiausia? Už kokią pinigų sumą yra pateikiama skaitytojui dienraščiu?
- Papildykite programą veiksmais su dviejų įmonių prenumeratorių rinkiniais. Kiekvieno rinkinio duomenys saugomi atskiruose failuose. Kurios įmonės prenumeratorių skaitomi dienraščiai už didesnę pinigų sumą ir kurioje įmonėje yra dienraštis, kurio piniginė vertė didesnė? Surašykite iš atskirų rinkinjų visus abiejų sąrašų dienraščius, kurių kaina neviršija n pinigų.

**U3–19. Žurnalas**

- Žurnalų duomenys yra faile: pavadinimas, puslapių skaičius ir kaina. Pirmoje eilutėje yra valstybės pavadinimas. Sukurkite klasę Žurnalas vieno žurnalo duomenims saugoti. Kurio žurnalo puslapio kaina didžiausia? Kokia bendra visų žurnalų puslapiai suma ir kokia vieno puslapio vidutinė kaina?
- Papildykite programą veiksmais su dviejų valstybių žurnalų rinkiniais. Kiekvieno rinkinio duomenys saugomi atskiruose failuose. Kurioje valstybėje žurnalo puslapio kaina didžiausia? Kurios valstybės leidžiamų žurnalų vieno puslapio vidutinė kaina didesnė? Surašykite iš atskirų rinkinjų visus abiejų valstybių žurnalus, kurių kaina neviršija n pinigų.

**U3–20. Vaiduoklis**

- Senose pilyse gyvena vaiduokliai. Duomenys apie juos yra faile: pavadinimas, amžius ir pasirodymo dažnumas. Pirmoje eilutėje yra valstybės pavadinimas. Sukurkite klasę Vaiduoklis vieno vaiduoklio duomenims saugoti. Koks seniausio vaiduoklio pasirodymo dažumas? Koks valstybės pilyse gyvenančių vaiduoklių amžiaus vidurkis?
- Papildykite programą veiksmais su dviejų valstybių vaiduoklių rinkiniais. Kiekvieno rinkinio duomenys saugomi atskiruose failuose. Kurios valstybės seniausio vaiduoklio pasirodymo dažumas didesnis? Kurios valstybės vaiduoklių amžiaus vidurkis mažesnis? Surašykite iš atskirų rinkinjų visus abiejų valstybių vaiduoklius, kurių amžius neviršija n metų.

**U3–21. Batai**

- Duomenys apie parduodamus batus yra faile: gamintojo pavadinimas, dydis, svoris ir kaina. Pirmoje eilutėje yra savininko (parduotuvės) pavadinimas. Sukurkite klasę Batas vieno bato duomenims saugoti. Kiek kainuoja sunkiausia batų pora? Kiek yra nurodytos firmos nurodyto dydžių batų?
- Papildykite programą veiksmais su dviejų parduotuvų batų rinkiniais. Kiekvieno rinkinio duomenys saugomi atskiruose failuose. Kurios parduotuvės sunkiausia batų pora pigesnė? Kurioje parduotuveje yra daugiau nurodytos firmos nurodyto dydžio batų? Surašykite iš atskirų rinkinjų visus abiejų parduotuvų batus, kurių kaina neviršija n pinigų.

**U3–22. Kaliausė**

- Duomenys apie ūkininkų pagamintas kaliauses yra faile: tipas, aprėdu skaičius, svoris ir kaina. Pirmoje eilutėje yra ūkininko pavardė ir vardas. Sukurkite klasę Kaliausė vienos kaliausės duomenims saugoti. Kiek aprėdu turi sunkiausia kaliausė? Kokia vidutinė kaliausės kaina?
- Papildykite programą veiksmais su dviejų ūkininkų kaliausės rinkiniais. Kiekvieno rinkinio duomenys saugomi atskiruose failuose. Kurio ūkininko sunkiausia kaliausė daugiau aprėyta? Kurio ūkininko vidutinė kaliausės kaina mažesnė? Surašykite iš atskirų rinkinjų visus abiejų ūkininkų nurodyto tipo kaliausės duomenis.

**U3–23. Saldainiai**

- Duomenys apie saldainius yra faile: pavadinimas, tipas ir kilogramo kaina. Pirmoje eilutėje yra savininko pavardė ir vardas. Sukurkite klasę Saldainis vieno saldainio duomenims saugoti. Kiek kainuoja saldainiai, jeigu kiekvieno pavadinimo yra po n kilogramų. Kokie nurodyto tipo saldainiai brangiausi?

- Papildykite programą veiksmais su dviejų studentų saldainių rinkiniais. Kiekvieno rinkinio duomenys saugomi atskiruose failuose. Kurio studento rinkinys kainuoja brangiau, jeigu pirmasis studentas turi visų saldainių pavadinimų po n1 kilogramų, o antrasis – po n2? Surašykite iš atskirų rinkinių visus abiejų studentų saldainių, kurių kaina didesne už k pinigų, duomenis.

**U3–24. Reklama**

- Duomenys apie reklamas yra faile: užsakovas, minutės kaina, parodymų skaičius paroje. Pirmoje eilutėje yra TV pavadinimas. Sukurkite klasę **Reklama** vienos reklamos duomenims saugoti. Kiek pinigų sumoka reklamą užsakovai per parą? Kiek kainuoja ilgiausios trukmės reklama?
- Papildykite programą veiksmais su dviejų televizijų reklamų rinkiniais. Kiekvieno rinkinio duomenys saugomi atskiruose failuose. Kuri televizija per parą uždirba daugiau? Kurioje televizijoje ilgiausios trukmės reklama brangesnė? Surašykite iš atskirų rinkinių visus abiejų televizijų reklamų, kurių minutės kaina didesne už k pinigų, duomenis.

**4. Konteinerinė klasė**

Susipažinsite su:

- konteinerine klase, jos sasajos metodais;
- palyginimo ir loginių operacijų užklojimu konteineryje;
- rikiavimo, pašalinimo algoritmai.

**4.1. Konteinerinės klasės sasajos metodai**

- Dvi klasės: viena iš jų – konteinerinė klasė,
- sasajos metodai `Dėti()` ir `Imti()`.

**Užduotis.** Sodas.

Sode yra n obelų. Kiekviena obelis apibūdinama tokiomis charakteristikomis: pirmaisiais metais užderėjusių obuolių kiekiu kiek, obuolių priaugiu priaug kiekvienais metais ir dėsniu, pagal kurį didėja obuolių kiekis. Dėsniu nurodomi 2 koeficientai `koef1` ir `koef2`. Dėsnis yra bendras visoms obelims, o koeficientai gali būti skirti. Reikia:

- rasti kiekvienais metais kiekvienos obels užderančių obuolių kiekį, kai žinomas metų kiekis. Metų kiekis nurodomas konstanta arba įvedamas klaviatūra;
- rasti nurodytais metais kiekvienos obels užderančių obuolių kiekį;
- suformuoti naują sodą iš obelų, kurios per nurodytą metų kiekį sunokino ne mažesnį, nei nurodyta bendrą obuolių kiekį. Duoti dydžiai nurodomi konstantomis arba įvedami klaviatūra.

Dėsnis:

$$y = z * t^2 - koef2 * t + koef1, \quad t = \sqrt{\sin(koef1 * z) - 0.1}$$

z kinta nuo kiek žingsniu priaug, `koef1`, `koef2` – prieaugio koeficientai.

**Pradiniai duomenys ir rezultatai.**

| Pradiniai duomenys                                   | Pirmais žingsniais rezultatai                                                                         |
|------------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| 4<br>2 8 8 10<br>1 12 10 14<br>3 8 7 10<br>5 12 6 14 | Informacija apie obelis<br>Koef1 koef2 kiek priaug<br>2 8 8 10<br>1 12 10 14<br>3 8 7 10<br>5 12 6 14 |

**Programos kūrimo eiga.**

- Sukuriama klasę **Obelis** vienos obels duomenims saugoti. Parašomas konstruktorius ir spaustinimo metodas.
- Sukuriama konteinerinė klasę **Sodas** sodo duomenims saugoti. Parašomas konstruktorius ir sasajos metodai. Pagrindinėje programe parašomas įvedimas ir spaustinimas.
- Realizuojamas užduoties a) punktas.
- Realizuojamas užduoties b) punktas.
- Realizuojamas užduoties c) punktas.

**1 Pirmas žingsnis.**

- Sukurkite klasę **Obelis** duomenims saugoti:

```
//-----
class Obelis
{
private:
    int kiek, priaug; // atitinka užduotyje z0 ir zh
    int koef1, koef2; // atitinka užduotyje a ir b
public:
    Obelis(): kiek(0), priaug(16), koef1(1), koef2(2) { }
```

```

        Obelis(int kiek, int priaug, int koef1, int koef2):
            kiek(kiek), priaug(priaug), koef1(koef1), koef2(koef2) { }
            string Spausdinti();
};

//-----
    • Parašykite klasės metodą Spausdinti():

//-----
#include "obelis.h"
#include <sstream>
#include <iomanip>
using namespace std;
//-----
// Atspausdina duomenis į eilutę
string Obelis::Spausdinti()
{
    stringstream srautas;
    srautas << setw(6) << koef1 << setw(6) << koef2
        << setw(5) << kiek << setw(7) << priaug ;
    return srautas.str();
}
//-----
    • Sukurkite klasę Sodas sodo duomenims saugoti:

//-----
#include "obelis.h"
class Sodas
{
public:
    static const int CMaxi = 100;
private:
    Obelis Obelys[CMaxi];
    int n;
public:
    Sodas():n(0) { }
    Obelis Imti(int i) { return Obelys[i]; }
    int Imti() { return n; } // užklotas metodas
    void Dėti(Obelis ob) { Obelys[n++] = ob; }
};
//-----
    • Parašykite main() funkciją:

//-----
#include "sodas.h"
#include <fstream>
#include <iostream>
#include <iomanip>
using namespace std;
//-----
const char Cul[]="U1.txt";
//-----
void Ivesti(Sodas & sodas, const char fvard[]);
void Išvesti(Sodas & sodas);
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Sodas sodas;
    Ivesti(sodas, Cul);
    Išvesti(sodas);
    return 0;
}
//-----
// Skaitomi pradiniai duomenys į objekta sodas iš failo fvard
void Ivesti(Sodas & sodas, const char fvard[])
{

```

```

    ifstream fd(fvard);
    int n;
    fd >> n;
    int koef1, koef2, kiek, priaug;
    for (int i = 0; i < n; i++) {
        fd >> koef1 >> koef2 >> kiek >> priaug;
        sodas.Dėti(Obelis(kiek, priaug, koef1, koef2));
    }
    fd.close();
}

//-----
// Išvedami objekto sodas duomenys
void Išvesti(Sodas & sodas)
{
    cout << " Informacija apie obelis\n";
    cout << " Koef1 koef2 kiek priaug\n";
    for (int i = 0; i < sodas.Imti(); i++)
        cout << sodas.Imti(i).Spausdinti() << endl;
}

    • Sukurkite duomenų failą vardu U1.txt. Kompiliuokite programą. Patikrinkite atsakymą:
Informacija apie obelis
Koef1 koef2 kiek priaug
    2      8      8      10
    1      12     10      14
    3      8       7      10
    5      12     6      14

¶ Antras žingsnis.
    • Papildykite klasę Obelis metodais Dėsnis() ir Obuoliai(). Jų prototipus įrašykite į klasės aprašo failą. Metodo Dėsnis() prototipą talpinkite į klasės aprašo private dalį. Atkreipkite dėmesį į metodo Dėsnis() realizaciją – obuoliai dalimis neauga, dėl to buvo panaudota apvalinimo funkcija į mažesnę pusę floor() ir tipų konversija. Nepamirškite dviejų direktyvų, reikalingų atitinkamai pirmajam ir antrajam metodams:
//-----
#include <cmath>
#include <iostream>
//-----
// Formulės metų derliui paskaičiuoti realizacija
int Obelis::Dėsnis(double a, int b, double z)
{
    if (sin(a * z) > 0.1) {
        double t = pow(sin(a * z) - 0.1, 0.5);
        int y = (int) floor(a - b * t + z * t * t);
        return y;
    }
    else
        return 0;
}
//-----
// Skaičiuoja kiekvienais metais iki nurodytų metų obuolių kieki
void Obelis::Obuoliai(int metai)
{
    int z = kiek;
    int y;
    for (int i = 0; i < metai; i++) {
        if ((y = Dėsnis(koef1, koef2, z)) > 0)
            cout << i << setw(10) << y << endl;
        else
            cout << i << setw(16) << "nėra obuolių" << endl;
        z = z + priaug;
    }
}

```

- Pagrindinę programą papildykite funkcija `Skaiciuoti()`. Jos prototipą paskelbkite virš pagrindinės funkcijos. Papildykite pagrindinę programą kreipiniu į šią funkciją. Pasirinkite, kokiui būdu nurodysite metus: konstanta ar klaviatūra:
- ```
//-----
// Spausdina sodo obelų derlių iki nurodytų metų
void Skaiciuoti(Sodas & sodas, int metai)
{
    cout << " Informacija apie derliu\n";
    for (int i = 0; i < sodas.Imti(); i++){
        cout << i + 1 << " obelis\n";
        sodas.Imti(i).Obuoliai(metai);
    }
}

//-----
```

- Jei metų kiekis 4, tai atsakymas turi būti tokis:

Informacija apie derliu  
1 obelis  
0      nėra obuoliu  
1      nėra obuoliu  
2      nėra obuoliu  
3            14  
2 obelis  
0      nėra obuoliu  
1      nėra obuoliu  
2            3  
3            35  
3 obelis  
0            1  
1            6  
2      nėra obuoliu  
3      nėra obuoliu  
4 obelis  
0      nėra obuoliu  
1      nėra obuoliu  
2            7  
3            34

### Trečias žingsnis.

- Suraskite nurodytų metų kiekvienos obels išaugintų obuolių kiekį. Jei metai 4, tai atsakymas turi būti tokis:

Informacija apie derliu  
1 obelis 14  
2 obelis 35  
3 obelis 0  
4 obelis 34

### Ketvirtas žingsnis.

- Papildykite klasę `Obelis` metodu `VisoObuoliu()`, kuris suskaičiuotų obels derlių per prabėgusius metus.
- Pagrindinę programą papildykite funkcija `Naujas()`, kuri atrinks į naujų objektų sąrašą tas obelis, kurių derlius yra didesnis už nurodytą kiek (pirmaisiais metais užderėjusių obuolių kiekij) reikšmę. Pasirinkite, kokiui būdu nurodysite obels derliaus kiekij: konstanta ar įvedimu klaviatūra. Prototipą paskelbkite virš pagrindinės funkcijos.
- Nepamirškite paskelbtį naujo `Sodas` objekto.
- Idėkite kreipinį į naujų funkciją, o taip pat ir į rezultatų spausdinimo funkciją:

```
//-----
// Formuoja naują sodą iš esamo pagal derlingumo reikšmę, didesnę už kiek,
// nurodytais metais
void Naujas(Sodas & sodas, int metai, Sodas & sodasn, int kiek)
```

```
{  

    for (int i = 0; i < sodas.Imti(); i++){  

        if (sodas.Imti(i).VisoObuoliu(metai) > kiek)  

            sodasn.Deti(sodas.Imti(i));  

    }  

}  

//-----
```

- Išbandykite programą. Jei nurodytas derliaus dydis yra 37, tai atsakymas turi būti tokis:

Informacija apie obelis
Koef1   koef2   kiek   priaug
1            12      10      14
5            12      6       14

### Savarankiško darbo užduotis.

Pateikiamas 9 aukštų namo parduodamų butų sąrašas. Kiekviename laiptinės aukšte yra po 3 butus. Žinomas buto numeris, bendras plotas, kambarių skaičius, pardavimo kaina, telefono Nr. Suraskite butus, kurie turi nurodytą kambarių skaičių ir kurių kaina neviršija nurodytos kainos, ir juos surašykite į tinkamų butų masyvą.

### 4.2. Operatorių užklojimas konteineryje

- Operatorių `<=`, `!` užklojimų realizavimas.
- Rikiavimas.
- Šalinimas.

### Užduotis. Fakultetas.

Tekstiniame faile saugoma informacija apie studentus: pavardė, vardas, grupė, pažymiu kiekis, pažymiai. Sudaryti konteinerinę klasę, saugančią informaciją apie studentus. Reikia:

- sudaryti naują sąrašą, kuriame būtų tik pirmūnai (gavo tik 10 arba 9);
- surikiuoti pradinį sąrašą pagal pavardę ir vardą abėcėliškai;
- pašalinti iš pradinio sąrašo studentus, kurie nėra pirmūnai.

### Pradiniai duomenys ir rezultatai.

Pradiniai duomenys				
Petraitis, Jonas, IF-1/8, 4	10	8	9	9
Algaitis, Algis, IF-1/8, 4	10	9	9	9
Petraitis, Kazys, IF-1/8, 4	9	9	10	10
Algaitis, Rimas, IF-1/8, 4	9	9	9	9
Petraitis, Vytaas, IF-1/9, 4	10	8	9	7
Petraitis, Anupras, IF-1/9, 4	9	8	7	6
Petraitis, Vidas, IF-1/9, 4	8	9	8	9
Petraitis, Anzelmas, IF-1/9, 4	8	9	9	9
Petraitis, Aurimas, IF-1/9, 4	10	9	10	9

Rezultatai				
Pavardė	Vardas	Grupė	Pažymiai	
Algaitis	Algis	IF-1/8	10	9
Algaitis	Rimas	IF-1/8	9	9
Petraitis	Anupras	IF-1/9	9	8
Petraitis	Anzelmas	IF-1/9	8	9
Petraitis	Aurimas	IF-1/9	10	9
Petraitis	Jonas	IF-1/8	10	8
Petraitis	Kazys	IF-1/8	9	9
Petraitis	Vidas	IF-1/9	8	9
Petraitis	Vytaas	IF-1/9	10	8

Naujas sąrašas				
Pavardė	Vardas	Grupė	Pažymiai	
Algaitis	Algis	IF-1/8	10	9 9 9
Algaitis	Rimas	IF-1/8	9	9 9 9
Petraitis	Aurimas	IF-1/9	10	9 10 9
Petraitis	Kazys	IF-1/8	9	9 10 10

Koreguotas pradinis sąrašas				
Pavardė	Vardas	Grupė	Pažymiai	
Algaitis	Algis	IF-1/8	10	9 9 9
Algaitis	Rimas	IF-1/8	9	9 9 9
Petraitis	Aurimas	IF-1/9	10	9 10 9
Petraitis	Kazys	IF-1/8	9	9 10 10

### Programos kūrimo eiga.

- Sudaroma klasė Studentas studento duomenims saugoti. Parašomas konstruktorius be parametru, metodas Dėti(), ir metodas spausdinimui į eilutę.
- Sudaroma konteinerinė klasė Fakultetas fakulteto duomenims saugoti. Parašomas konstruktorius be parametru, sąsajos metodas skaitliuko paémimui, sąsajos metodai elemento paémimui ir įrašymui.
- Pagrindinėje programoje parašomas įvedimas ir spausdinimas.
- Realizuojamas užduoties a) punktas: papildoma klasė Studentas ir pagrindinė programa.
- Realizuojamas užduoties b) punktas: papildoma klasė Studentas ir Fakultetas bei pagrindinė programa.
- Realizuojamas užduoties c) punktas.

### ¶ Pirmas žingsnis.

- Sukurkite klasę studento duomenims saugoti:

```
-----  
#include <iomanip>  
using namespace std;  
-----  
class Studentas  
{  
public:  
    static const int Cpaž = 10;  
private:  
    string pavardė, vardas, grupė;  
    int np, Paž[Cpaž];  
public:  
    Studentas():np(0), pavardė(""), vardas(""), grupė("") {}  
    void Dėti(string pav, string vard, string grup, int np, int pž[]);  
    string Spausdinti();  
};  
-----
```

- Parašykite klasės metodus:

```
#include "Studentas.h"  
#include <sstream>  
#include <iomanip>  
using namespace std;  
-----  
// Sąsajos metodas  
void Studentas::Dėti(string pav, string vard, string grup, int np, int pž[]){  
    pavardė = pav;  
    vardas = vard;  
    grupė = grup;
```

```
Studentas::np = np;  
for (int i = 0; i < np; i++)  
    Paž[i] = pž[i];  
}  
-----  
// Reikšmių spausdinimas į eilutę  
string Studentas::Spausdinti()  
{  
    stringstream eil;  
    eil << left << setw(12) << pavardė << setw(10) << vardas  
        << setw(8) << grupė << right;  
    for (int i = 0; i < np; i++)  
        eil << setw(3) << Paž[i];  
    eil << endl;  
    return eil.str();  
}  
-----
```

- Sukurkite klasę fakulteto duomenims saugoti:

```
#include "Studentas.h"  
-----  
class Fakultetas  
{  
public:  
    static const int CMax = 100;  
private:  
    Studentas St[CMax];  
    int n;  
public:  
    Fakultetas(): n(0) {}  
    int Imti() { return n; }  
    Studentas Imti(int i) { return St[i]; }  
    void Dėti(const Studentas & ob)  
    { St[n++] = ob; }  
};  
-----
```

- Parašykite funkciją main(), įvedimo bei išvedimo funkcijas: duomenys įvedami iš failo, išvedami į failą. Pavardės ir vardai gali būti sudaryti iš kelių žodžių, todėl įvedimo funkcija bus pakankamai sudėtinga. Eilučių skaičius duomenų faile nenurodytas. Taip pat reikia tikrinti, kad neviršytume masyvo dydžio. Ruošdami duomenų failą, po paskutinės eilutės „Enter“ klavišo nespauskite, nes įvesite papildomą nereikalingą eilutę.

```
-----  
#include <fstream>  
#include <iomanip>  
using namespace std;  
#include "Fakultetas.h"  
-----  
const char CDfv[] = "U1.txt";  
const char CRfv[] = "Rezultatai.txt";  
-----  
void Duomenys(Fakultetas & R);  
void Spausdinti(Fakultetas & R, const string & antraštė);  
-----  
int main()  
{  
    ifstream fr(CRfv);  
    fr.close();  
    Fakultetas info; // duomenys  
    Duomenys(info); // skaitomi duomenys  
    Spausdinti(info, "Pradiniai duomenys");  
    return 0;  
}  
-----  
// Skaitomi pradiniai duomenys į objekta R  
void Duomenys(Fakultetas & R)  
{
```

```

ifstream fd(CDfv);
bool yra = true;
string pv, vd, gr;
int np, Pž[Studentas::Cpaž];
Studentas s1;
int ns = 0; // studentų skaitiklis
while (!fd.eof() && yra) { // kol yra duomenų ir jie telpa į masyvą
    getline(fd, pv, ','); // įvedami keli žodžiai
    fd >> ws; // praleidžiamai tarpai
    getline(fd, vd, ',');
    fd >> ws;
    getline(fd, gr, ',');
    fd >> np;
    for (int i = 0; i < np; i++)
        fd >> Pž[i];
    fd.ignore();
    s1.Dėti (pv, vd, gr, np, Pž);
    if (ns - 1 < Fakultetas::CMax )
        R.Dėti(s1);
    else
        yra = false;
}
fd.close();
//-----
// Spausdinami objekto R duomenys
void Spausdinti(Fakultetas & R, const string & antraštė)
{
    ofstream fr(CRfv, ios::app);
    fr << setw(30) << antraštė << endl;
    fr << " Pavarde Vardas Grupė Pažymiai \n";
    fr << "-----\n";
    for (int i = 0; i < R.Imti(); i++)
        fr << R.Imti(i).Spausdinti();
    fr << "-----\n\n\n";
    fr.close();
}
//-----

```

- Sukurkite duomenų failą U1.txt. Kompiliuokite programą. Patikrinkite atsakymą.

#### ¶ Antras žingsnis.

- Sukurkite naują Fakultetas klasės objektą.
- Papildykite klasę Studentas operatoriaus != užklojimo metodu:

```

//-----
bool Studentas::operator != ()
{
    for (int i = 0; i < np; i++)
        if(Paž[i] < 9)
            return true;
    return false;
}
//-----
• Pagrindinę programą papildykite funkcija Kurti(). Jos prototipą paskelbkite virš pagrindinės
funkcijos. Papildykite pagrindinę programą kreipiniu į šią funkciją:
//-----
// Kopijuojami pirmūnai iš objekto D į objektą R
void Kurti(Fakultetas & D, Fakultetas & R)
{
    int ns = 0;
    for (int i = 0; i < D.Imti(); i++)
        if (!D.Imti(i))
            ;
        else
            R.Dėti(D.Imti(i));
}

```

```

}
//-----

```

#### ¶ Trečias žingsnis.

- Realizuokite rikiavimo metodą klasėje Fakultetas. Tuo tikslu papildykite klasę Studentas operatoriaus <= užklojimo metodu:

```

//-----
bool operator <= (const Studentas & kitas)
{
    return (pavarde < kitas.pavarde ||
            pavarde == kitas.pavarde && vardas < kitas.vardas);
}
//-----

```

- Papildykite klasę Fakultetas rikiavimo metodu:

```

//-----
void Fakultetas::Rikiuoti()
{
    for (int i = 0; i < n - 1; i++) {
        Studentas min = St[i];
        int im = i;
        for (int j = 0; j < n; j++)
            if (St[i] <= min) {
                min = St[i];
                im = j;
            }
        St[im] = St[i];
        St[i] = min;
    }
}
//-----

```

- Išbandykite programą. Deja, rezultatai nesurikiuoti. Ieškokite klaidos. Taisykite programą. Vėl bandykite.
- Savarankiškai pakeiskite operatoriaus <= užklojimo metodą, kad rikiuotų pagal grupę, pavardę ir vardą abécéliškai.

#### ¶ Ketvirtas žingsnis.

Savarankiškai realizuokite studentų, kurie nėra pirmūnai, pašalinimo iš pradinio sąrašo metodą, nekeisdami rikiavimo tvarkos.

#### Savarankiško darbo užduotis.

Pirmaje failo eilutėje nurodytas fakulteto pavadinimas. Tekstiniame faile yra fakulteto žiemos sesijos pažymų sąrašas. Eilutėje apie studentą yra tokie duomenys: pavardė, vardas, grupė, pažymų kiekis, pažymiai. Nustatykite kiekvienos grupės bendrą mokymosi vidurkį. Rezultatus išspausdinkite surikiuotus mažėjančiai pagal vidurkį ir pagal grupes abécéliškai.

### 4.3. Kontroliniai klausimai

- Kokią reikšmę grąžina užklotas operatorius, pateiktas klasės apraše?

```

class Medis {
private:
    string pav;
    int aukstis;
public:
    ...
    bool operator * (const Medis & kitas) {
        return aukstis > kitas.aulstis ||
               aukstis == kitas.aulstis &&
               pav > kitas.pav;
    }
};

```

- operatorius grąžina true, jeigu abu medžiai yra vienodo aukščio, tačiau pirmo medžio pavadinimas yra uosis, o kitas – obelis.

- b. operatorius grąžina true, jeigu kitas medis aukštesnis, o pavadinimai vienodi.  
 c. operatorius grąžina true, jeigu abu medžiai yra uosiai ir jie vienodo aukščio.  
 d. operatorius klaidingas.
2. Kuris iš tolimesnių teiginių yra teisingas apie pateiktą programos fragmentą?
- ```
class Taškas {
private:
    double x, y;
    double atstumas;
    void Atstumas() {
        atstumas = sqrt(x * x + y * y);
    }
public:
    . .
    void Dėti(double x1, double y1) {
        x = x1; y = y1; Atstumas();
    }
};

a. programos fragmente klaidų nėra;
b. metodas Atstumas() privalo būti public dalyje;
c. metodas negali kreiptis į kitą tos pačios klasės metodą;
d. visi klasės metodai privalo būti tik public dalyje.
```
3. Ar yra kлаida tolimesniame programos fragmente? Jei taip, tai kokia kлаida?
- ```
int A[8];
A[0] = 1;
A[1] = 2;
for (int i = 2; i < 9; i++)
    A[i] = A[i-1] + A[i-2];
```
4. Funkcijos parametrai yra 2 vienodo dydžio sveikujų skaičių masyvai ir sveikojo tipo kintamasis, nurodantis masyvų ilgį. Parašykite funkciją, kuri kopijuoja pirmojo masyvo reikšmes atvirkštine tvarka į antrajį masyvą.
5. Kokią klasę vadiname konteinerine klase?
6. Išvardinkite konteinerinės klasės sąsajos metodus ir nurodykite jų funkcijas
7. Kokio tipo parametras turi būti klasėje užklojamame dviviečiam operatoriuje?
8. Kada iškviečiamas klasėje užklotas operatorius?
9. Kokią funkciją atlieka toks operatorius `fd >> ws;`, kai `fd` – ifstream tipo failo kintamasis?
10. Papasakokite kokiomis ypatybėmis pasižymi išrinkimo rikiavimo metodas.

#### 4.4. Užduotys

##### U4-1. Bukletai

Spaustuvėje spausdinamų bukletų duomenys saugomi faile: bukleto formatas, 500 lapų tokio formato popieriaus pakuotės kaina, bukleto lapų skaičius ir egzempliorių kiekis. Parašykite programą, kuri suskaičiuotų kiekvieno bukleto gamybai reikalingo popieriaus kainą ir surastą pigiausią užsakymą. Papildykite programą veiksmais, kurie leistų atrinkti bukletus, kurių gamybai užtenka vienos 500 lapų pakuotės, ir ši sąrašą surikiuoti pagal bukleto lapų skaičių ir egzempliorių kiekį mažėjimo tvarka.

##### U4-2. Mobiliojo ryšio kortelės

Norėdamas palyginti mobiliojo ryšio operatorių siūlomas išankstinio mokėjimo korteles Sirvydas surinko šią informaciją į tekstinį failą. Faile eilutėmis yra kortelių duomenys: kortelės (tinklo) pavadinimas, pradinė suma kortelėje, tarifas savame tinkle, tarifas į kitus tinklus, SMS žinučių tarifas savame tinkle ir į kitus tinklus. Parašykite programą, kuri spausdintų kortelių duomenis lentele, surastą kortelę, kurios SMS žinučių tarifai į kitus tinklus mažiausi. Papildykite programą veiksmais, kurie leistų atrinkti korteles, kurios leidžia skambinti ir siusti SMS žinutes savame tinkle nemokamai, ir ši sąrašą surikiuoti pagal pradinę sumą mažėjimo tvarka ir kortelės pavadinimą abécéliskai.

##### U4-3. Dėžės

Ūkininkas, ieškodamas taros savo auginamai produkcijai sandeliuoti, analizuoja dėžes. Tekstiniame faile yra surašyti ūkininko analizuojamos taros duomenys. Eilutėmis surašyti dėžių charakteristikos: medžiaga, iš kurios pagaminta dėžė, ilgis, plotis, aukštis, maksimalus produkcijos svoris, kurį gali atlaikyti dėžė, bei kiek dėžių galima sukruti viena ant kitos. Parašykite programą, kuri spausdintų

dėžių duomenis lentele, surastą dėžę, į kurią telpa daugiausiai produkcijos ir suskaičiuotų kokio aukščio lentynas turi pastatyti sandėlyje (trūkstamus sandėlio duomenis programa paprašo įvesti klaviatūra), jeigu ūkininkas naudos šias dėžes. Papildykite programą veiksmais, kurie leistų atrinkti nurodytos medžiagos dėžes ir ši sąrašą surikiuoti pagal maksimalų produkcijos svorį ir leistiną dėžių sukravimą viena ant kitos mažėjimo tvarka.

##### U4-4. Indėliai

Banke yra laikomi terminuoti indėliai. Jų duomenys surašyti tekstiniame faile eilutėmis: indėlio dydis, sutarties terminas, metinė palūkanų norma, kaip skaičiuojamos palūkanos. Palūkanos gali būti skaičiuojamos: a) indėlio palūkanos apskaičiuojamos sutartinio laikotarpio pabaigoje; b) indėlio palūkanos apskaičiuojamos kas mėnesį ir pridedamos prie indėlio; c) indėlio palūkanos apskaičiuojamos kartą per metus ir pridedamos prie indėlio. Parašykite programą, kuri spausdintų indėlių duomenis lentele, surastą didžiausią indėlį, suskaičiuotų bendrą sumą, kurią bankas turės išmokėti visiems indėlių savininkams. Papildykite programą veiksmais, kurie leistų atrinkti indėlius, kurių palūkanos apskaičiuojamos nurodytu būdu, ir ši sąrašą surikiuoti pagal sutarties terminą ir indėlio dydį mažėjimo tvarka.

##### U4-5. Matavimo prietaisai

Tekstiniame faile surašyta įvairių matavimo prietaisų informacija: pavadinimas, matavimo ribos [nuo, iki], matavimo vienetai, skalės didžiųjų padalų skaičius, mažųjų padalų skaičius didžiojoje padaloje. Parašykite programą, kuri spausdintų matavimo prietaisų informaciją lentele, surastą didžiausią mažiausios padalos vertę turinčią prietaisą (absoliutine reikšme, matavimo vienetai nesvarbūs), suskaičiuotų vienos rūšies prietaisų kiekį (prietaisų rūšių nusako pavadinimas, ją reikia įvesti klaviatūra). Papildykite programą veiksmais, kurie leistų atrinkti prietaisus, turinčius nurodytą mažiausios padalos vertę (absoliutine reikšme) ir ši sąrašą surikiuoti pagal matavimo ribas didėjimo tvarka.

##### U4-6. Lazeriniai spausdintuvai

Tekstiniame faile turime įvairių gamintojų lazerinių spausdintuvų informaciją. Faile eilutėmis surašytos spausdintuvų charakteristikos: gamintojas, modelis, vienpusio spausdinimo sparta, dvipusio spausdinimo sparta (jeigu spausdintuvas neturi dvipusio spausdinimo galimybės, sparta - 0), pirmojo puslapio išspausdinimo laikas. Parašykite programą, kuri spausdintų spausdintuvų duomenis lentele, suskaičiuotų, kiek modelių gali spausdinti dvipusius spaudinius, surastą sparčiausią vienpusi spausdintuvą. Papildykite programą veiksmais, kurie leistų atrinkti tik dvipusio spausdinimo galimybę turinčius spausdintuvus ir ši sąrašą surikiuoti pagal gamintoją ir modelį abécéliskai.

##### U4-7. Šulinį užterštumas

Seniūnas suregistravo kaimo sodybose esančius geriamo vandens šulinius ir atliko nitratų kiekio šių šulinii vandenye tyrimus. Faile eilutėmis surašyti sodybų šulinii duomenys: sodybos adresas, šulinio gylis, šulinio skersmuo, nitratų kiekis. Parašykite programą, kuri spausdintų šulinii informaciją lentele, surastą giliausią šulinį, suskaičiuotų, keliuose šuliniuose nitratų kiekis viršija leistiną normą (normą programa paprašo įvesti klaviatūra). Papildykite programą veiksmais, kurie leistų atrinkti nurodytus gatvės sodybų šulinius ir ši sąrašą surikiuoti pagal nitratų kiekį didėjimo tvarka ir šulinio gylį mažėjimo tvarka.

##### U4-8. Moduliai

Tekstiniame faile duota informacija apie dėstytojų vedamus modulius: modilio pavadinimas, kreditų kiekis, atestacijos laikotarpis [nuo, iki], dėstytojo pavardė, vardas. Parašykite programą, kuri atrinktų nurodyto dėstytojo vedamus modulius į atskirą sąrašą, suskaičiuotų šio sąrašo kreditų sumą, surikiuotų pagal kreditų kiekį mažėjimo tvarka ir modilio pavadinimą abécélės tvarka. Papildykite programą veiksmais, kurie leistų iš pradinio modulių sąrašo pašalinti modulius, kurių atestacija baigiasi šiais metais.

##### U4-9. Kopijuokliai

Tekstiniame faile surašytos skaitmeninių kopijavimo aparatu techninės charakteristikos: gamintojas, modelis, maksimalus popieriaus formatas, kopijavimo sparta, pirmojo puslapio išspausdinimo laikas, standartinės popieriaus kasetės talpa. Parašykite programą, kuri spausdintų kopijavimo aparatu duomenis lentele, surastą didžiausią popieriaus talpą turinčią kopijavimo aparatu, suskaičiuotų pastarojo aparato gamintojo gaminamų modelių kiekį. Papildykite programą veiksmais, kurie leistų atrinkti nurodyta sparta (spartos intervalas įvedamas klaviatūra) kopijuojančius aparatus ir surikiuotų pagal spartą mažėjimo tvarka ir gamintojų abécélės tvarka.

**U4-10. Kopijavimo darbai**

Tekstiniame faile surašyti kopijavimo darbai: popieriaus formatas, lapų skaičius, kopijų skaičius, vienpusis ar dvipusis kopijavimas, kopijų grupavimas, papildoma kopijos apdorojimo funkcija. Papildoma kopijų apdorojimo funkcija, jeigu jos reikia, gali būti susegimas arba skylių išmušimas. Parašykite programą, kuri spausdintų darbų sąrašą lentelę, suskaičiuotų, kiek darbų turi papildomą susegimo funkciją ir kiek darbų turi skylių išmušimo funkciją, surastą daugiausiai kopijavimo aparato ciklų reikalaujantį darbą. Papildykite programą veiksmais, kurie leistų atrinkti vienpusio kopijavimo darbus ir surikiuotų juos pagal formatą ir bendrą reikalingo popieriaus kiekį mažėjimo tvarka.

**U4-11. Reklaminiai klipai**

Tekstiniame faile surašyti reklaminių video klipų duomenys: reklamuojamas produktas, produkto grupė, video klipo trukmė, klipo kraštinių santykis pikseliais (formatas). Parašykite programą, kuri spausdintų klipų duomenis lentelę, suskaičiuotų bendrą klipų trukmę, sudarytų reklaminių klipų sąrašą rodymui, kuriame būtų rodomi nurodyto vienodo formato klipai ir būtų rodomi po vieną klipą iš kiekvienos produkto grupės, jį surikiuotų pagal klipo trukmę ir produkto grupę ir suskaičiuotų atrinktų klipų rodymo trukmę.

**U4-12. Lojalūs klientai**

Tekstiniame faile yra lojalių parduotuvės klientų sąrašas: kortelės numeris, sukaupta virtualių pinigų suma, dažniausiai perkamos prekės pavadinimas, nupirktas kiekis, šiai prekei išleista pinigų suma. Parašykite programą, kuri spausdintų lojalių klientų duomenis lentele, surastą didžiausią populiariausiai prekei išleistą sumą, suskaičiuotų vidutinę virtualių pinigų sumą. Papildykite programą veiksmais, kurie leistų pasiūlyti klientams jų populiariausią prekę su nuolaida nuo vidutinės šios prekės kainos. Nuolaida klientui gali būti suteikama, jeigu jo išleista pinigų suma ir nupirktas prekės kiekis viršija nurodytus dydžius. Pastarieji kartu su nuolaidos dydžiu įvedami klaviatūra. Sudarytų nuolaidų sąrašą programa turi spausdinti surikiuotą pagal prekės kainą ir kortelės numerį didėjimo tvarka.

**U4-13. Stipendijos**

Tekstiniame faile pateikta informacija apie studentus. Pirmojoje failo eilutėje nurodytas stipendijų fondo dydis ir pažymių vidurkis stipendijai gauti. Tolimesnėse eilutėse tokia informacija: studento pavardė, vardas, grupė, pažymių kiekis, pažymiai. Studentui skiriama stipendija, jei jo pažymių vidurkis viršija nurodytą dydį ir jis neturi skolų (visi pažymiai >4). Studentui skiriama 10% didesné stipendija, jei jo visi pažymiai didesni už 8. Toks studentas vadinamas pirmūnu. Paskirstykite studentams stipendijas pagal duotą fondą. Fondą reikia maksimaliai išnaudoti, bet negalima viršyti fondo dydžio. Spausdinamas sąrašas turi būti surikiuotas pagal stipendijų dydį, pavares ir vardus. Iš sąrašo pašalinkite studentus, kurie negauna stipendijos. Suformuokite ir atspausdinkite nurodytos grupės (ivedamas klaviatūra) pirmūnų sąrašą.

**U4-14. Įvertinimai**

Tekstiniame faile pateikta informacija apie studentus. Pirmoje failo eilutėje nurodytas fakulteto pavadinimas ir teisingas atsakymas, pvz.: NTNNTNNTT. Vienoje eilutėje yra tokia informacija: studento Nr., studento pavardė, vardas, atsakymas. Reikia surasti, kiek taškų T1 surinko geriausiai atsakęs studentas. Reikia įvertinti visus studentus. Įvertinimai turi būti tokie: jei studento surinktas taškų skaičius lygus T1 arba T1-1, tai įvertinimas „gerai“, jei taškų skaičius lygus T1-2 arba T1-3, tai įvertinimas „patenkinamai“, likusiais atvejais – „nepatenkinamai“. Atspausdinti rezultatus, surikiuotus pagal taškus ir pavares. Sąraše palikti tik tuos, kurie gavo įvertinimą „gerai“, o kitus pašalinti. Atspausdinkite po pašalinimo.

**U4-15. Žaidėjai**

Tekstiniame faile pateikta informacija apie krepšinio komandos žaidėjus. Pirmoje failo eilutėje nurodytas komandos pavadinimas, žaistų rungtynių skaičius. Kitose eilutėse - informacija apie krepšininkus: pavardė, vardas, ūgis, žaista rungtynių, įmesta taškų, padaryta klaidų. Sudaryti sąrašą žaidėjų, kurie žaidė visas komandos rungtynes ir pelnė daugiau taškų, nei komandos žaidėjų taškų vidurkis. Surikiuokite sudarytą sąrašą pagal ūgi ir žaidėjo pavardę. Sąraše palikite tik tuos žaidėjus, kurių ūgis patenka į intervalą [n1, n2], o kitus – pašalinkite iš sąrašo.

**U4-16. Prenumeratoriai**

Tekstiniame faile pateikta informacija apie futbolininkus. Failo eilutėje nurodyta prenumeratoriaus pavaidė, adresas, laikotarpio pradžia (sveikasis skaičius 1..12), laikotarpio ilgis (sveikasis skaičius 1..12), leidinio kodas, leidinių kiekis. Laikotarpio pradžia + laikotarpio ilgis <= 13. Sudaryti sąrašą prenumeratorių, kurie užsisakė leidinius daugiau kaip 1 mėnesiui. Atspausdinti pagal adresus ir

pavardes, vaizduojant užsakytius mėnesius žvaigždutėmis, o neužsakytius - taškais. Išrinkti tarp jų prenumeratorių, kuris užsisakė daugiausiai leidinių.

**U4-17. Futbolininkai**

Tekstiniame faile pateikta informacija apie prenumeratorius: komanda, pavardė, vardas, žaistų rungtynių skaičius, įmuštių įvarčių skaičius. Surasti kiekvienos komandos futbolininką, kurio naudingumo koeficientas didžiausias (žaista ne mažiau kaip vidutinis rungtynių skaičius ir įmušta daugiausiai įvarčių) ir suformuoti iš jų sąrašą. Surikiuoti futbolininkus pagal komandas ir pavaides.

**U4-18. Fakultetai**

Tekstiniame faile pateikta informacija apie studentus. Eilutėje apie studentą yra tokie duomenys: fakultetas, pavardė, vardas, grupė, lytis, dešimtukų, devynetų, aštuonetų kiekiai ir bendras studento laikytų egzaminų kiekis. Nustatyti, kurio fakulteto studentai yra geriausi. Juos atrinkti į atskirą sąrašą. Surikiuoti šio fakulteto studentus pagal grupes ir pavaides, pirmūnus (visi pažymiai > 8) pažymint žvaigždute.

**U4-19. Dėstytojai**

Tekstiniame faile pateikta informacija apie studentų pasirenkamus modulius. Eilutėje yra tokie duomenys: modulio pavadinimas, atsakingo dėstytojo pavardė, vardas, kreditų kiekis, studento pavardė, vardas, grupė. Dėstytojas gali būti atsakingas už keletą moduliu. Suformuokite dėstytojų sąrašą. Sąrašas turi būti surikiuotas pagal pavaides abėcėlės tvarka. Pašalinkite iš šio sąrašo dėstytojus, kurių modulius pasirinko tik po vieną studentą. Suraskite, kuris dėstytojas turi daugiausiai modulių. Sudarykite nurodyto dėstytojo (ivedamas klaviatūra) modulių sąrašą.

**U4-20. Leidiniai**

Tekstiniame faile pateikta informacija apie leidinių prenumerata. Eilutėje yra tokie duomenys: prenumeratoriaus pavaidė, adresas, laikotarpio pradžia (sveikasis skaičius 1..12), laikotarpio ilgis (sveikasis skaičius 1..12), leidinio pavadinimas, vieno mėnesio leidinio kaina. Nustatykite kiekvienam mėnesiui, kurio leidinio pajamos yra didžiausios. Nustatykite bendrasias leidinių pajamas. Sudarykite sąrašą leidinių, kurių pajamos mažesnės už vidutines. Spausdinamas sąrašas turi būti surikiuotas pagal leidinius abėcėlės tvarka. Sudarykite nurodyto leidinio (ivedamas klaviatūra) nurodyto mėnesio (ivedamas klaviatūra) prenumeratorių sąrašą.

**U4-21. Premijos**

Tekstiniame faile pateikta informacija apie mokslių darbuotojų darbo rezultatus. Moksliiniai darbuotojai atliko darbus 4 skirtinėse temose. Pirmoje eilutėje pateikti premijų dydžiai, tolesnėse eilutėse – darbuotojų pavardės, vardai, banko pavadinimas, sąskaitos numeris, darbuotojų indėliai, kurie išreikštį naudingumo koeficientu, į eilinę temą. Remiantis darbuotojų indėliais, suskaičiuokite kiekvienam darbuotojui priklausančios premijos dydį pagal kiekvieną temą atskirai ir bendrą premijų sumą. Darbuotojo indėlis rodo, kokia premijos dalis jam priklauso. Indėlis gali būti išreikštasis bet kokiui skaičiumi. Bet tas pats matas naudojamas visiems darbuotojams. Neišdalinkite pinigų daugiau, negu turite, ir turite išdalinti visus pinigus. Suformuokite sąrašą darbuotojų, kurie uždirbo mažiau už vidurkį. Suformuokite kiekvienam bankui atskirai pavedimų sąrašą.

**U4-22. Grupės**

Tekstiniame faile pateikta informacija apie studentų pasirenkamus modulius. Eilutėje yra tokie duomenys: modulio pavadinimas, atsakingo dėstytojo pavardė, vardas, kreditų kiekis, studento pavardė, vardas, grupė. Dėstytojas gali būti atsakingas už keletą moduliu. Nustatykite, kurio dėstytojo modulius pasirinko daugiausiai studentų. Atspausdinkite gautą rezultatą pagal modulio kreditų kiekį ir pavadinimą. Nustatykite, ar visų grupių studentai pasirinko šio dėstytojo modulius. Atspausdinkite grupes, kurių studentai nepasirinko šio dėstytojo moduliu.

**U4-23. Krepšininkai**

Tekstiniame faile pateikta informacija apie krepšinio komandos žaidėjus. Eilutėje yra tokia informacija apie krepšininkus: komanda, pavardė, vardas, ūgis, gimimo metai, žaidimo pozicija (puolėjas, gynėjas, centras), žaista rungtynių, įmesta taškų. Sudarykite kiekvienos pozicijos geriausiuju žaidėjų sąrašus (taškai/rungtynių skaičius). Iš sąrašų pašalinkite žaidėjus, kurie žaidė ne daugiau kaip vienas rungtynes. Nustatykite, kelių skirtinų komandų žaidėjai užima pirmąsias tris vietas geriausiuju žaidėjų sąrašuose. Atspausdinkite tas komandas.

**U4-24. Darbininkai**

Tekstiniame faile pateikta informacija apie darbininkų darbo rezultatus. Eilutėje yra tokia informacija: įmonė, data (metai, mėnuo, diena), darbininko pavardė, detalės pavadinimas, įkainis, pagamintų vienetų skaičius. Suraskite daugiausiai uždirbusio darbininko pavardę, kiek dienų jis dirbo, kiek iš viso detalij pagamino ir už kokią sumą. Sudarykite tik vieno pavadinimo detales gaminusių darbininkų sąrašą, pagamintų detalij skaičių ir sumą. Realizuokite duomenų atrinkimą į kitą rinkinį pagal nurodytą požymį (pagamintų vienetų skaičius > S, įkainis < K, įvedami klaviatūra). Spausdinamas sąrašas turi būti surikiuotas pagal pavardes abėcėlės tvarka.

**5. Teksto analizė ir redagavimas**

Susipažinsite su:

- string klasės kintamaisiais.
- Klasės string metodais.
- Žodžių išskyrimu.
- Teksto analizės ir redagavimo elementais.

**5.1. Eilutės simbolių analizė**

- Raidžių pasikartojimų radimas.
- Raidžių konversijos funkcijų naudojimas.

**Užduotis.** Raidės.

Tekstiniame faile duotas tekstas. Parašykite programą, kuri surastų, kiek kartų tekste sutinkamos didžiosios ir mažosios raidės.

Pradiniai duomenys			
Petriukas, programuotojo sūnus, eina į pirmą klasę.			
Mokytoja vaikams aiškina:			
- Vaikai, pamokose reikia būti tyliems, jei norësite kažko paklausti, pirma pakelkite ranką.			
Petriukas iškart pakelia ranką. Mokytoja klausia:			
- Na, Petriuk, ką norėjai paklausti?			
- Nieko. Tiesiog testuoju sistemą.			
Rezultatai			
a	31		A 0
b	1		B 0
c	0		C 0
d	0		D 0
e	16		E 0
f	0		F 0
g	2		G 0
h	0		H 0
i	28		I 0
j	6		J 0
k	24		K 0
l	7		L 0
m	7		M 2
n	7		N 2
o	16		O 0
p	8		P 3
q	0		Q 0
r	13		R 0
s	16		S 0
t	16		T 1
u	10		U 0
v	1		V 1
w	0		W 0
x	0		X 0
y	3		Y 0
z	0		Z 0

**Programos kūrimo eiga.**

- Parašoma funkcija, kuri suranda duotoje simbolių eilutėje raidžių pasikartojimų skaičius.
- Parašoma funkcija, kuri tekste suranda raidžių pasikartojimų skaičius.
- Parašoma funkcija, kuri spausdina gautus rezultatus.

**OPirmas žingsnis.**

- Parašykite funkciją, kuri surastų duotoje simbolių eilutėje raidžių pasikartojimų skaičius:

```
//-----
const char CDrv[] = "Duomenys.txt";
const char CRfv[] = "Rezultatai2.txt";
```

```

const int CMax = 256;
//-----
void Kiek(string E, int R[]);
//-----
int main()
{
    int R[CMax]; // simboliu pasikartojimai
    for (int i = 0; i < CMax; i++) R[i] = 0;
    Kiek("Kaunas yra Kaunas", R);
    return 0;
}
//-----
// Simbolių eilutėje E skaičiuojamas raidžių pasikartojimas
// Pasikartojimai kaupiami masyve R
// Masyve R atžymos daromos, panaudojant simbolį kaip indeksą
void Kiek(string E, int R[])
{
    for (int i = 0; i < E.length(); i++) {
        if ('a' <= E[i] && E[i] <= 'z') ||
            ('A' <= E[i] && E[i] <= 'Z'))
            R[E[i]]++;
    }
}
//-----


- Patikrinkite, kaip dirba programa. Tam papildomai reikia funkcijoje main() parašyti masyvo R spausdinimo sakinus.

```

#### ① Antras žingsnis.

- Parašykite teksto analizės funkciją:

```

//-----
// R - simbolių pasikartojimų tekste sąrašas
// Simbolio skaitinė reikšmė naudojama masyve indeksu
void Tekstas(int R[])
{
    string eil;
    ifstream fd(CDfv);
    while (!fd.eof()) {
        getline(fd, eil);
        Kiek(eil, R);
    }
    fd.close();
}
//-----


- Papildykite funkciją main() krepiniu į teksto analizės funkciją.

```

```

void Kiek(string E, int R[]);
void Tekstas(int R[]);
//-----
int main()
{
    int R[CMax]; // simbolių dažnai
    for (int i = 0; i < CMax; i++)
        R[i] = 0;
    Tekstas(R);
    return 0;
}
//-----


- Patikrinkite, kaip dirba programa. Tam papildomai reikia funkcijoje main() parašyti masyvo R spausdinimo sakinus.

```

#### ② Trečias žingsnis.

- Parašykite teksto analizės rezultatų spausdinimo failo funkciją:

```
//-----
```

```

void Spausdinti(int R[])
{
    ofstream fr(CRfv);
    for (char sim = 'a'; sim <= 'z'; sim++)
        fr << sim << setw(4) << R[sim] << " " | "
            << (char) toupper(sim) << setw(4) << R[toupper(sim)] << endl;
    fr.close();
}
//-----
```

- Papildykite `main()` funkciją krepiniu į teksto analizės rezultatų spausdinimo failo funkciją.
- Patikrinkite, kaip dirba programa.
- Patikrinkite programos darbą su skirtingais duomenimis. Pavyzdžiu, kaip programa dirba, jeigu tekste yra tik viena raidė? Arba kai tekste nėra raidžių.

#### Programos papildymas.

- Papildykite programą veiksmais, kuriais rastumėte dažniausiai vartojamą tekste raidę.
- Papildykite programą veiksmais, kuriais rezultatai būtų pateikiami raidžių pasikartojimų skaičių mažėjimo tvarka.

#### Savarankiško darbo užduotis.

Lietuviškuose tekstuose naudojamos lietuviškos abécélės raidės, kurių nėra lotyniškoje abécélėje. Rašydami programos papildymą įvertinkite, kad lietuviškos raidės simbolių lentelėje nėra surašytos eilės tvarka.

## 5.2. Teksto eilučių šalinimas

- Teksto eilučių analizė, naudojant klasės `string` metodą `length()`.
- Ilgiausios eilutės paieška.
- Nurodytos eilutės pašalinimas iš teksto.

#### Užduotis. Ilgiausia teksto eilutė.

Tekstiniame faile duotas tekstas. Tekstą sudaro viena ir daugiau eilučių. Parašykite programą, kuri pašalintų iš teksto ilgiausią eilutę.

#### Pradiniai duomenys ir rezultatai.

Pradiniai duomenys
Petriukas, programuotojo sūnus, eina į pirmą klasę. Mokytoja vaikams aiškina: - Vaikai, pamokose reikia būti tyliems, jei norėsite kažko paklausti, pirma pakelkite ranką. Petriukas iškart pakelia ranką. Mokytoja klausia: - Na, Petriuk, ką norėjai paklausti? - Nieko. Tiesiog testuoju sistemą.
Rezultatai
Petriukas, programuotojo sūnus, eina į pirmą klasę. Mokytoja vaikams aiškina: pakelkite ranką. Petriukas iškart pakelia ranką. Mokytoja klausia: - Na, Petriuk, ką norėjai paklausti? - Nieko. Tiesiog testuoju sistemą.

#### Programos kūrimo eiga.

- Parašoma funkcija, kuri suranda tekste ilgiausią eilutę.
- Parašoma funkcija, kuri iš teksto pašalina nurodytą eilutę. Šalinimas vykdomas perrašant duotą tekštą į naują failą. Šalinamoji eilutė neperrašoma.

#### ① Pirmas žingsnis.

- Parašykite funkciją, kuri duotame tekste surastą ilgiausią eilutę:

```
//-----
```

```
int Rasti();
```

```

int main()
{
    cout << Rasti();
    return 0;
}
-----
int Rasti()
{
    int nr = 0; string ilg = "";
    int n = 0; string eil;
    ifstream fd(Fd);
    while (!fd.eof()) {
        getline(fd, eil);
        if (eil.length() > ilg.length()) {
            nr = n; ilg = eil;
        }
        n++;
    }
    fd.close();
    return nr;
}
-----
```

- Patikrinkite, kaip dirba programa. Ekrane turi būti matomas skaičius. Tai ilgiausios eilutės numeris. Duotam duomenų pavyzdžiui tai turi būti 2, kai eilutės numeruojamos pradedant 0.

#### ¶ Antras žingsnis.

- Parašykite funkciją, kuri duotame tekste pašalintų nurodytą eilutę:

```

void Mesti( int k)
{
    string eil;
    ifstream fr("Rezultatai.txt");
    ifstream fd("Duomenys.txt");
    // perrašomos eilutės iki šalinamosios
    for (int i = 0; i < k; i++) {
        getline(fd, eil);
        fr << eil << endl;
    }
    // šalinamoji eilutė neperrašoma
    // perrašomos likusios eilutės
    while (!fd.eof()) {
        getline(fd, eil);
        fr << eil << endl;
    }
    fd.close();
    fr.close();
}
-----
```

- Papildykite `main()` funkciją, kreipiniu į eilutės šalinimo funkciją.

```

int main()
{
    Mesti(Rasti());
    return 0;
}
-----
```

- Patikrinkite, kaip dirba programa. Rezultatų failo turite matyti duoto teksto kopiją be ilgiausios eilutės.

#### Programos patikrinimas.

- Patikrinkite, kaip dirba programa, kai tekstą sudaro tik viena eilutė.

- Patikrinkite, kaip dirba programa, kai tekste yra kelios vienodos ilgiausios eilutės.

#### Programos papildymas.

- Pakeiskite programą taip, kad būtų šalinamos visos ilgiausios eilutės, jeigu jų yra ne viena.

#### Savarankiško darbo užduotis.

Duotame tekste reikia pašalinti tuščias eilutes.

### 5.3. Teksto eilučių dalių šalinimas

- Teksto šalinimas, panaudojant `string` metodą `erase()`.

**Užduotis.** C++ vienos eilutės komentavimas `//`.

Tekstiniame faile duotas C++ programos tekstas. Pašalinkite iš teksto komentarirus, kurie žymimi `//`.

#### Pradiniai duomenys ir rezultatai.

Pradiniai duomenys
<pre> void DuomenysInternet(Grybai &amp; grybai) {     ifstream fd(u2);     // string pav, tip;     // GrybasInfo s1;     int ns = 0;     bool yra = true;     while(!fd.eof() &amp;&amp; yra) { // kol yra duomenų ir jie telpa į masyva         fd &gt;&gt; pav &gt;&gt; tip;         s1.Deti (pav, tip);         if(!fd.eof() &amp;&amp; (ns - 1 &lt; Grybai::CMax ) )             grybai[ns++] = s1; // išrašo nauja elementą         else             yra = false;     }      fd.close();     grybai.Deti(ns); }</pre>

Rezultatai
<pre> void DuomenysInternet(Grybai &amp; grybai) {     ifstream fd(u2);     int ns = 0;     bool yra = true;     while(!fd.eof() &amp;&amp; yra) {         fd &gt;&gt; pav &gt;&gt; tip;         s1.Deti (pav, tip);         if(!fd.eof() &amp;&amp; (ns - 1 &lt; Grybai::CMax ) )             grybai[ns++] = s1;         else             yra = false;     }      fd.close();     grybai.Deti(ns); }</pre>

#### Programos kūrimo eiga.

- Parašoma funkcija, kuri įveda teksto eilutes, jas analizuoją ir formuoja rezultatų failą. Ši funkcija kreipiasi į eilutės analizės funkciją.
- Parašoma funkcija, kuri analizuoją eilutę ir pašalina rastus komentaritus.

#### ¶ Pirmas žingsnis.

- Parašykite funkciją, kuri įvestų duomenis ir formuotų analizės bei rezultatų failus:

```

//-----
void ApdorotiTeksta(const char dfv[], const char rfv[])
{
    ifstream fd(dfv);
    ofstream fr(rfv);
    string eil;
    while (!fd.eof()) {
        getline(fd, eil);
        if (eil.length() > 0) {
            MestiKomentarus(eil);
            if (eil.length() > 0)
                fr << eil << endl;
        }
    }
    fd.close();
    fr.close();
}
//-----

```

#### ¶ Antras žingsnis.

- Parašykite funkciją, kuri ieško ir šalina C++ komentarus, kurie žymimi //:

```

void MestiKomentarus(string & eil)
{
    for (unsigned int i = 0; i < eil.length() - 1; i++)
        if (eil[i] == '/' & eil[i+1] == '/')
            eil.erase(i);
    return;
}

• main() funkcija bus tokia:
//-----

const char CDuom[] = "Duomenys.txt";
const char CRez[] = "Rezultatai.txt";
#include <string>
#include <fstream>
#include <iomanip>
using namespace std;
//-----
void ApdorotiTeksta(const char dfv[], const char rfv[]);
void MestiKomentarus(string & eil);
//-----
int main()
{
    ApdorotiTeksta(CDuom, CRez);
    return 0;
}
//-----

```

- Patikrinkite, kaip veikia programa. Jei būsite atidūs, pastebėsite, jog rezultatų faile nėra tuščios eilutės, kuri pradiniame faile buvo trečia nuo apačios. Papildykite funkciją ApdorotiTeksta(), kad tuščios eilutės neprarastumėte.

#### Savarankiško darbo užduotis.

Duotame C++ programos tekste pašalinkite visus komentarus /\* \*/, //)

### 5.4. Žodžių išskyrimas eilutėje

- Žodžių išskyrimas eilutėje, panaudojant klasės string metodus find\_first\_not\_of(), find\_first\_of() ir substr().

#### Užduotis. Žodžių išskyrimas ir analizė.

Rasti, kiek tekste yra žodžių, kurių pirma ir paskutinė raidės vienodos. Žodžiai skiriami skyrikliais.

Pradiniai duomenys	Rezultatai
V.M. Putinas Margi sakalai  Lydédami gęstančią žarą vélai Pakilo į dangų margi sakalai. Paniekinę žemės vylingus sapnus, Padangéje ištisésé savo sparnus. Ir taré margieji: negrišim į žemę, Kol josios kalnai ir pakalnés aptemę. ...	Žodžių skaičius: 6

#### Programos kūrimo eiga.

- Skaitomas teksto eilutės.
- Išskiriami žodžiai ir skaičiuojamas žodžių, kurių pirma ir paskutinė raidė vienodos, skaičius eilutėje.

#### ¶ Pirmas žingsnis.

- Parašykite funkciją, kuri skaičiuoja, kiek eilutėje yra žodžių, kurių pirma ir paskutinė raidė vienodos. Po paskutinio žodžio eilutėje gali nebūti skyriklio. Eilutė gali būti tuščia. Funkcija gali būti tokia:

```

//-----
// Suranda ir gražina žodžiu,
// kurie prasideda ir pasibaigia ta pačia raide skaičiu eilutėje
// eil - eilutė, kurioje ieškomi žodžiai
// skirt - skyriklių eilutė
int KiekisŽodžiųPrIrrPbSimbVienodi(string eil, string skirt)
{
    int kiekSutampa = 0;
    eil = eil + " ";
    int zpr, zpb = 0; // žodžio pradžia ir pabaiga eilutėje
    zpr = eil.find_first_not_of(skirt, zpb);
    while (zpr != -1) {
        zpb = eil.find_first_of(skirt, zpr);
        cout << eil.substr(zpr, zpb - zpr) << endl; // žodžio išskyrimas ir išvestis
        if (eil[zpr] == eil[zpb-1])
            kiekSutampa++;
        zpr = eil.find_first_not_of(skirt, zpb);
    }
    return kiekSutampa;
}
//-----

```

- Patikrinkite, kaip dirba programa su viena bandomaja eilute:

```

int main()
{
    string skirt = " .,:;-!?\t"; // skyrikliai tarp žodžių eilutėje
    string bandEil = "Ji atsisveikina: iki pasimatymo, iki!";
    cout << KiekisŽodžiųPrIrrPbSimbVienodi(bandEil, skirt) << endl;
    return 0;
}

```

- Ekrane turite matyti atspausdintus stulpeliu žodžius ir skaičių 3.
- Pakeiskite bandomają eilutę, palikdami vieną žodį, vieną raidę ir pan. Patikrinkite, ar gerai dirba sudaryta funkcija.

#### ¶ Antras žingsnis.

- Parašykite funkciją, kuri skaičiuoja, kiek visame tekste yra žodžių, kurių pirma ir paskutinė raidė vienodos.

```

//-----
// Teksto skaitymas po vieną eilutę ir analizavimas:
// suskaičiuoja ir gražina kiek tekste yra žodžių,
// kurie prasideda ir pasibaigia ta pačia raide
// dfv - teksto failo vardas
// skirt - skirtukų eilutė
int TekstoAnalize(string dfv, string skirt)

```

```

{
    int kiekSutampa = 0;
    ifstream fd(dfv.c_str());
    string eil; // eilutė, įvedama iš srauto
    while (!fd.eof()) {
        getline(fd, eil);
        kiekSutampa += KiekisŽodžiuPrIrPbSimbVienodi(eil, skirt);
    }
    fd.close();
    return kiekSutampa;
}
//-----
• Patikrinkite kaip dirba programa su tekstu, esančiu failė Duomenys.txt:
//-----
int main()
{
    string skirt = " .,:-!?\t"; // skirtukai tarp žodžių eilutėje
    cout << "Žodžių skaičius: " << TekstoAnalize("Duomenys.txt", skirt) << endl;
    return 0;
}
//-----

```

#### Programos papildymas.

- Papildykite funkciją KiekisŽodžiuPrIrPbSimbVienodi() atvejui, kai žodis prasideda ir pasibaigia skirtingo dydžio (kodo) raidėmis (pvz.: 'A' ir 'a'). Atsižvelkite tik į lotyniškos abécélės raides.

#### Savarankiško darbo užduotis.

Parašykite programą, kuri surastų kiek tekste yra žodžių palindromų, vienodai skaitomų iš abiejų pusių, pvz., „sūnūs“, „émé“, „iki“ ir pan.

### 5.5. Eilučių redagavimas

- Eilutės redagavimas, panaudojant klasės string metodą insert().

#### Užduotis. Žodžių išskyrimas ir eilutės redagavimas.

Tekste visus vardus (pvz., Arvydas) papildyti nurodyta pavarde (pvz., SABONIS). Žodžiai skiriami skyrikliais.

Pradiniai duomenys
Arvydas (g. 1964 m. gruodžio 19 d. Kaune) – Lietuvos krepšininkas, olimpinis ir pasaulio čempionas, nuo 2011 m. spalio 24 d. Arvydas Lietuvos krepšinio federacijos prezidentas. Profesionalaus žaidėjo karjerą Arvydas pradėjo 1981 m. Kauno krepšinio klube "Žalgiris". Tris sezonus iš eilės (1985–1987 m.) Arvydas padėjo komandai iškovoti SSRS krepšinio čempionato aukso medalius. 1982 m. SSRS rinktinės sudėtyje Arvydas dalyvavo pasaulio krepšinio čempionate ir laimėjo auksą.

Rezultatai
ArvydasSABONIS (g. 1964 m. gruodžio 19 d. Kaune) – Lietuvos krepšininkas, olimpinis ir pasaulio čempionas, nuo 2011 m. spalio 24 d. ArvydasSABONIS Lietuvos krepšinio federacijos prezidentas. Profesionalaus žaidėjo karjerą ArvydasSABONIS pradėjo 1981 m. Kauno krepšinio klube "Žalgiris". Tris sezonus iš eilės (1985–1987 m.) ArvydasSABONIS padėjo komandai iškovoti SSRS krepšinio čempionato aukso medalius. 1982 m. SSRS rinktinės sudėtyje ArvydasSABONIS dalyvavo pasaulio krepšinio čempionate ir laimėjo auksą.

#### Programos kūrimo eiga.

- Skaitomos teksto eilutės.
- Išskiriame žodžiai lyginami su nurodytu žodžiu (vardu) ir, jeigu reikia, papildomi kitu žodžiu (pavarde).

#### OPirmas žingsnis.

- Parašykite funkciją, kuri eilutėje išskiria žodžius, juos lygina su duotu žodžiu (vardas) ir jeigu reikia papildo kitu nurodytu žodžiu (pavarde). Po paskutinio žodžio eilutėje gali nebūti skyriklis. Eilutė gali būti tuščia. Funkcija gali būti tokia:

```

//-----
// Šalia nurodyto(u) vardo(u) užrašo nurodyta(as) pavarde(es)
// eil - eilutė, kurioje atliekami nurodyti veiksmai
// skirt - skirtukų eilutė
// vard - vardas
// pav - pavarde
void VardoPapildymasPavarde(string & eil, string skirt, string vard, string pav)
{
    eil = eil + " ";
    int zpr, zpb = 0; // žodžio pradžia ir pabaiga eilutėje
    zpr = eil.find_first_not_of(skirt, zpb);
    while (zpr != -1) {
        zpb = eil.find_first_of(skirt, zpr);
        int zilg = zpb - zpr;
        if (eil.substr(zpr, zilg) == vard) {
            eil.insert(zpb, pav);
            zpb = zpb + pav.length() + 1;
        }
        zpr = eil.find_first_not_of(skirt, zpb);
    }
    eil.erase(eil.length()-1, 1);
}
//-----

```

- Patikrinkite, kaip dirba programa su viena bandomaja eilute:

```

//-----
int main()
{
    string skirt = " .,:-!?\t"; // skirtukai tarp žodžių eilutėje
    string bandEil = "Arvydas - Lietuvos krepšininkas Nr.1.";
    VardoPapildymasPavarde(bandEil, skirt, "Arvydas", "Sabonis");
    cout << eil << endl;
    return 0;
}
//-----

```

- Ekrane turite matyti:

ArvydasSabonis – Lietuvos krepšininkas Nr.1.

- Pakeiskite bandomają eilutę, palikdami vieną žodį (vardą), daug vienodų žodžių (vardų) ir pan. Patikrinkite, ar gerai dirba sudaryta funkcija.

#### OAntras žingsnis.

- Parašykite funkciją, kuri visame tekste atlieka aukščiau nurodytus veiksmus – žodžių (vardų) papildymą.

```

//-----
// Teksto skaitymas po viena eilute ir redagavimas:
// prie surasto žodžio (vardo) užrašant kita žodį (pavarde)
// dfv - pradinio teksto failo vardas
// rfv - pakeisto teksto failo vardas
// skirt - skirtukai tarp žodžių
// vard - vardas
// pav - pavarde
void TekstoRedagavimas(string dfv, string rfv, string skirt,
                        string vard, string pav)
{
    ifstream fd(dfv.c_str());
    ofstream fr(rfv.c_str());
    string eil; // eilutė, įvedama iš srauto
    while (!fd.eof()) {
        getline(fd, eil);

```

```

VardoPapildymasPavarde(eil, skirt, vard, pav);
fr << eil << endl;
}
fd.close();
fr.close();
}
//-----
• Patikrinkite, kaip dirba programa su tekstu, esančiu failu Duomenys.txt:
//-----
int main()
{
    string skirt = " .,:;-!?\t"; // skirtukai tarp žodžių eilutėje
    TekstoRedagavimas("Duomenys.txt", "Rezultatai.txt", skirt,
                       "Arvydas", "SABONIS");
    return 0;
}
//-----

```

#### Programos papildymas.

- Papildykite funkciją VardoPapildymasPavarde() taip, kad papildomas žodis būtų įrašomas, paliekant tarp žodžių tarpeļį.

#### Savarankiško darbo užduotis.

- Parašykite programą, kuri pašalintų iš teksto nurodytus žodžius kartu su už jų esančiais skyrikliais.

## 5.6. Analizės failo sukūrimas

- Analizės failo, kuris padėtų sekti teksto keitimo eiga, sukūrimas.
- Klasės string konstruktorius panaudojimas eilutės užpildymui simboliais – linijos brėžimui.

**Užduotis.** Analizės failo, kuriame bus atspindėti tarpiniai veiksmai, sukūrimas.

Tekstiniame faile duotas tekstas. Žodžiai iš eilutės į eilutę nekeliami. Skyrikliai žinomi. Pašalinti iš kiekvienos eilutės ilgiausio žodžio (vieno) visas balses.

Pradiniai duomenys		
Kūčių ryta		
Anksti Kūčių ryta šeimininkė budina savo vyra:		
- Eik greičiau, saulei netekėjus, kur dalgės kabo, ištverk dalges. Dalges padék po stogu, o dalgiakočius sudék svirnan.		
Šeimininkė ieško kubilo lanko, kad būt visai apskritas, nepertrūkės niekur. Ta lanka neša vištų tvartan, vidury tvarto paguldo. O tada šeimininkė skuba tvartant prie kodžio, kur būna žirniai supilti. Šeimininkė tuos žirnius semia negailėdama didžiuli gorčiu, kad visos vištros prilestų lig soties. Šeimininkė pila tuos žirnius tan kubilo lankan, kad nei vienas žirnis nebūt už kubilo lanko - kad vištros visos dėtu kiaušinius vienan daiktan, nemėtytu kiaušinių. Berdama žirnius tan lankan, šeimininkė garsiai sako vištoms:		
- Žiūrėkite, kad nei vieno kiaušinio, nei vieno niekur nepamestut, visus vienon vieton dékite!		
Na ir visos vištros šeimininkės įsakymą vykdo.		
Analizė		
Ilgiausias žodis   Pradžia   Ilgis		
Kūčių	0	5
Šeimininkė	20	10
netekėjus	25	9
dalgiakočius	18	12
nepertrūkės	58	11
Šeimininkė	59	10
Šeimininkė	48	10
negailėdama	0	11
žirnius	10	7
kiaušinius	30	10
Šeimininkė	28	10

nepamestut	56	10
vienon	0	6
šeimininkės	19	11

#### Rezultatai

##### Kč ryta

Anksti Kūčių ryta šmnnk budina savo vyra:

- Eik greičiau, saulei ntkjs, kur dalgės kabo, ištverk dalges. Dalges padék po stogu, o dlkcs sudék svirnan.

Šeimininkė ieško kubilo lanko, kad būt visai apskritas, nprtrks niekur. Ta lanka neša vištų tvartan, vidury tvarto paguldo. O tada šmnnk skuba tvartant prie kodžio, kur būna žirniai supilti. Šmnnk tuos žirnius semia ngldm didžiuli gorčiu, kad visos vištros prilestų lig soties. Šeimininkė pila tuos žrns tan kubilo lankan, kad nei vienas žirnis nebūt už kubilo lanko - kad vištros visos dėtu kšns vienan daiktan, nemėtytu kiaušinių. Berdama žirnius tan lankan, šmnnk garsiai sako vištom:

- Žiūrėkite, kad nei vieno kiaušinio, nei vieno niekur nprmstt, visus vnn vieton dékite!

Na ir visos vištros šmnnks įsakymą vykdo.

#### Programos kūrimo eiga.

- Sukuriamos trys funkcijos: ApdorotiTeksta(), AnalizuotiEilute(), RedaguotiEilute().
- Funkcija ApdorotiTeksta() skaito tekstą po eilutę ir kviečia funkcijas: AnalizuotiEilute() ir RedaguotiEilute().
- Funkcija AnalizuotiEilute() ieško eilutėje ilgiausio žodžio ir grąžina jį, jo pradžios numerį eilutėje ir žodžio ilgi per parametrus.
- Funkcija RedaguotiEilute() pagal gautus ilgiausio žodžio parametrus eilutėje iš ilgiausio žodžio šalina balses.

#### ¶ Pirmas žingsnis.

- Parašykite pagrindinę funkciją. Paruoškite 3 failus: duomenų, analizės ir rezultatų. Parašykite funkciją ApdorotiTeksta(), kurios parametrais būtų paruoštu failų vardai, kuri skaitytų eilutes iš duomenų failo ir rašytų į rezultatų failą. Funkcijos turėtų atrodyti taip:

```

//-----
const char Cduom[] = "Duomenys.txt";
const char Crez[] = "Rezultatai.txt";
const char Canalize[] = "Analize.txt";
#include <string>
#include <fstream>
#include <iomanip>
using namespace std;
//-----
void ApdorotiTeksta(const char dfv[], const char rfv[], const char afv[]);
//-----
int main()
{
    ApdorotiTeksta(Cduom, Crez, Canalize);
    return 0;
}
//-----
// Teksto skaitymas po eilutę
// dfv - pradinio teksto failo vardas, rfv - pakeisto teksto failo vardas
// afv - analizės rezultatų failo vardas
void ApdorotiTeksta(const char dfv[], const char rfv[], const char afv[])
{
    ifstream fd(dfv);
    ofstream fr(rfv);
    ofstream fa(afv);
    string E;
    while (!fd.eof()) {
        getline(fd, E);
        fr << E << endl;
    }
}

```

```

    }
    fd.close();
    fr.close();
    fa.close();
}
//-----

```

### ¶ Antras žingsnis.

- Parašykite funkciją `AnalizuotiEilute()`, kuri eilutėje ieškotų ilgiausio žodžio. Papildykite funkciją `ApdorotiTeksta()` kreipiniu į šią funkciją ir analizės failo formavimu. Nepamirškite iðėti apsaugos dël galimų tuščių eilučių ar eilučių, kuriose nera žodžių, o yra tik skyrikliai. Funkcija `AnalizuotiEilute()` turėtų atrodyti taip:

```

//-----
// Randa ilgiausią žodį eilutėje, jo pradžią ir ilgi
// eil - eilutė, kurioje atliekama paieška
// IlgZodis - gražinamas ilgiausias žodis eilutėje
// IlgZPradzia - gražinama jo pradžia
// IlgZIlgis - gražinamas jo ilgis
void AnalizuotiEilute(string eil, string & IlgZodis, int & IlgZPradzia,
{
    unsigned int & IlgZIlgis)
{
    string Skirt = " .,!?:;()\t"; // skirtukai tarp žodžiu
    string Zodis;
    int zpr = 0, zpb = 0;
    IlgZodis = "";
    IlgZPradzia = 0; IlgZIlgis = 0;
    while ((zpr = eil.find_first_not_of(Skirt, zpb)) != string::npos) {
        zpb = eil.find_first_of(Skirt, zpr);
        Zodis = eil.substr(zpr, zpb - zpr);
        if (Zodis.length() > IlgZIlgis) {
            IlgZodis = Zodis;
            IlgZPradzia = zpr;
            IlgZIlgis = Zodis.length();
        }
    }
}
//-----

```

- Analizės faile rezultatus kaupsime lentelėje. Lentelės linijų brėžimui skelbiame `string` kintamajį su konstruktoriaumi:

```
string bruksnys(38, '-');
```

- Toks paskelbimas užpildo eilutės kintamajį 38 brūkšneliais. Privalumai yra tokie:
  - nereikia brėžti ilgos brūkšninės linijos;
  - kintamajį paskelbėme vieną kartą, o galėsime naudoti tiek kartų, kiek reikia.

- Analizės failo formavimas turėtų atrodyti taip:

```

fa << bruksnys << endl;
fa << "| Ilgiausias žodis | Pradžia | Ilgis |\n";
fa << bruksnys << endl;
fa << "| " << left << setw(16) << IlgZodis << " | "
    << right << setw(7) << IlgZPradzia;
fa << " | " << setw(5) << IlgZIlgis << " |\n";
fa << bruksnys << endl;

```

- Iterpkite analizės failo formavimo tekstą į tinkamas funkcijos `ApdorotiTeksta()` vietas. Nepamirškite deklaruoti kintamujų.

### ¶ Trečias žingsnis.

- Parašykite funkciją `RedaguotiEilute()`, kuri eilutėje pagal gautus ilgiausio žodžio parametrus pašalintų iš ilgiausio žodžio balses. Papildykite funkciją `ApdorotiTeksta()` kreipiniu į šią funkciją. Funkcija `RedaguotiEilute()` turėtų atrodyti taip:

```

//-----
// Šalina balses iš ilgiausio žodžio
// eil - eilutė, kuri yra keičiamā

```

```

// IlgZPradzia - ilgiausio eilutės žodžio pradžia
// IlgZIlgis - ilgiausio eilutės žodžio ilgis
void RedaguotiEilute(string & eil, int IlgZPradzia, int IlgZIlgis)
{
    string Balses = "AEIYOUaeiouAaEęĘéłłIūū";
    int i = IlgZPradzia;
    while (i < IlgZPradzia + IlgZIlgis) {
        if (Balses.find_first_of(eil[i]) != string::npos) {
            eil.erase(i, 1);
            IlgZIlgis--;
        }
        else
            i++;
    }
}
//-----

```

## 5.7. Kontroliniai klausimai

1. Ką suskaičiuos tolimesnė funkcija?

```

int Eilučių (string fv)
{
    string Balsės("aeiou");
    ifstream f(fv.c_str());
    int k = 0; string E;
    while (!f.eof()) {
        getline(f, E);
        if (E.length() > 0)
            if (Balsės.find(E[E.length()-1]) != -1)
                k++;
    }
    f.close();
    return k;
}

```

Duomenų failas yra tokis:

Ryte  
Anksti ryte budina  
- Kelkis. Esi greitas

2. Ką suskaičiuos tolimesnė funkcija?

```

int Eilute(string E, string Skyr)
{
    string Sk("0123456789");
    int zpr, zpb = 0, k = 0; string E1 = E + " ";
    while ((zpr = E1.find_first_not_of(Skyr, zpb)) != -1) {
        zpb = E1.find_first_of(Skyr, zpr);
        string S(E1, zpr, zpb-zpr);
        if (S.find_first_not_of(Sk) == -1)
            k++;
    }
    return k;
}

```

Duomenų eilutė yra tokia:

Anksti ryte 11 12 13 14 15 16 budinal7. Esi

3. Kokios funkcijos naudojamos žodžio išskyrimui eilutėje?

4. Kaip reikia kreiptis į eilutę, norint išrinkti vieną eilutės simbolį?

5. Koks tipas yra naudojamas kintamojo, kuris galėtų saugoti tik vieną simbolį, paskelbimui?

6. Kokiu būdu galėtume patalpinti simbolį į nurodytą eilutės vietą?

7. Užrašykite ne mažiau trijų skirtingų būdų, kaip suteikti pradines reikšmes eilutei.

8. Ką vadiname žodžiu?

9. Ar galima lyginti dvi eilutes tarpusavyje? Jei taip, pateikite pavyzdį su dviem eilutėmis ir pasakykite, kuri eilutė yra didesnė.

10. Ar žodžių įvedimui iš failo galima naudoti srauto nukreipimo simbolį (>>)? Jei taip, paaiškinkite, kada tai galima daryti.

## 5.8. Užduotys

### U5-1. Ilgiausias žodis

Tekstiniame faile pateikiamas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Žodžiai eilutėse skiriami bent vienu tarpu. Tarpai gali būti eilutės pradžioje bei gale, gali būti tuščios eilutės. Pašalinkite kiekvienos eilutės ilgiausią žodį. Jei yra keli ilgiausi žodžiai, tuomet reikia pašalinti juos visus.

### U5-2. Nelyginis žodžių skaičius

Tekstiniame faile pateikiamas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Žodžiai eilutėse skiriami bent vienu tarpu. Tarpai gali būti eilutės pradžioje bei gale, gali būti tuščios eilutės. Eilutėse, kuriose yra nelyginis žodžių skaičius n, n / 2 + 1 žodį pakeisti žodžiu „xxooxx“.

### U5-3. Žodžių poros

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Kiek yra žodžių porų, kuriose vieno žodžio paskutinė raidė sutampa su sekančio žodžio pirmaja raide? Kiekvienoje eilutėje raskite ilgiausią (daugiausia simbolų) iš šių porų ir perkelkite jos antrajį žodį į eilutės pradžią kartu su už jo esančiais skyrikliais.

### U5-4. Palindromas

Tekstiniame faile pateikiamas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Žodžiai eilutėse skiriami bent vienu tarpu. Tarpai gali būti eilutės pradžioje bei gale, gali būti tuščios eilutės. Po eilutėmis, kuriose yra vienodai iš abiejų pusių skaitomu žodžiu (palindromu), išterpti eilutę, kurioje išrašytas sakiny „Buvo palindromu“.

### U5-5. Nelyginis simbolių skaičius

Tekstiniame faile pateikiamas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Žodžiai eilutėse skiriami bent vienu tarpu. Tarpai gali būti eilutės pradžioje bei gale, gali būti tuščios eilutės. Pašalinkite žodžius, sudarytus iš nelyginio simbolių skaičiaus.

### U5-6. Lyginis žodžių skaičius

Tekstiniame faile pateikiamas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Žodžiai eilutėse skiriami bent vienu tarpu. Tarpai gali būti eilutės pradžioje bei gale, gali būti tuščios eilutės. Eilutėse, kuriose yra lyginis žodžių skaičius n, po n / 2 žodžio išterpti žodį „aaabbbaaa“.

### U5-7. Lyginis simbolių skaičius

Tekstiniame faile pateikiamas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Žodžiai eilutėse skiriami bent vienu tarpu. Tarpai gali būti eilutės pradžioje bei gale, gali būti tuščios eilutės. Po žodžiu, kuriuos sudaro lyginis simbolių skaičius, išterpti tekštą „(LYGINIS)“.

### U5-8. Ilgesni žodžiai

Tekstiniame faile pateikiamas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Žodžiai eilutėse skiriami bent vienu tarpu. Tarpai gali būti eilutės pradžioje bei gale, gali būti tuščios eilutės. Kiekvienos eilutės pradžioje parašyti, kiek eilutėje yra žodžiu, ilgesniu už pirmajį teksto žodį. Teksto pabaigoje naujoje eilutėje parašyti eilutės, kurioje daugiausiai žodžių, kopiją. Jei yra kelios eilutės su vienodu didžiausiu žodžių skaičiumi, reikia parašyti jų visų kopijas.

### U5-9. Vienodi simboliai

Tekstiniame faile pateikiamas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Žodžiai eilutėse skiriami bent vienu tarpu. Tarpai gali būti eilutės pradžioje bei gale, gali būti tuščios eilutės. Po kiekvieno žodžio skliausteliuose užrašyti to žodžio ilgi ir kiek vienodų simbolių yra tame žodyje. Didžiosios ir mažosios raidės laikomos vienodais simboliais.

### U5-10. Skaitmenys

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Kiekvienoje eilutėje po vieną teksto žodį, sudarytą tik iš skaitmenų, jei toks yra, kartu su už jo esančiais skyrikliais perkelkite į eilutės pradžią.

### U5-11. Skirtingos raidės

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Kiek yra kiekvienoje eilutėje žodžių, kurie baigiasi ir prasideda ta pačia raide. Kuris iš šių žodžių turi daugiausiai skirtingų raidžių? Perkelkite šį žodį kartu su už jo esančiais skyrikliais į eilutės pradžią.

### U5-12. Ilgiausio žodžio simbolai

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. I kiekvienos eilutės pabaigą perkelkite žodžius, turinčius bent vieną ilgiausio eilutės žodžio simbolį, išskyrus jį patį, kartu su už jo esančiais skyrikliais.

### U5-13. Dažniausiai pasikartojantys simbolai

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Kiekvienos eilutės kiekvienam žodyje raskite dažniausiai (>1) pasikartojančius simbolus. Trumpiausiai eilutės žodži, turinti vieną iš tokius simbolius, kartu su už jo esančiais skyrikliais pašalinkite iš eilutės.

### U5-14. Eilučių išlyginimas

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Išlyginti teksto dešinijį kraštą pagal ilgiausią eilutę. Išterpiamus tarpus tarp žodžių paskirstyti tolygiai. Tekstą pertvarkyti taip, kad žodis, einantis po taško, prasidetų didžiaja raide. Didžiajā raide privalo prasideti ir pirmasis teksto žodis.

### U5-15. Balsės

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. I kiekvienos eilutės pradžią perkelkite po vieną žodį, turintį bent vieną ilgiausio eilutės žodžio balsę, jei toks yra, išskyrus jį patį, kartu su už jo esančiais skyrikliais.

### U5-16. Skirtingi simboliai

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Pašalinkite iš teksto žodžius, sudarytus tik iš skirtingų simbolių.

### U5-17. Trumpiausias žodis

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. I kiekvienos eilutės pradžią perkelkite po vieną žodį, turintį bent vieną trumpiausio eilutės žodžio, kuris yra netrumpesnis už nurodytą ilgi, balsę, jei toks yra, išskyrus jį patį, kartu su už jo esančiais skyrikliais.

### U5-18. Teksto formato keitimas

Tekstiniame faile pateiktas teksts. Duotas teksts, prasideda žodžiu „Begin“ ir baigiasi žodžiu „End“. Jie užima atskiras eilutes. Likusį tekštą sudaro tokio formato eilutės: žodis1 = žodis2 (žodis3, žodis4,..., žodisn). Tokio formato teksts gali užimti keletą eilučių. Pakeisti šį teksto formatą į tokį: žodis2 (žodis1, žodis3, žodis4,..., žodisn). Atspausdinti eilučių numerius, kur tokio formato sakiny užima keletą eilučių.

### U5-19. Tituliniai žodžiai

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. I atskirą failą surašyti žodžius, kurie prasideda didžiaja raide, o iš senojo failo juos pašalinti. Jei netelpa į eilutę (eilutės ilgis - 81 simbolis), formuoti naują eilutę. Skyrikliai tarp žodžių naujame faile yra kablelis ir taškas.

### U5-20. Skaičių sumos

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Ar yra žodžių, kuriuos sudaro tik skaitmenys? Jei yra, raskite šias skaitmenų sekas atitinkančių skaičių sumą ir pašalinkite iš teksto pirmuosius n (n įvedamas) žodžius, sudarytus tik iš skaitmenų.

### U5-21. Didžiosios raidės

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Ilgiausią eilutės žodį iš tų žodžių, kurie prasideda didžiaja raide, perkelkite į eilutės pradžią kartu su už jo esančiais skyrikliais.

### U5-22. Žodžių poros

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Kiek yra žodžių porų, kuriose vieno žodžio paskutinė raidė sutampa su tolimesnio žodžio pirmaja raidė? Kiekvienoje eilutėje raskite ilgiausią (daugiausia simbolų) iš šių porų ir perkelkite jos pirmajį žodį kartu su jo už esančiais skyrikliais į eilutės pabaiga.

### U5-23. „Didesnės“ raidės

Tekstiniame faile pateiktas teksts. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi.. Kokios žodžių daugiausia: tu, kurių pirmoji raidė „didesnė“ už paskutinią ar tu, kurių pirmoji raidė „mažesnė“ už paskutinią? Pašalinkti žodžius, kurių pirmosios dvi raidės sutampa su dviem paskutinėmis raidėmis.

**U5-24. Baziniai žodžiai**

Tekstiniame faile pateiktas tekstas. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Kitame faile duotas bazinių žodžių sąrašas, kur žodžiai išvardinti stulpeliu. Kiekvienoje eilutėje rasti žodžių turintį daugiausia skirtingu simbolių ir pašalinti jį iš eilutės kartu su už jo esančiais skyrikliais. Bazinių žodžių nenagrinėti.

**6. Susieti rinkiniai**

Susipažinsite su:

- dvimatės konteinerinės klasės aprašymu;
- matricos elementų reikšmių įvedimu, spausdinimu, sumos, kiekio ir vidurkio skaičiavimu;
- matricos eilučių/stulpelių elementų reikšmių sumos, kiekio, didžiausios reikšmės radimui;
- susietais masyvais.

**6.1. Veiksmai su sveikujų skaičių dvimačiu masyvu**

- Konteinerinės klasės aprašymas.
- Duomenų skaitymas ir spausdinimas.
- Veiksmai su visomis konteinerio reikšmėmis.

**Užduotis.** Prekybos bazė.

Tekstiniame faile yra surašyti prekybos bazės kasose per tam tikrą laikotarpį aptarnautų pirkėjų skaičiai. Pirmoje failo eilutėje yra užrašyti du skaičiai: n - kasų skaičius ir m - dienų skaičius. Tolesnėse n eilutėse užrašyta po m skaičių - atitinkamai kasose aptarnautų pirkėjų skaičiai. Parašykite programą, kuri įvestų duomenis iš failo, juos išspausdintų faile ir suskaičiuotų, kiek iš viso buvo aptarnauta pirkėjų.

**Pradiniai duomenys ir rezultatai.**

Pradiniai duomenys	Rezultatai
6 7	5 9 8 7 3 5 6
5 9 8 7 3 5 6	6 9 8 2 1 5 2
6 9 8 2 1 5 2	8 9 0 8 8 8 8
8 9 0 8 8 8 8	4 5 6 2 3 4 3
4 5 6 2 3 4 3	5 6 9 1 0 5 8
5 6 9 1 0 5 8	3 4 5 4 6 5 7
3 4 5 4 6 5 7	Viso aptarnauta: 220

**Programos kūrimo eiga.**

- Paruošiamas pradinių duomenų failas.
- Sukuriama klasė dvimačio sveikujų skaičių masyvo duomenims saugoti.
- Pagrindinėje funkcijoje `main()` skelbiamas objektas, skirtas prekybos bazės kasose aptarnautų pirkėjų skaičiams saugoti.
- Sukuriama funkcija duomenims iš failo skaityti.
- Sukuriama funkcija pradiniam duomenims failے spausdinti.
- Sukuriama funkcija prekybos bazėje aptarnautų pirkėjų skaičiui rasti.

**OPirmas žingsnis.**

- Sukurkite klasę, skirtą dvimačiam sveikujų skaičių masyvui – prekybos bazės kasose aptarnautų pirkėjų skaičiams saugoti:

```
//-----
class Matrica
{
public:
    static const int CMaxEil = 10; // didžiausias galimas eilučių skaičius
    static const int CMaxSt = 100; // didžiausias galimas stulpelių skaičius
private:
    int A[CMaxEil][CMaxSt]; // duomenys
    int n; // eilučių skaičius
    int m; // stulpelių skaičius
public:
    Matrica(int nn = 0, int mm = 0) : n(nn), m(mm) { }
    ~Matrica() { }
    int ImtiN() { return n; }
    int ImtiM() { return m; }
    void DetiN(int nn) { n = nn; }
```

```

void DėtiM(int mm) { m = mm; }
void DėtiReikšmę(int i, int j, int r) { A[i][j] = r; }
int ImtiReikšmę(int i, int j) { return A[i][j]; }
};

//-----


- Parašykite main() funkciją, kurioje būtų paskelbtas prekybos bazėje aptarnautų pirkėjų skaičių objektas.


//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Matrica prekybosBazė;
    cout << "Programa baigė darbą" << endl;
    return 0;
}
//-----

```

- Patikrinkite, kaip programa dirba. Ekrane turite matyti:

Programa baigė darbą

### ¶ Antras žingsnis.

- Parašykite funkciją, kuri užpildytų objektą – konteinerį duomenimis iš failo:

```

//-----
// Iveda duomenis iš failo į konteinerį
// fv - failo vardas
// A - konteinerio vardas
void Įvesti(const char fv[], Matrica & A)
{
    int n, m;
    int r;
    ifstream fd(fv);
    fd >> n >> m;
    A.DėtiN(n);
    A.DėtiM(m);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++) {
            fd >> r;
            A.DėtiReikšmę(i, j, r);
        }
    fd.close();
}
//-----

```

- Parašykite funkciją, kuri išspausdintų objekto duomenis failė:

```

// Išveda konteinerio duomenis į failą
// fv - failo vardas
// A - konteinerio vardas
void Spausdinti(const char fv[], Matrica & A)
{
    ofstream fr(fv, ios::app);
    for (int i = 0; i < A.ImtiN(); i++) {
        for (int j = 0; j < A.ImtiM(); j++)
            fr << setw(4) << A.ImtiReikšmę(i, j);
        fr << endl;
    }
    fr << endl;
    fr.close();
}
//-----

```

- Papildykite programą duomenų įvedimo iš failo Duomenys.txt ir spausdinimo failė Rezultatai.txt veiksmais – užrašykite kreipinius į sukurtas funkcijas:

```
//-----
const char CDFv[] = "Duomenys.txt";

```

```

const char CRfv[] = "Rezultatai.txt";
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Matrica prekybosBazė;
    Įvesti(CDFv, prekybosBazė);
    ofstream fr(CRfv); fr.close(); // rezultatų failo išvalymas
    Spausdinti(CRfv, prekybosBazė);
    cout << "Programa baigė darbą" << endl;
    return 0;
}
//-----

```

- Išbandykite, kaip veikia programa. Rezultatų failė turite matyti:

5	9	8	7	3	5	6
6	9	8	2	1	5	2
8	9	0	8	8	8	8
4	5	6	2	3	4	3
5	6	9	1	0	5	8
3	4	5	4	6	5	7

### ¶ Trečias žingsnis.

- Parašykite funkciją, kuri suskaičiuotų, kiek iš viso pirkėjų aptarnavo prekybos bazė:

```

//-----
// Suskaičiuoja ir gražina prekybos bazėje aptarnautu pirkėjų skaičių
// A - konteinerio vardas
int VisoAptarnauta(Matrica & A)
{

```

```

    int suma = 0;
    for (int i = 0; i < A.ImtiN(); i++)
        for (int j = 0; j < A.ImtiM(); j++)
            suma = suma + A.ImtiReikšmę(i, j);
    return suma;
}
//-----

```

- Papildykite programą: atverkite rezultatų failą papildymui ir užrašykite kreipinį į auksčiau sukurtą funkciją:

```

fr.open(CRfv, ios::app);
fr << "Viso aptarnauta: " << VisoAptarnauta(prekybosBazė) << endl;
fr.close();

```

- Išbandykite programą. Rezultatų failė, be pradinių duomenų, matysite, kad iš viso buvo aptarnauta 220 pirkėjų.

### Programos patikrinimas.

Pakeiskite duomenų failo duomenis arba sukurkite kitus duomenų failus. Patikrinkite, kaip dirba programa, kai prekybos bazėje:

- viena kasa ( $n = 1$ );
- $n$  kasų, o aptarnavimo laikotarpis viena diena ( $m = 1$ );
- viena kasa ( $n = 1$ ), o aptarnavimo laikotarpis viena diena ( $m = 1$ ).

### Programos papildymas.

Papildykite spausdinimo funkciją veiksmais, kurie padarytų skaičiavimo rezultatus vaizdesniais: juos įrėmintų, o taip pat užrašytų eilučių (kasų) numerius ir stulpelių (dienu) numerius.

### Savarankiško darbo užduotis.

Parašykite ir išbandykite funkciją, kuri suskaičiuotų:

- kiek vidutiniškai pirkėjų aptarnavo viena kasa per vieną dieną;
- kiek dienų kuri iš kasų nedirbo (duomenyse yra skaičius 0).

## 6.2. Veiksmai su sveikujų skaičių dvimačio masyvo eilutėmis ir stulpeliais

- Veiksmai su konteinerio eilutėmis.
- Veiksmai su konteinerio stulpeliais.
- Didžiausios reikšmės paieška.

### Užduotis.

Papildykite ankstesnio pratimo programą veiksmais, kurie apskaičiuotų:

- kiek pirkėjų aptarnavo kiekviena kasa;
- kiek pirkėjų buvo aptarnauta kiekvieną dieną;
- kuri kasa, kurią dieną, aptarnavo daugiausia pirkėjų.

### OPirmas žingsnis.

- Parašykite funkciją, kuri suskaičiuotų ir išspausdintų, kiek pirkėjų aptarnavo kiekviena kasa:

```
//-----
// Suskaičiuoja ir išspausdina, kiek pirkėjų aptarnavo kiekviena kasa
// fr - išvedimo srauto vardas
// A - konteinerio vardas
void KiekvienaKasaAptarnavo(ofstream & fr, Matrica & A)
{
    for (int i = 0; i < A.ImtiN(); i++) {
        int suma = 0;
        for (int j = 0; j < A.ImtiM(); j++)
            suma = suma + A.ImtiReikšmę(i, j);
        fr << "Kasa nr." << setw(2) << i+1 << " aptarnavo: " << suma << endl;
    }
    fr << endl;
}
//-----
```

- Papildykite programą skaičiavimais – kreipiniu į funkciją (prieš rezultatų failo uždarymą `fr.close()`):

`KiekvienaKasaAptarnavo(fr, prekybosBazė);`

- Išbandykite programą. Rezultatų failo, be ankstesnių rezultatų, matysite:

Kasa nr. 1 aptarnavo: 43  
 Kasa nr. 2 aptarnavo: 33  
 Kasa nr. 3 aptarnavo: 49  
 Kasa nr. 4 aptarnavo: 27  
 Kasa nr. 5 aptarnavo: 34  
 Kasa nr. 6 aptarnavo: 34

### OAntras žingsnis.

- Parašykite funkciją, kuri suskaičiuotų ir išspausdintų, kiek pirkėjų buvo aptarnauta kiekvieną dieną:

```
//-----
// Suskaičiuoja ir išspausdina, kiek pirkėjų buvo aptarnauta kiekvieną dieną
// fr - išvedimo srauto vardas
// A - konteinerio vardas
void KiekvienąDienąAptarnauta(ofstream & fr, Matrica & A)
{
    for (int j = 0; j < A.ImtiM(); j++) {
        int suma = 0;
        for (int i = 0; i < A.ImtiN(); i++)
            suma = suma + A.ImtiReikšmę(i, j);
        fr << "Dieną nr." << setw(2) << j+1 << " aptarnauta: " << suma << endl;
    }
    fr << endl;
}
//-----
```

- Papildykite programą kreipiniu į funkciją (prieš rezultatų failo uždarymą `fr.close()`):

`KiekvienąDienąAptarnauta(fr, prekybosBazė);`

- Išbandykite programą. Rezultatų failo, be ankstesnių rezultatų, matysite:

Diena nr. 1 aptarnauta: 31  
 Diena nr. 2 aptarnauta: 42  
 Diena nr. 3 aptarnauta: 36  
 Diena nr. 4 aptarnauta: 24  
 Diena nr. 5 aptarnauta: 21  
 Diena nr. 6 aptarnauta: 32  
 Diena nr. 7 aptarnauta: 34

### OTrečias žingsnis.

- Parašykite funkciją, kuri surastų, kuri kasa aptarnavo daugiausiai pirkėjų:

```
//-----
// Suranda ir gražina, kuri kasa patarnavo daugiausiai pirkėjų
// A - konteinerio vardas
int KasosNumerisMaxPirkėjų(Matrica & A)
{
    int max = 0;
    int nr = 0;
    for (int i = 0; i < A.ImtiN(); i++) {
        int suma = 0;
        for (int j = 0; j < A.ImtiM(); j++)
            suma = suma + A.ImtiReikšmę(i, j);
        if (suma > max) {
            max = suma;
            nr = i+1;
        }
    }
    return nr;
}
//-----
```

- Papildykite programą kreipiniu į funkciją (prieš rezultatų failo uždarymą `fr.close()`):

`fr << "Daugiausia pirkėjų aptarnavo (kasa): "`  
`<< KasosNumerisMaxPirkėjų(prekybosBazė) << endl;`

- Išbandykite programą. Rezultatų failo, be ankstesnių rezultatų, matysite, kad daugiausiai pirkėjų aptarnavo trečia (3) kasa. Tuo galima iš tikinti ir iš prieš tai gautų rezultatų.

### Programos patikrinimas.

Pakeiskite duomenų failo duomenis arba sukurate kitus duomenų failus. Patikrinkite, kaip dirba programa, kai prekybos bazėje:

- vienu kasa ( $n = 1$ );
- $n$  kasų, o aptarnavimo laikotarpis viena diena ( $m = 1$ );
- vienu kasa ( $n = 1$ ), o aptarnavimo laikotarpis viena diena ( $m = 1$ ).

### Programos papildymas.

Pakeiskite funkciją `KasosNumerisMaxPirkėjų()` taip, kad ji surastų ir aptarnautų pirkėjų skaičių. Atitinkamai pakeiskite ir kreipinį funkciją.

### Savarankiško darbo užduotis.

Parašykite ir išbandykite funkciją, kuri suskaičiuotų:

- kurią dieną buvo aptarnauta mažiausiai pirkėjų ir keli pirkėjai buvo aptarnauti tą dieną;
- kiek pirkėjų vidutiniškai aptarnavo kiekviena kasa.

## 6.3. Dvimatis objektų masyvas

- Objektai konteineryje.
- Veiksmai su objektais visame konteineryje.
- Objekto pasiekiamumas.

**Užduotis.** Šeimos išlaidos.

Tekstiniame faile yra surašyta šeimos nario tam tikro laikotarpio (pvz.: pusės metų) išlaidos pirkiniams kiekvieną dieną savaitėmis (7 dienos). Pirmoje failo eilutėje yra užrašytas savaičių skaičius n ir skaičius 7. Tolesnėse n eilutėse yra užrašyta po 7 poras: šeimos nario pavadinimas (vyras arba žmona) ir išlaidos (jei išlaidų tą dieną nebuvo: simboliai "----" ir skaičius 0.0).

Parašykite programą, kuri įvestų duomenis iš failo, išspausdintų faile ir apskaičiuotų, kiek iš viso šeima turėjo išlaidų.

**Pradiniai duomenys ir rezultatai.**

Pradiniai duomenys						
3 7						
vyras 10.40 vyra 15.20 žmona 50.50 žmona 100.20 ---- 0.0 žmona 10.20 ---- 0.0						
žmona 15.30 ---- 0.0 ---- 0.0 žmona 20.50 vyra 55.50 vyra 10.10 žmona 30.30						
vyra 10.10 vyra 20.20 vyra 30.30 vyra 50.50 vyra 20.10 vyra 30.10 vyra 30.10						
Rezultatai						
1-dienis	2-dienis	3-dienis	4-dienis	5-dienis	6-dienis	7-dienis
vyra 10.40	vyra 15.20	žmona 50.50	žmona 100.20	---- 0.0	žmona 10.20	---- 0.0
žmona 15.30	---- 0.0	---- 0.0	žmona 20.50	vyra 55.50	vyra 10.10	žmona 30.30
vyra 10.10	vyra 20.20	vyra 30.30	vyra 50.50	vyra 20.10	vyra 30.10	vyra 30.10
Viso išleista: 509.60						

**Programos kūrimo eiga.**

- Paruošiamas pradinį duomenų failas.
- Sukuriama klasė Asmuo, skirta simbolių eilutei (string) ir realiam skaičiui (double) saugoti.
- Sukuriama klasė Matrica klasės Asmuo objektams saugoti.
- Pagrindinėje funkcijoje main() skelbiamas objektas, skirtas šeimos išlaidoms saugoti.
- Sukuriama funkcija duomenims iš failo skaityti.
- Sukuriama funkcija pradiniams duomenims rezultatų faile spausdinti.
- Sukuriama funkcija visoms šeimos išlaidoms skaičiuoti.

**1 Pirmas žingsnis.**

- Sukurkite klasę, skirtą simbolių eilutei (string) ir realiam skaičiui (double) – šeimos asmens duomenims saugoti:

```
//-----
class Asmuo
{
private:
    string vardas;
    double pinigai;
public:
    Asmuo(string vrd = "", double png = 0.0) :
        vardas(vrd), pinigai(png) { }
    ~Asmuo() { }
    void Dėti(string vrd, double png) { vardas = vrd; pinigai = png; }
    string ImtiVarda() { return vardas; }
    double ImtiPinigus() { return pinigai; }
};
```

- Sukurkite klasę (galite klasę kopijuoti iš ankstesnio darbo ir po to ją nežymiai modifikuoti), skirtą dvimačiam klasės Asmuo objektų masyvui – šeimos išlaidoms saugoti:

```
//-----
class Matrica
{
public:
    static const int CMaxEil = 100; // didžiausias galimas eilučių skaičius
    static const int CMaxSt = 7; // didžiausias galimas stulpelių skaičius
private:
    Asmuo A[CMaxEil][CMaxSt]; // duomenys: objektai
    int n; // eilučių skaičius (savaičių skaičius)
    int m; // stulpelių skaičius (7 dienos)
public:
```

```
Matrica(int nn = 0, int mm = 0) : n(nn), m(mm) { }
~Matrica()
{
    int ImtiN() { return n; }
    int ImtiM() { return m; }
    void DėtiN(int nn) { n = nn; }
    void DėtiM(int mm) { m = mm; }
    void DėtiReikšmę(int i, int j, Asmuo r) { A[i][j] = r; }
    Asmuo ImtiReikšmę(int i, int j) { return A[i][j]; }
};
```

- Parašykite main() funkciją, kurioje būtų šeimos išlaidas aprašantis objektas.

```
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Matrica šeimosIšlaidos;
    cout << "Programa baigė darbą" << endl;
    return 0;
}
```

- Patikrinkite, kaip programa dirba. Ekrane turite matyti:

Programa baigė darbą

**2 Antras žingsnis.**

- Parašykite funkciją, kuri užpildytų objekto duomenimis iš failo:

```
//-----
// Iveda duomenis iš failo į konteinerį
// fv - failo vardas
// A - konteinerio vardu
void Įvesti(const char fv[], Matrica & A)
{
    int n, m;
    string vardas;
    double pinigai;
    Asmuo asmuo;
    ifstream fd(fv);
    fd >> n >> m;
    A.DėtiN(n);
    A.DėtiM(m);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++) {
            fd >> vardas >> pinigai;
            asmuo.Dėti(vardas, pinigai);
            A.DėtiReikšmę(i, j, asmuo);
        }
    fd.close();
}
```

- Parašykite funkciją, kuri išspausdintų objekto duomenis faile:

```
//-----
// Išveda konteinerio duomenis į failą
// fv - failo vardas
// A - konteinerio vardu
void Spausdinti(const char fv[], Matrica & A)
{
    Asmuo x;
    ofstream fr(fv, ios::app);
    for (int j = 1; j <= A.ImtiM(); j++)
        fr << j << "-dienis" << " ";
    fr << endl;
    for (int i = 0; i < A.ImtiN(); i++) {
        for (int j = 0; j < A.ImtiM(); j++) {
            x = A.ImtiReikšmę(i, j);
            fr << x.vardas << " ";
            fr << x.pinigai << endl;
        }
    }
}
```

```

    fr << setw(5) << x.ImtiVarda() << " " << right
        << fixed << setw(6) << setprecision(2) << x.ImtiPinigus() << " ";
    }
    fr << endl;
}
fr << endl;
fr.close();
}
//-----

```

- Papildykite programą duomenų įvedimo iš failo Duomenys.txt ir spausdinimo rezultatų failė Rezultatai.txt veiksmais – užrašykite kreipinius į sukurtas funkcijas:

```

//-----
const char CDFv[] = "Duomenys.txt";
const char CRFv[] = "Rezultatai.txt";
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Matrica šeimosIšlaidos;
    Ivesti(CDFv, šeimosIšlaidos);
    Spausdinti(CRFv, šeimosIšlaidos);
    cout << "Pradiniai duomenys išspausdinti faile " << CRFv << endl;
    cout << "Programa baigė darbą" << endl;
    return 0;
}
//-----

```

- Išbandykite, kaip veikia programa. Ekrane turėtumėte matyti:

Pradiniai duomenys išspausdinti faile Rezultatai.txt  
Programa baigė darbą

- Rezultatų failė Rezultatai.txt bus išspausdintos šeimos išlaidos:

1-dienis	2-dienis	3-dienis	4-dienis	5-dienis	6-dienis	7-dienis
vyras 10.40	vyras 15.20	žmona 50.50	žmona 100.20	----- 0.00	žmona 10.20	----- 0.00
žmona 15.30	----- 0.00	----- 0.00	žmona 20.50	vyras 55.50	vyras 10.10	žmona 30.30
vyras 10.10	vyras 20.20	vyras 30.30	vyras 50.50	vyras 20.10	vyras 30.10	vyras 30.10

### ¶ Trečias žingsnis.

- Parašykite funkciją, kuri suskaičiuotų, kiek iš viso šeima turėjo išlaidų:

```

//-----
// Suskaičiuoja ir gražina, kiek iš viso šeima turėjo išliadų
// A - konteinerio vardas
double VisosIšlaidos(Matrica & A)
{
    double suma = 0;
    for (int i = 0; i < A.ImtiN(); i++)
        for (int j = 0; j < A.ImtiM(); j++) {
            Asmuo x = A.ImtiReikšmę(i, j);
            suma = suma + x.ImtiPinigus();
        }
    return suma;
}
//-----

```

- Papildykite programą: atverkite rezultatų failą papildymui ir užrašykite kreipinį į aukščiau sukurtą funkciją:

```

fr.open(CRFv, ios::app);
fr << "Viso išleista: " << fixed << setw(7) << setprecision(2)
    << VisosIšlaidos(šeimosIšlaidos) << endl;
fr.close();

```

- Išbandykite programą.

### Programos patikrinimas.

Pakeiskite duomenų failo duomenis arba sukurkite kitą duomenų failą. Patikrinkite, kaip dirba programa, kai šeimos išlaidos skaičiuojamos vieną savaitę (n = 1).

### Programos papildymas.

Papildykite spausdinimo funkciją veiksmais, kurie padarytų skaičiavimo rezultatus vaizdesniais: juos ižėmiant, o taip pat užrašytų eilučių (savaičių) numerius.

### Užduotis savarankiškam darbui.

Parašykite ir išbandykite funkciją, kuri suskaičiuotų:

- kelias dienas šeima neturėjo išlaidų (duomenyse skaičius lygus 0.0);
- kiek išlaidų turėjo vienas šeimos narys (bus reikalingi du kreipiniai į šią funkciją: žmonos ir vyro išlaidoms skaičiuoti).

### 6.4. Veiksmai su objektais dvimacho masyvo eilutėse ir stulpeliuose

- Veiksmai su objektais konteinerio eilutėje.
- Veiksmai su objektais konteinerio stulpelyje.
- Veiksmai su konteinerio objektu.

### Užduotis.

Papildykite ankstesnio pratimo programą, kuri papildomai apskaičiuotų:

- kiek išlaidų šeima turėjo kiekvieną savaitę;
- kiek išlaidų šeima turėjo nurodytomis savaitės dienomis, pavyzdžiui antradieniais, šeštadieniais ir t. t.;
- kuris šeimos narys per vieną dieną išleido didžiausią pinigų sumą ir kiek.

### ¶ Pirmas žingsnis.

- Parašykite funkciją, kuri suskaičiuotų ir išspausdintų, kiek išlaidų šeima turėjo kiekvieną savaitę:

```

//-----
// Suskaičiuoja ir išspausdina, kiek šeima turėjo išlaidų kiekviena savaitę
// fr - išvedimo srauto vardas
// A - konteinerio vardas
void IšlaidosSavaitėmis(ofstream & fr, Matrica & A)
{
    for (int i = 0; i < A.ImtiN(); i++) {
        double suma = 0;
        for (int j = 0; j < A.ImtiM(); j++) {
            Asmuo x = A.ImtiReikšmę(i, j);
            suma = suma + x.ImtiPinigus();
        }
        fr << "Savaitė nr." << setw(2) << i+1 << ", išlaidos: "
            << fixed << setw(7) << setprecision(2) << suma << endl;
    }
    fr << endl;
}
//-----

```

- Papildykite programą kreipiniu į funkciją (prieš rezultatų failo uždarymą fr.close()):

IšlaidosSavaitėmis(fr, šeimosIšlaidos);

- Išbandykite programą. Rezultatų failė, be ankstesnių rezultatų, matysite:

Savaitė nr. 1, išlaidos: 186.50

Savaitė nr. 2, išlaidos: 131.70

Savaitė nr. 3, išlaidos: 191.40

### ¶ Antras žingsnis.

- Parašykite funkciją (prototipas ir tekstas), kuri suskaičiuotų, kiek išlaidų šeima turėjo nurodytą savaitės dieną:

```

//-----
// Suskaičiuoja ir gražina, kiek išlaidų šeima turėjo nurodyta savaitė diena
// A - konteinerio vardas
// nr - savaitės dienos numeris
double IšlaidosSavaitėsDienąX(Matrica & A, int nr)
{

```

```

double suma = 0;
for (int i = 0; i < A.ImtiN(); i++) {
    Asmuo x = A.ImtiReikšmė(i, nr-1);
    suma = suma + x.ImtiPinigus();
}
return suma;
}

```

- Papildykite programą kreipiniais į funkciją (prieš rezultatų failo uždarymą `fr.close()`). Patikrinkite, kiek pinigų šeima išleido antradieniais ir šeštadieniais:

```

fr << "Išlaidos 2-dieniais:" << IšlaidosSavaitėsDienąX(šeimosIšlaidos, 2)
     << endl;
fr << "Išlaidos 6-dieniais:" << IšlaidosSavaitėsDienąX(šeimosIšlaidos, 6)
     << endl;

```

- Išbandykite programą. Rezultatų faile, be ankstesnių rezultatų, matysite:

Išlaidos 2-dieniais: 35.40  
Išlaidos 6-dieniais: 50.40

#### Trečias žingsnis.

- Parašykite funkciją, kuri suskaičiuotų, kada (kurią savaitę ir kurią savaitės dieną) buvo išleista didžiausia pinigų suma:

```

// Suskaičiuoja, kuria savaitę ir kuria savaitės diena
// buvo išleista didžiausia pinigų suma
// A - konteinerio vardas
// eilNr - savaitės numeris
// stNr - savaitės dienos numeris
void DienaMaxIšlaidos(Matrica & A, int & eilNr, int & stNr)
{
    eilNr = -1; stNr = -1;
    double max = 0;
    for (int i = 0; i < A.ImtiN(); i++)
        for (int j = 0; j < A.ImtiM(); j++) {
            double x = A.ImtiReikšmė(i, j).ImtiPinigus();
            if (x > max) {
                max = x;
                eilNr = i + 1;
                stNr = j + 1;
            }
        }
}

```

- Papildykite programą kintamaisiais ir skaičiavimais, kada ir kas išleido daugiausia (prieš rezultatų failo uždarymą `fr.close()`):

```

int savaitė, diena;
Asmuo a;
DienaMaxIšlaidos(šeimosIšlaidos, savaitė, diena);
fr << "Daugiausia išleista: " << savaitė << " sav. " << diena << " d., ";
a = šeimosIšlaidos.ImtiReikšmė(savaitė-1, diena-1);
fr << " pinigus išleido " << a.ImtiVarda() << " "
     << fixed << setw(7) << setprecision(2) << a.ImtiPinigus() << endl;

```

- Išbandykite programą. Rezultatų faile, be ankstesnių rezultatų, matysite:

Daugiausiai išleista 1 sav. 4 d., pinigus išleido žmona 100.20

#### Programos patikrinimas.

Pakeiskite duomenų failo duomenis arba sukurkite kitą duomenų failą. Patikrinkite, kaip dirba programa, kai šeimos išlaidos skaičiuojamos vieną savaitę ( $n = 1$ ) ir tą savaitę:

- visai nebuvu išlaidų;
- buvu tik žmonos išlaidos;
- buvu tik vyro išlaidos.

#### Programos papildymas.

Pakeiskite funkciją `IšlaidosSavaitėmis()` taip, kad šios funkcijos rezultatai būtų spausdinami rezultatų faile lentele, sudaryta iš dviejų skilčių: savaitės numeris ir išlaidos.

#### Savarankiško darbo užduotis.

Parašykite ir išbandykite funkciją:

- kuri suskaičiuotų, kurią savaitę išlaidos buvo mažiausios;
- kuri suskaičiuotų ir išspausdintų rezultatų faile lentele šeimos išlaidas visomis savaitės dienomis: pirmadieniais, antradieniais ir t. t. Pasinaudokite funkcija `IšlaidosSavaitėsDienąX()`.

#### 6.5. Objektų masyvas ir dvimatis sveikujų skaičių masyvas

- Klasės su dvimi susietais masyvais aprašymas.
- Duomenų skaitymas ir spausdinimas.
- Veiksmai susietuose masyvuose.

#### Užduotis.

Mokinį laikas, praleistas interne.

Tekstiniams faile yra mokyklos 5-12 klasių mokinų sąrašas. Pirmoje failo eilutėje užrašytas mokinį skaičius n. Tolesnėse failo eilutėse užrašyta informacija apie mokinius: pavardė, vardas, klasė, pažangumas (mokymosi vidurkis). Po to užrašytas skaičius n ir dienų skaičius m. Žemiau pateikta informacija apie mokinį kiekvieną dieną praleistą laiką minutėmis interne: mokiniai (eilutės), dienos (stulpeliai).

Parašykite programą, kuri įvestų duomenis iš failo, išspausdintų faile, surikiuotų mokinius pagal klasses ir praleistą laiką interne, suskaičiuotų, kiek vidutiniškai nurodytos klasės mokiniai praleido interne.

#### Pradiniai duomenys ir rezultatai.

Pradiniai duomenys					
7	Jonaitis	Jonas	6	7.5	
	Aleksaitė	Alina	6	9.5	
	Petraitis	Petras	5	8.5	
	Antanaitis	Antanas	6	5.5	
	Juozaitytė	Juozas	5	8.5	
	Rimaitis	Rimas	6	7.5	
	Rasaitė	Rasa	5	6.0	
7 5					
120	100	90	60	50	
100	200	150	200	10	
80	90	80	90	80	
120	80	60	140	70	
60	60	60	60	60	
0	0	0	0	0	
0	50	60	120	40	

Rezultatai					
Mokyklos mokiniai (laikai = 0)					
Nr.	Pavardė	Vardas	Klasė	Pažangumas	Laikas
1.	Jonaitis	Jonas	6	7.50	0
2.	Aleksaitė	Alina	6	9.50	0
3.	Petraitis	Petras	5	8.50	0
4.	Antanaitis	Antanas	6	5.50	0
5.	Juozaitytė	Juozas	5	8.50	0
6.	Rimaitis	Rimas	6	7.50	0
7.	Rasaitė	Rasa	5	6.00	0

Mokinii laikai, praleisti interne per 5 dienas

- 120 100 90 60 50
- 100 200 150 200 10
- 80 90 80 90 80
- 120 80 60 140 70

5.	60	60	60	60	60
6.	0	0	0	0	0
7.	0	50	60	120	40

Mokyklos mokiniai (papildyta, laikai != 0)

Nr.	Pavardė	Vardas	Klasė	Pažangumas	Laikas
-----	---------	--------	-------	------------	--------

1.	Jonaitis	Jonas	6	7.50	420
2.	Aleksaitė	Alina	6	9.50	660
3.	Petraitis	Petras	5	8.50	420
4.	Antanaitis	Antanas	6	5.50	470
5.	Juozaitytė	Juozas	5	8.50	300
6.	Rimaitis	Rimas	6	7.50	0
7.	Rasaitė	Rasa	5	6.00	270

Mokyklos mokiniai (surikiuoti)

Nr.	Pavardė	Vardas	Klasė	Pažangumas	Laikas
-----	---------	--------	-------	------------	--------

1.	Rasaitė	Rasa	5	6.00	270
2.	Juozaitytė	Juozas	5	8.50	300
3.	Petraitis	Petras	5	8.50	420
4.	Rimaitis	Rimas	6	7.50	0
5.	Jonaitis	Jonas	6	7.50	420
6.	Antanaitis	Antanas	6	5.50	470
7.	Aleksaitė	Alina	6	9.50	660

Mokinių laikai, praleisti interneite (po rikiavimo) per 5 dienas

1.	0	50	60	120	40
2.	60	60	60	60	60
3.	80	90	80	90	80
4.	0	0	0	0	0
5.	120	100	90	60	50
6.	120	80	60	140	70
7.	100	200	150	200	10

5 klasės mokiniai interneite vidutiniškai praleido 330 minučių

### Programos kūrimo eiga.

- Paruošiamas pradinių duomenų failas.
- Sukuriama klasė Mokinys, skirta vieno mokinio duomenims saugoti.
- Sukuriama klasė Mokykla klasės Mokinys objektams ir dvimačiam sveikujų skaičių masyvui (mokinių laikams praleistiems interneite) saugoti.
- Pagrindinėje funkcijoje main() skelbiamas objektas, skirtas mokyklos mokinių duomenims saugoti.
- Sukuriamos dvi funkcijos duomenims – mokiniams ir jų interneite praleistiems laikams – iš failo skaityti.
- Sukuriamos dvi funkcijos pradiniam duomenims – mokinių sąrašui ir jų interneite praleistiems laikams – rezultatų faile spausdinti.
- Sukuriama metoda klasės Mokinys masyvo objektams papildyti laikais, praleistais interneite.
- Sukuriama funkcija nurodytos klasės mokinių vidutinam praleistam laikui interneite skaičiuoti.

### ¶ Pirmas žingsnis.

- Sukurkite klasę mokinio duomenims saugoti (failas Mokinys.h):

```

-----
class Mokinys
{
private:
    string pav, vard; // pavardė, vardas
    int klas; // klasė
    double vid; // mokymosi vidurkis
    int laikas; // laikas, praleistas interneite
public:
    Mokinys(string pv = "", string vrd = "", int kls = 0, double vd = 0.0,
             int laik = 0);
    ~Mokinys() { }
    void Dėti(string pv, string vrd, int kls, double vd);
    void Dėti(int laik) { laikas = laik; }
    string ImtiPav() { return pav; }
    string ImtiVard() { return vard; }
    int ImtiKlas() { return klas; }
    double ImtiVid() { return vid; }
    int ImtiLaika() { return laikas; }
    bool operator <=(const Mokinys & kitas);
};

-----
• Parašykite klasės Mokinys metodus, kurių tekstai netilpo klasės viduje (failas Mokinys.cpp):
-----

Mokinys::Mokinys(string pv, string vrd, int kls, double vd, int laik):
    pav(pv), vard(vrd), klas(kls), vid(vd), laikas(laik)
{
}

-----
void Mokinys::Dėti(string pv, string vrd, int kls, double vd)
{
    pav = pv;
    vard = vrd;
    klas = kls;
    vid = vd;
}

-----
bool Mokinys::operator <=(const Mokinys & kitas)
{
    return klas < kitas.klas ||
           klas == kitas.klas && laikas < kitas.laikas;
}

-----
• Sukurkite klasę Mokykla, skirtą klasės Mokinys objektų masyvui ir dvimačiam sveikujų skaičių
masyvui – laikams saugoti (failas Mokykla.h):
-----

class Mokykla
{
public:
    static const int CMaxMk = 1000; // didžiausias galimas mokinių skaičius
    static const int CMaxDn = 30; // didžiausias galimas dienų skaičius
private:
    Mokinys Mokiniai[CMaxMk]; // mokinių sąrasas
    int n; // mokinių skaičius
    int WWW[CMaxMk][CMaxDn]; // laikas, praleistas interneite
    int m; // dienų skaičius
public:
    Mokykla(): n(0), m(0) { }
    ~Mokykla() { }
    void DėtiMokinį(Mokinys mok) { Mokiniai[n++] = mok; }
    Mokinys ImtiMokinį(int nr) { return Mokiniai[nr]; }
    void PakeistiMokinį(int nr, Mokinys mok) { Mokiniai[nr] = mok; }
    void DėtiWWW(int i, int j, int r) { WWW[i][j] = r; }
    int ImtiWWW(int i, int j) { return WWW[i][j]; }
}

```

```

void DėtiN(int nn) { n = nn; }
void DėtiM(int mm) { m = mm; }
int ImtiN() { return n; }
int ImtiM() { return m; }
void SukeistiEilutesWWW(int nr1, int nr2);
};

//-----


- Parašykite klasės Mokykla metodą, kurio tekstas netilpo klasės viduje (failas Mokykla.cpp):


//-----
// Sukeičia dvi eilutes vietomis dvimačiame masyve WWW(n,m)
// nr1 - pirmos eilutės numeris
// nr2 - antros eilutės numeris
void Mokykla::SukeistiEilutesWWW(int nr1, int nr2)
{
    for (int j = 0; j < m; j++) {
        int d = WWW[nr1][j];
        WWW[nr1][j] = WWW[nr2][j];
        WWW[nr2][j] = d;
    }
}

//-----


- Parašykite main() funkciją, kurioje būtų mokyklos mokinius aprašantis objektas.


//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Mokykla mokykl; // mokyklos mokinii duomenys
    cout << "Programa baigė darbą" << endl;
    return 0;
}

//-----


- Patikrinkite, kaip dirba programa. Ekrane turite matyti:


Programa baigė darbą

```

### ¶ Antras žingsnis.

- Parašykite pirmąjį funkciją, kuri užpildytų objektų masyvą Mokiniai(n) duomenimis iš failo:

```

// Iveda duomenis iš atverto srauto į objektų masyvą Mokiniai(n)
// fd - srauto vardas
// mokykl - objekto, kuriame yra masyvas, vardas
void ĮvestiMok(ifstream & fd, Mokykla & mokykl)
{
    string pav, vard;
    int klas;
    double vid;
    Mokinys mok;
    int n;
    fd >> n;
    for (int i = 0; i < n; i++) {
        fd >> pav >> vard >> klas >> vid;
        mok.Dėti(pav, vard, klas, vid);
        mokykl.DėtiMokinį(mok);
    }
}

```

- Parašykite antrąjį funkciją, kuri užpildytų dvimatį masyvą WWW(n, m) duomenimis iš failo:

```

// Iveda duomenis iš atverto srauto į dvimatį skaičių masyvą WWW(n,m)
// fd - srauto vardas
// mokykl - objekto, kuriame yra dvimatis skaičių masyvas, vardas
void ĮvestiLaik(ifstream & fd, Mokykla & mokykl)
{

```

```

int n, m;
int laikas;
fd >> n >> m;
mokykl.DėtiN(n);
mokykl.DėtiM(m);
for (int i = 0; i < n; i++) {
    for (int j = 0; j < m; j++) {
        fd >> laikas;
        mokykl.DėtiWWW(i, j, laikas);
    }
}

//-----


- Parašykite funkciją, kuri išspausdintų objektų masyvo Mokiniai(n) duomenis faile:


// Išveda objektų masyvo Mokiniai(n) duomenis į faila lentele
// fv - failo vardas
// mokykl - objekto, kuriame yra masyvas, vardas
// koment - komentarinis užrašas virš lentelės
void SpausdintiMok(const char fv[], Mokykla & mokykl, string koment)
{
    ofstream fr(fv, ios::app);
    fr << koment << endl;
    fr << "-----";
    fr << " Nr. Pavardė Vardas Klasė Pažangumas Laikas " << endl;
    fr << "-----";
    for (int i = 0; i < mokykl.ImtiN(); i++) {
        fr << setw(3) << i+1 << ".";
        fr << left << setw(15) << mokykl.ImtiMokinį(i).ImtiPav() << " "
        << setw(10) << mokykl.ImtiMokinį(i).ImtiVard() << " "
        << right << setw(3) << mokykl.ImtiMokinį(i).ImtiKlas() << " "
        << fixed << setw(7) << setprecision(2)
        << mokykl.ImtiMokinį(i).ImtiVid() << " "
        << setw(4) << mokykl.ImtiMokinį(i).ImtiLaika() << endl;
    }
    fr << "-----" << endl;
    fr << endl;
}

//-----


- Parašykite funkciją, kuri išspausdintų dvimatičio masyvo WWW(n, m) duomenis faile:


// Išveda dvimatičio skaičių masyvo WWW(n,m) duomenis į faila
// fv - failo vardas
// mokykl - objekto, kuriame yra dvimatis skaičių masyvas, vardas
// koment - komentarinis užrašas
void SpausdintiLaik(const char fv[], Mokykla & mokykl, string koment)
{
    ofstream fr(fv, ios::app);
    fr << koment + " per " << mokykl.ImtiM() << " dienas" << endl << endl;
    for (int i = 0; i < mokykl.ImtiN(); i++) {
        fr << setw(4) << i+1 << ".";
        for (int j = 0; j < mokykl.ImtiM(); j++) {
            fr << setw(4) << mokykl.ImtiWWW(i, j);
            fr << endl;
        }
        fr << endl;
        fr.close();
    }
}

//-----


- Papildykite programą duomenų įvedimo iš failo Duomenys.txt ir spausdinimo rezultatų failo Rezultatai.txt veiksmais – užrašykite kreipinius į sukurtas funkcijas:


const char CDrv[] = "Duomenys.txt";
const char CRfv[] = "Rezultatai.txt";

```

```

-----  

int main()
{
    setlocale(LC_ALL, "Lithuanian");

    Mokykla mokykl; // mokyklos mokiniai duomenys
    ofstream fr(CRfv);
    ifstream fd(CDfv);
    IvestiMok(fd, mokykl);
    SpausdintiMok(fr, mokykl, "Mokyklos mokiniai (laikai = 0)");
    IvestiLaik(fd, mokykl);
    SpausdintiLaik(fd, mokykl, "Mokiniai laikai, praleisti interne");
    fd.close();
    fr.close();

    cout << "Pradiniai duomenys išspausdinti faile " << CRfv << endl;
    cout << "Programa baigė darbą" << endl;
    return 0;
}
-----
```

- Išbandykite, kaip veikia programa. Ekrane turėtumėte matyti:

Pradiniai duomenys išspausdinti faile Rezultatai.txt  
Programa baigė darbą

- Rezultatų failas Rezultatai.txt bus išspausdintas mokiniai sąrašas ir dvimačiame masyve esantys duomenys (žiūr. rezultatus šio skyrelio pradžioje).

### ¶ Trečias žingsnis.

- Parašykite klasės Mokykla metodą (prototipas ir tekstas), masyvo Mokiniai(n) objektams papildyti laikais, praleistais interne iš masyvo WWW(n,m):

```

-----  

// Objektų masyvo Mokiniai(n) papildymas laikais praleistais interne  

// iš dvimačio masyvo WWW(n,m)
void Mokykla::PapildytMokinijDuomenis()
{
    int suma;
    Mokinys mok;
    for (int i = 0; i < n; i++) {
        suma = 0;
        for (int j = 0; j < m; j++) {
            suma = suma + WWW[i][j];
        }
        mok = ImtiMokinij(i);
        mok.Deti(suma);
        PakertiMokinij(i, mok);
    }
}
-----
```

- Išbandykite aukščiau parašytą metodą – parašykite main() funkcijoje kreipinį į ši metodą ir kreipinį į pirmają spausdinimo funkciją (prieš rezultatų srauto uždarymą):

```

mokykl.PapildytMokinijDuomenis();
SpausdintiMok(fr, mokykl, "Mokyklos mokiniai (papildyta, laikai != 0)");


- Rezultatų failas Rezultatai.txt bus papildytas dar viena lentele (žiūr. rezultatus šio skyrelio pradžioje).

```

### ¶ Ketvirtas žingsnis.

- Parašykite klasės Mokykla metodą (prototipas ir tekstas) masyvui Mokiniai(n) ir dvimačiam masyvui WWW(n,m) rikiuoti pagal klasses ir laikus interne:

```

-----  

// Surikiuoja objektų masyva Mokiniai(n) pagal klasses ir laikus
// praleistus interne
// Pastaba: kartu atliekami pakeitimai ir dvimačiame skaičiu masyve WWW(n,m)
```

```

void Mokykla::RikiuotiMinMax()
{
    Mokinys mok;
    for (int i = 0; i < n-1; i++) {
        int minnr = i;
        for (int j = i+1; j < n; j++)
            if (ImtiMokinij(j) <= ImtiMokinij(minnr))
                minnr = j;
        mok = ImtiMokinij(i);
        // pakeitimai masyvuose Mokiniai ir WWW
        PakertiMokinij(i, ImtiMokinij(minnr));
        PakertiMokinij(minnr, mok);
        SukeistiEilutesWWW(i, minnr);
    }
}
-----
```

- Išbandykite aukščiau parašytą metodą – parašykite main() funkcijoje kreipinį į ši metodą ir kreipinį į abi spausdinimo funkcijas (prieš rezultatų srauto uždarymą):

```

mokykl.RikiuotiMinMax();
SpausdintiMok(fr, mokykl, "Mokyklos mokiniai (surikiuoti)");
SpausdintiLaik(fr, mokykl,
    "Mokiniai laikai, praleisti interne (po rikiavimo)");
-----
```

- Rezultatų failas Rezultatai.txt bus papildytas dviems lentelėmis (žiūr. rezultatus šio skyrelio pradžioje).

### ¶ Penktas žingsnis.

- Parašykite funkciją (prototipas ir tekstas) nurodytos klasės mokiniai vidutinam praleistam laikui interne skaičiuoti:

```

-----  

// Suskaičiuoja ir gražina nurodytos klasės mokiniai vidutinį laika
// praleistą interne
// mokykl – objekto vardas
// klasė – klasės numeris
int VidLaikasKl(Mokykla & mokykl, int klasė)
{
    int suma = 0,
        kiek = 0;
    for (int i = 0; i < mokykl.ImtiN(); i++)
        if (mokykl.ImtiMokinij(i).ImtiKlas() == klasė) {
            kiek++;
            suma += mokykl.ImtiMokinij(i).ImtiLaik();
        }
    if (kiek)
        return suma / kiek;
    else
        return 0;
}
-----
```

- Išbandykite aukščiau parašytą funkciją – parašykite main() funkcijoje kreipinį į šią funkciją:

```

int klasė;
cout << "Užrašykite klasę (1-12): ";
cin >> klasė;
fr << klasė << " klasės mokiniai interne vidutiniškai praleido "
    << VidLaikasKl(mokykl, klasė) << " minučių" << endl;


- Paleidus programą ir klaviatūra įvedus 5 (penkta klasė) rezultatų failas Rezultatai.txt bus papildytas viena eilute (žiūr. rezultatus šio skyrelio pradžioje).

```

### Programos patikrinimas.

Pakeiskite duomenų failo duomenis arba susikurkite kitą duomenų failą. Patikrinkite, kaip dirba programa, kai dienų skaičius m = 1.

## Programos papildymas.

Pakeiskite funkciją SpausdintiLaik() taip, kad būtų sunumeruoti ir mokinį laikai praleisti internete, t. y. virš stulpelių būtų užrašyti dienų numeriai.

### **Savarankiško darbo užduotis.**

Parašykitė ir išbandykite funkciją, kuri suskaičiuotų, kiek mokykloje yra mokiniai, kurie nesinaudoja internetu.

## 6.6. Kontroliniai klausimai

1. Kokie yra pagrindiniai vienmačio ir dvimačio konteinerio skirtumai?
  2. Kaip kompiuterio atmintyje yra išdėstomi dvimačio konteinerio duomenys?
  3. Aprašykite dvimatį sveikujų skaičių masyvą, kuriame būtų viena eilutė ir CMax stulpelių.
  4. Aprašykite dvimatį realiųjų skaičių masyvą, kuriame būtų CMax eilučių ir vienas stulpelis.
  5. Užrašykite sakinius, kurie dvimačiame sveikujų masyve A(n, m) sukeistų vietomis pirmo ir paskutinio elemento reikšmes.
  6. Duota konteinerinė klasė A, kurioje yra dvimatis sveikujų skaičių masyvas A(n, m). Užrašykite klasės A metodą Didinti(), kuris masyve A nurodytos eilutės pirmojo elemento reikšmę padidintų 1 (vienetu).
  7. Duota konteinerinė klasė A, kurioje yra dvimatis sveikujų skaičių masyvas A(n, m). Užrašykite klasės A metodą Mažinti(), kuris masyve A nurodyto stulpelio paskutiniojo elemento reikšmę sumažintų 1 (vienetu).
  8. Duota konteinerinė klasė A, kurioje yra dvimatis sveikujų skaičių masyvas A(n, m). Užrašykite klasės A metodą ŠalintiEilutę(), kuris iš masyvo A pašalintų nurodytą eilutę.
  9. Duota konteinerinė klasė A, kurioje yra dvimatis sveikujų skaičių masyvas A(n, m). Užrašykite klasės A metodą ŠalintiStulpelių(), kuris iš masyvo A pašalintų nurodytą stulpelį.
  10. Duotas funkcijos aprašymas ir prototipas:  

```
// Funkcija suskaičiuoja ir grąžina masyvo A(n) reikšmių sumą  
int Suma(int A[], int n);
```

Kuris iš pateiktų kreipinių į funkciją gražina dvimačio sveikujų skaičių masyvo A(n, m) antros eilutės reikšmių sumą?

- a) Suma (A[1], m);
  - b) Suma (A[1], m-1);
  - c) Suma (A[2], n);
  - d) Suma (A[2], n-1)

11. Duota funkcija:

```
int Funkcija(int A[], int n)
{
    int s = 0;
    for (int j = 0; j < n; j++)
        if (A[j] > A[n-1]) s += A[j];
    return s;
}
```

Ka matysite ekrane, atlikus tokius veiksmus?

```

int A[10][10] = { { 1, 3, 5, 2 },  

                  { 0, 2, -4, 2 },  

                  { 1, 4, 5, 0 } };  

int n = 3, m = 4;  

cout << Funkcija(A[1], m) << " " << Funkcija(A[2], m) << endl;

```

- a) 0 10
  - b) 8 0
  - c) 10 8
  - d) 0 8

- ### 12. Duotas programos fragmentas:

```

12. int A[10][10]; int n = 5;
...
int s = 0;
for (int i = 0; i < n; i++)
    s += A[i][i];
...

```

Kurios matricos dalies elementų reikšmių suma skaiciuojama?

- a) pagrindinės įstrižainės;
  - b) šalutinės įstrižainės;
  - c) i-osios eilutės;
  - d) virš pagrindinės įstrižainės

## 6.7. Užduotys

### U6-1. Pelnyti taška

Pirmoje failo eilutėje nurodytas krepšinio komandos pavadinimas, ištais, krepšininkų skaičius ir žaistų rungtynių skaičius. Tolesnėse eilutėse pateikta informacija apie krepšininkus: pavardė, vardas, gimimo metai, ūgis, žaidimo pozicija. Žemiau atitinkamai krepšininkų (stulpeliai) išdėstymo tvarka pateikta kiekvienose rungtynėse (eilutės) įmestų taškų skaičius. Suraskite rezultatyviausią komandos krepšininką. Suraskite kiekvienos pozicijos rezultatyviausią krepšininką. Atleiskite iš komandos du mažiausiai taškų pelniusius krepšininkus (pašalinkite jų rezultatus ir jų duomenis).

## U6–2. Futbolas

Pirmoje failo eilutėje nurodytas futbolo komandų skaičius. Tolesnėse eilutes pateikta informacija apie futbolo komandas: pavadinimas, miestas, trenerio pavardė, vardas. Žemiau pateikta I rato rezultatų lentelė, išreikšta pelnytais įvarčiais. Suskaičiuokite kiekvienos komandos surinktų taškų skaičių, jei už pergalę skiriami 3 taškai, o už lygiąsias – 1 taškas. Sudarykite komandų turnyrinę lentelę – surikiuokite surinktų taškų mažėjimo tvarka. Jei komandos surinko taškų vienodai, aukščiau ta komanda, kuri turi daugiau pergalų. Suraskite daugiausiai įvarčių pelniusią komandą. Suraskite komandas, kurios daugiausiai rungtynių nepraleido įvarčių.

### U6-3. Studentai

3. Studentai  
Pirmoje failo eilutėje nurodytas studentų kiekis ir mėnesio dienų skaičius. Tolesejantiame eilutes pateikta nurodyta informacija apie studentus: studento pavardė, vardas, fakultetas, specialybė. Žemiau pateikta informacija apie studentų paskaitų lankomumą: studentai (stulpeliai), praleista paskaitų per dieną (eilutės). Nustatykite, ar yra dienų, kai paskaitose dalyvavo visi studentai, jei taip, kurios tai dienos. Surikiuokite studentus pagal fakultetus. Suskaičiuokite, kiek kiekvieno fakulteto studentai praleido paskaitų. Nustatykite, ar yra studentų, kurios reikia šalinti (nelankė paskaitų daugiau nei pusę mėnesio dienų). Pašalinkite tokius studentus.

## U6-4. Prenumerata

Paštas atlieka leidinių prenumeratą. Pirmoje failo eilutėje nurodytas leidinių kiekis ir mensesio dienų skaičius. Tolesnėse eilutėse pateikta informacija apie leidinius: pavadinimas, mensesio prenumeratos kaina, banko pavadinimas, sąskaitos numeris, procentai, atiduodami prenumeratos rinkėjui. Žemiau pateikta informacija apie kiekvieno leidinio prenumeratos eiga: leidiniai (stulpeliai), kiek jų užsakytas (eilutės). Suskaičiuokite, kiek per mensesį kurio leidinio užsakytas. Nustatykite, kuris leidinys yra lyderis. Kiekvienam bankui sudarykite pavedimų sąrašą, kad pversti leidiniui prenumeratos pinigus. Nustatykite, ar buvo leidinių, kurie nebuvo užsakomi daugiau nei vieną dieną.

### U6-5. Darbininkai

Pirmoje failo eilutėje nurodytas darbininkų skaičius, mėnesio dirbtų dienų skaičius, vienos detalės įkainis. Tolesnėse eilutėse pateikta informacija apie darbininkus: pavardė, vardas, banko pavadinimas ir sąskaitos numeris. Žemiau pateikta, kiek kiekvieną dieną (eilutės) darbininkas (stulpeliai) pagamino detalijų. Suskaičiuokite kiekvienam darbininkui jo atlyginimą. Nustatykite, kuriam darbininkui blogiausiai sekėsi dirbti. Nustatykite, kurią mėnesio dieną buvo pagaminta daugiausiai detalių. Kiekvienam bankui atskirai sudarykite pavedimų sąrašą, kur nurodysite darbininko pavardę, vardą, sąskaitos numerį ir pervedamą sumą.

**U6–6. Žaistos minutės**

Pirmoje failo eilutėje nurodytas komandos pavadinimas ir miestas, krepšininkų skaičius ir žaistų rungtynių skaičius. Tolesnėse eilutėse pateikta informacija apie krepšininkus: pavardė, vardas, gimimo metai, ūgis, žaidimo pozicija. Žemiau atitinkamai krepšininkų (eilutės) išdėstymo tvarka pateikta kiekvienose rungtynėse (stulpeliai) žaistų minučių skaičius. Suraskite daugiausiai laiko žaidusį krepšininką. Atleiskite komandos krepšininką. Suraskite kiekvienos pozicijos daugiausiai laiko žaidusį krepšininką. Atleiskite iš komandos du mažiausiai laiko žaidusius krepšininkus (pašalinkite jų rezultatus ir jų duomenis).

**U6–7. Aukščiausia lyga**

Pirmoje failo eilutėje pateiktas aukščiausios futbolo lygos komandų skaičius. Tolesnėse eilutėse pateikta informacija apie futbolo komandas: pavadinimas, miestas, trenerio pavardė, vardas. Žemiau pateikta I rato rezultatų lentelė, išreikšta pelnytais įvarčiais. Suskaičiuokite kiekvienos komandos surinktų taškų skaičių, jei už pergalę skiriami 3 taškai, o už lygiąsias – 1 taškas. Sudarykite komandų turnyrinę lentelę – surikiuokite surinktų taškų mažėjimo tvarka. Jei komandos surinko taškų vienodai, aukščiau ta komanda, kuri turi daugiau pergalų. Suraskite mažiausiai pralaimėjimų turinčią komandą (-as).

**U6–8. Krepšinis**

Pirmoje failo eilutėje nurodytas krepšinio komandų skaičius. Tolesnėse eilutėse pateikta informacija apie krepšinio komandas: pavadinimas, miestas, trenerio pavardė, vardas. Žemiau pateikta I rato rezultatų lentelė, išreikšta įmestais taškais. Suskaičiuokite kiekvienai komandai bendrą įmestų ir praleistą taškų skirtumą. Surikiuokite komandas pagal šį skirtumą mažėjimo tvarka. Sudarykite komandų turnyrinę lentelę (surikiuokite pagal pergalų kiekį mažėjimo tvarka), jei už pergalę skiriami 2 taškai, o už pralaimėjimą – 1 taškas, lygių nebūna. Jei komandos surinko po vienodai taškų, tai aukščiau ta komanda, kuri įmetė daugiau taškų tarpusavio rungtynėse, jei vienodai taškų, vertinkite tarpusavio taškų skirtumą. Suraskite daugiausiai taškų įmetusią komandą. Nustatykite, ar yra komanda, kuri iškovojo visas pergalės.

**U6–9. Detalės**

Pirmoje failo eilutėje nurodytas darbininkų skaičius, mėnesio dirbtų dienų skaičius, vienos detalės įkainis. Tolesnėse eilutėse pateikta informacija apie darbininkus: pavardė, vardas, banko pavadinimas ir sąskaitos numeris. Žemiau pateikta, kiek kiekvieną dieną (stulpeliai) darbininkas (eilutės) pagamino detalių. Suskaičiuokite kiekvienam darbininkui jo atlyginimą. Nustatykite, kuriam darbininkui geriausiai sekėsi dirbtis. Nustatykite, kurią mėnesio dieną buvo pagaminta mažiausiai detalė. Kiekvienam bankui atskirai sudarykite pavedimų sąrašą, kur nurodysite darbininko pavardę, vardą, sąskaitos numerį ir pervedamą sumą.

**U6–10. Leidiniai**

Pirmoje failo eilutėje nurodytas prenumeruojamų leidinių kiekis ir mėnesio dienų skaičius. Tolesnėse eilutėse pateikta informacija apie leidinius: pavadinimas, mėnesio prenumeratos kaina, banko pavadinimas, sąskaitos numeris, procentai, atiduodami prenumeratos rinkėjui. Žemiau pateikta informacija apie kiekvieno leidinio prenumeratos eiga: leidiniai (eilutės), kiek jų užsakyta (stulpeliai). Suskaičiuokite, kiek per mėnesį kurio leidinio užsakyta. Nustatykite, kuriam leidiniui blogiausiai sekasi. Kiekvienam bankui sudarykite pavedimų sąrašą, kad pversti leidiniui prenumeratos pinigus. Nustatykite, kuris leidinys daugiausiai uždirbs.

**U6–11. Paskaitos**

Pirmoje failo eilutėje nurodytas studentų kiekis ir mėnesio dienų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie studentus: studento pavardė, vardas, fakultetas, specialybė. Žemiau pateikta informacija apie studentų paskaitų lankomumą: studentai (eilutės), praleista paskaitų per dieną (stulpeliai). Nustatykite, ar yra dienų, kai paskaitose dalyvavo visi studentai, jei taip, kurios tai dienos. Surikiuokite studentus pagal specialybės. Suskaičiuokite kiekvienai specialybei praleistą valandų kiekį. Nustatykite, ar yra studentų, kurios reikia šalinti (nelankė paskaitų daugiau nei pusę mėnesio dienų). Pašalinkite tokius studentus.

**U6–12. Dėstytojai**

Pirmoje failo eilutėje nurodytas dėstytojų kiekis, mėnesio dienų skaičius, vidutinis dėstytojo mėnesio valandų krūvis. Tolesnėse failo eilutėse nurodyta informacija apie dėstytojus: dėstytojo pavardė, vardas, fakultetas, katedra. Žemiau pateikta informacija apie dėstytojų skaitytų paskaitų kiekį kiekvieną dieną: dėstytojai (eilutės), valandų kiekis per dieną (stulpeliai). Nustatykite, ar yra dienų, kai paskaitas skaitė visi dėstytojai, jei taip, kurios tai dienos. Surikiuokite dėstytojus pagal fakultetus.

Nustatykite, kiekvieno fakulteto dėstytojų dirbtų valandų skaičių. Nustatykite, ar yra dėstytojų, kurie dirbo mažiau, nei vidutinis mėnesio dėstytojo darbo krūvis. Jei taip, kurie tai dėstytojai.

**U6–13. Rankinis**

Pirmoje failo eilutėje nurodytas rankinio komandos pavadinimas, vyr. treneris, rankininkų skaičius ir žaistų rungtynių skaičius. Tolesnėse eilutėse pateikta informacija apie rankininkus: pavardė ir vardas, gimimo metai, masė, žaidimo pozicija. Žemiau atitinkamai rankininkų (eilutės) išdėstymo tvarka pateikta kiekvienose rungtynėse (stulpeliai) įmestų taškų skaičius. Suraskite du rezultatyviausius komandos rankininkus. Atleiskite iš komandos du mažiausiai taškų pelniusius rankininkus (pašalinkite jų rezultatus).

**U6–14. Formulė 1**

Pirmoje failo eilutėje nurodytas lenktynėse dalyvaujančių pilotų skaičius ir trasų skaičius. Tolesnėse eilutėse pateikta informacija apie pilotus: pavardė ir vardas, komanda, variklio pavadinimas. Žemiau pateikta sezono lenktynių atskirose trasose rezultatų lentelė, išreikšta pilotų užimtomis vietomis. Sudarykite pilotų turnyrinę lentelę. Taškai skiriami: 1 vieta – 10 taškų, 2 – 8 t., 3 – 6 t., 4 – 5 t., 5 – 4 t., 6 – 3 t., 7 – 2 t., 8 – 1 t.. Surikiuokite lentelę surinktų taškų mažėjimo tvarka. Suraskite, kuris pilotas daugiausiai kartų lenktynėse buvo antras. Suraskite, kiek ir kokios komandos dalyvavo sezono varžybose.

**U6–15. Semestro darbai**

Pirmoje failo eilutėje nurodytas studentų skaičius ir laboratoriinių darbų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie studentus: studento pavardė, vardas, grupė, specialybė. Žemiau pateikta informacija apie studentų laboratoriinių darbų įvertinimus (realūs skaičiai): studentai (eilutės), įvertinimai (stulpeliai). Nustatykite, ar yra tokiai laboratoriinių darbų, kurių visi įvertinimai teigiami (> 4.5), jei taip, tai kurie. Surikiuokite studentus pagal vidurkius. Suskaičiuokite kiekvienos grupės kiekvieno laboratoriinio darbo įvertinimų vidurkį. Nustatykite, ar yra studentų, kuriuos reikia šalinti (vidurkis < 3). Pašalinkite tokius studentus.

**U6–16. Gripas Lietuvoje**

Pirmoje failo eilutėje nurodytas savivaldybių skaičius ir dienų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie savivaldybes: savivaldybės pavadinimas, savivaldybės dydis kvadratiniais kilometrais, gyventojų skaičius tūkstančiais. Žemiau pateikta informacija apie gyventojų sergamumą – kiek gyventojų kreipėsi į gydymo įstaigas atitinkamomis dienomis: savivaldybės (eilutės), dienos (stulpeliai). Nustatykite, kuriose savivaldybėse ir kuriomis dienomis buvo paskelpta gripo epidemija. Gripo epidemija skelbiama, kai 10 tūkstančių gyventojų tenka x sergančiųjų. Kuriose savivaldybėse gripo epidemija tėsėsi ilgiausiai. Surikiuokite savivaldybes pagal epidemijos dienų skaičių ir savivaldybių pavadinimus.

**U6–17. Gimstamumas Europoje**

Pirmoje failo eilutėje nurodytas valstybių skaičius ir metų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie valstybes: valstybės pavadinimas, valstybės pavadinimo sutrumpinimas, gyventojų skaičius tūkstančiais. Žemiau pateikta informacija apie gimstamumą – kiek naujagimių gimė atitinkamais metais: valsybės (eilutės), metai (stulpeliai). Nustatykite, kuriais metais buvo mažiausias gimstamumas. Suraskite, kuriais metais ir kurioje valstybėje gimstamumas buvo didžiausias 100-tui tūkstančių gyventojų. Surikiuokite valstybes pagal gyventojų skaičių ir valstybių pavadinimus.

**U6–18. Disko metimas**

Pirmoje failo eilutėje nurodytas sportininkų skaičius ir metimų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie sportininkus: pavardė ir vardas, valstybės pavadinimo, sportininko amžius ir sportininko masė. Žemiau pateikta informacija apie diskų nuskrietus atstumus metrais: sportininkai (eilutės), atstumai (6 stulpeliai – bandymai). Suraskite tris sportininkus, užėmusius tris pirmas vietas. Suraskite, kuriai vietą užėmė vyriausias sportininkas. Surikiuokite sportininkus pagal pirmojo bandymo rezultatus ir sportininkų pavardes.

**U6–19. Pražangos**

Pirmoje failo eilutėje nurodytas komandos pavadinimas ir valstybė, krepšininkų skaičius ir žaistų rungtynių skaičius. Tolesnėse eilutėse pateikta informacija apie krepšininkus: vardas ir pavardė, gimimo metai, ūgis, žaidimo pozicija. Žemiau atitinkamai krepšininkų (eilutės) išdėstymo tvarka pateikta kiekvienose rungtynėse (stulpeliai) gautos pražangos. Suraskite du mažiausiai ir du daugiausiai kartų prasižengusius krepšininkus. Kuriose rungtynėse krepšininkai surinko mažiausiai pražangų. Suraskite, kurios pozicijos žaidėjai surinko daugiausiai pražangų.

**U6–20. Kasininkai**

Pirmoje failo eilutėje nurodytas prekybos tinklo kasininkų/-ių skaičius ir dirbtų dienų skaičius. Tolesnėse eilutėse pateikta informacija apie kasininką/-ę: pavardė ir vardas, gimimo metai ir kolas, kurio dirba numeris. Žemiau pateikta, kiek kiekvieną dieną (eilutės) kasininkas (stulpeliai) surinko pinigų aptarnaudamas pirkėjus. Suskaičiuokite kiekvienam kasininkui/-ei jo/-s atlyginimą, jeigu jis yra 1 % nuo kasoje surinktų pinigų. Suskaičiuokite, kokį atlygi gaus jauniausias ir vyriausias kasininkas/-ė. Suraskite, kurią dieną vidutiniškai buvo didžiausi prekių pardavimai.

**U6–21. Būsimieji pirmakursiai**

Pirmoje failo eilutėje nurodytas fakultetų skaičius ir dokumentų pateikimo į universitetą dienų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie fakultetus: fakulteto pavadinimas, planuojamas priimti studentų skaičius, dėstytojų skaičius. Žemiau pateikta informacija apie studentų dokumentų pateikimo skaičius atitinkamomis dienomis: fakultetai (stulpeliai), dienos (eilutės). Nustatykite, kuris pirmasis fakultetas ir kurią dieną įvykdė studentų planuojamo priėmimo planą. Kurie fakultetai neįvykdė studentų priėmimo plano. Kurio fakulteto dėstytojams bus daugiausiai darbo – vienam dėstytojui teks vidutiniškai daugiausiai studentų. Surikiuokite fakultetus pagal priimtų dokumentų skaičius.

**U6–22. Bankų indelių palūkanos**

Pirmoje failo eilutėje nurodytas Lietuvoje veikiančių bankų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie bankus: pavadinimas, valstybė, veikiančių skyrių skaičius Lietuvoje. Žemiau pateikta informacija apie bankų (eilutės) palūkanų dydžius litais 1 mén., 2 mén., 3 mén., 6 mén., 1 metams, 2 metams ir 3 metams. Nustatykite, kuriuose bankuose geriausiai laikyti savo indelius kiekvienam laikotarpiui. Kelių valstybių bankai veikia Lietuvoje? Surikiuokite bankus pagal palūkanas 6 mėnesiams ir bankų pavadinimus.

**U6–23. Darbo birža**

Pirmoje failo eilutėje nurodytas miestų skaičius ir mėnesių skaičius. Tolesnėse failo eilutėse nurodyta informacija apie miestus: miesto pavadinimas, gyventojų skaičius, jaunimo nuo 19 iki 25 metų skaičius. Žemiau pateikta informacija apie jaunimo nuo 19 iki 25 metų nedarbą miestuose: miestai (eilutės), kiek bedarbių registruota kiekvieną mėnesį (stulpeliai). Nustatykite, kurį mėnesį buvo didžiausias nedarbas jaunimo tarpe. Suraskite, kurį mėnesį ir kuriame mieste santykinis nedarbo lygis buvo mažiausias. Surikiuokite miestus pagal jaunimo skaičių ir gyventojų skaičių.

**U6–24. Oficiali emigracija**

Pirmoje failo eilutėje nurodytas savivaldybių skaičius ir mėnesių skaičius. Tolesnėse failo eilutėse nurodyta informacija apie savivaldybes: savivaldybės pavadinimas, meras, gyventojų skaičius tūkstančiais. Žemiau pateikta informacija apie gyventojų emigraciją – kiek gyventojų emigravo atitinkamais mėnesiais: savivaldybės (eilutės), mėnesiai (stulpeliai). Raskite, kiek gyventojų emigravo kiekvieną mėnesį. Nustatykite, kurioje savivaldybėse ir kurį mėnesį emigravo daugiausiai gyventojų. Surikiuokite savivaldybes pagal santykinį emigracijos dydį, ivertinant savivaldybių gyventojų skaičių ir savivaldybių pavadinimus.

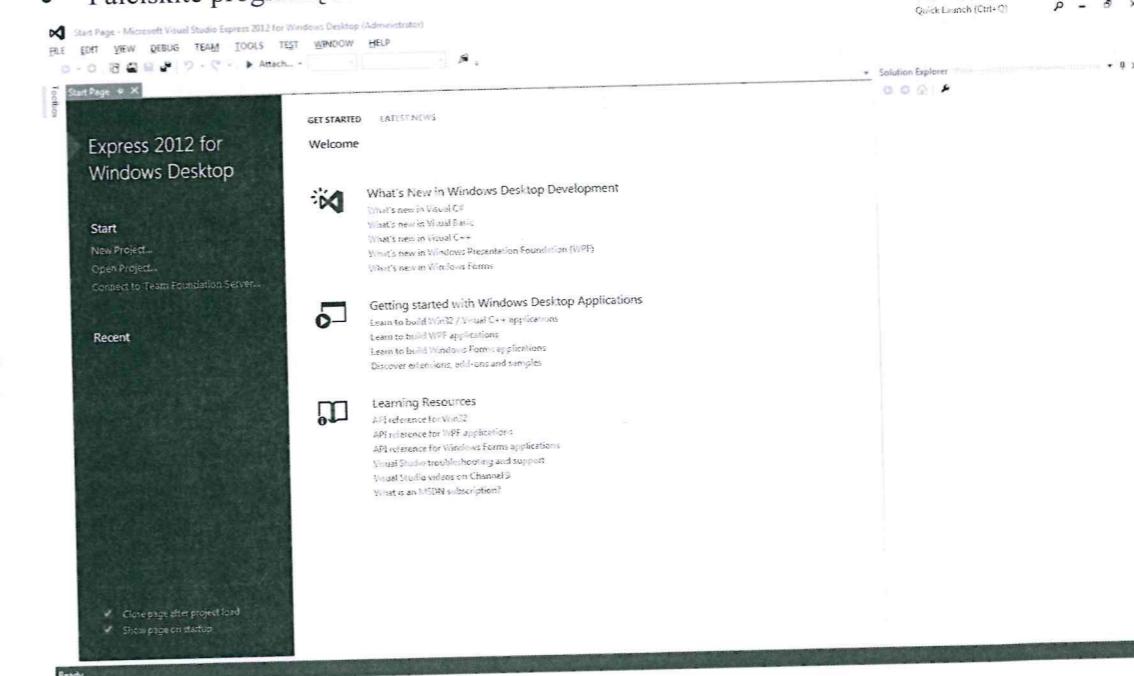
**1 priedas****Microsoft Visual C++ 2012 programavimo aplinka**

Susipažinsite su Microsoft Visual C++ programavimo aplinka ir programavimo kalbos C++ pagrindinėmis konstrukcijomis:

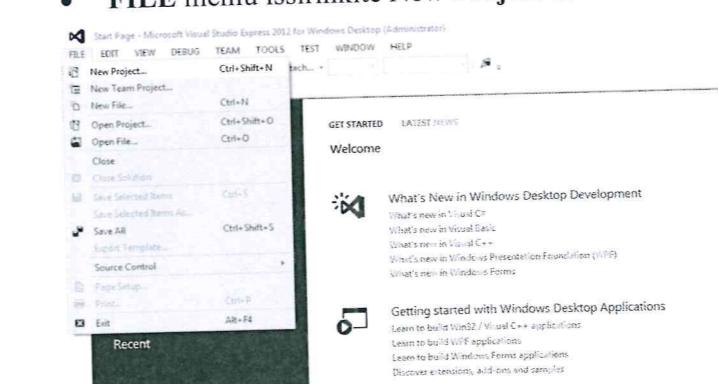
- programavimo aplinkos konsoliniu darbo režimu;
- C++ programos struktūra ir pagrindine funkcija `main()`;
- duomenų spausdinimu ekrane;
- duomenų įvedimu klaviatūra;
- `int` ir `double` tipo kintamaisiais;
- priskyrimo operacija;
- dialogo pradmenimis.

**1 Pirmas žingsnis.** Programos sukūrimo scenarijus

- Paleiskite programą **Microsoft Visual Studio Express 2012**.

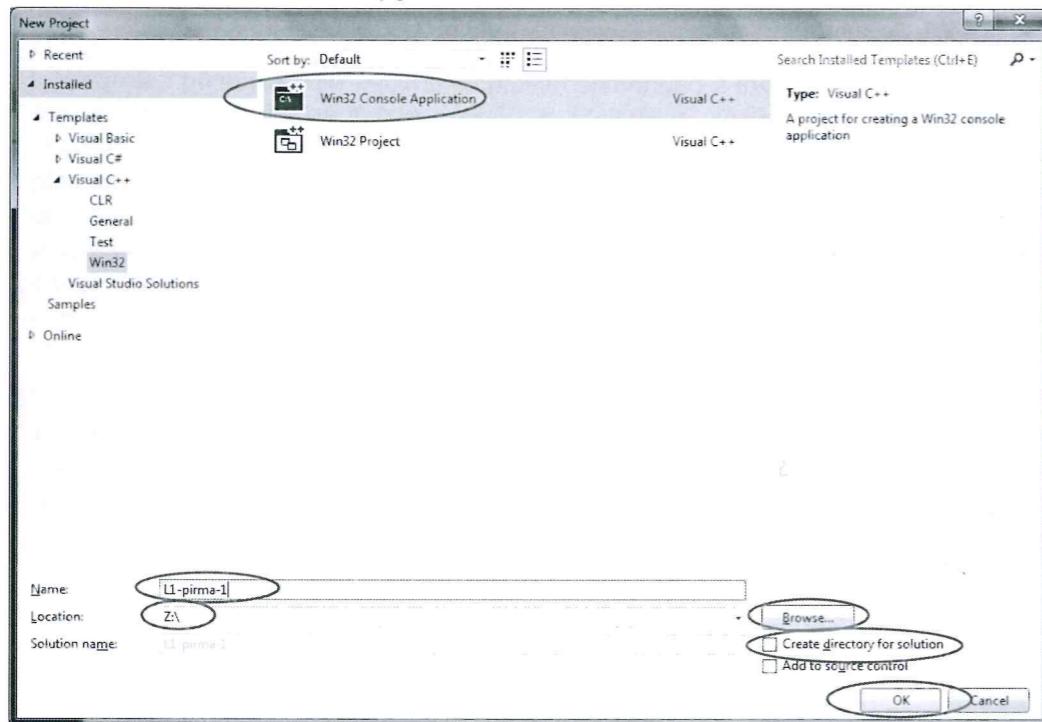


- Sukurkite naują projektą (naujos programos kūrimas **Microsoft Visual Studio Express** aplinkoje visada prasideda nuo projekto kūrimo):
- **FILE** menu išsirinkite **New Project ...**

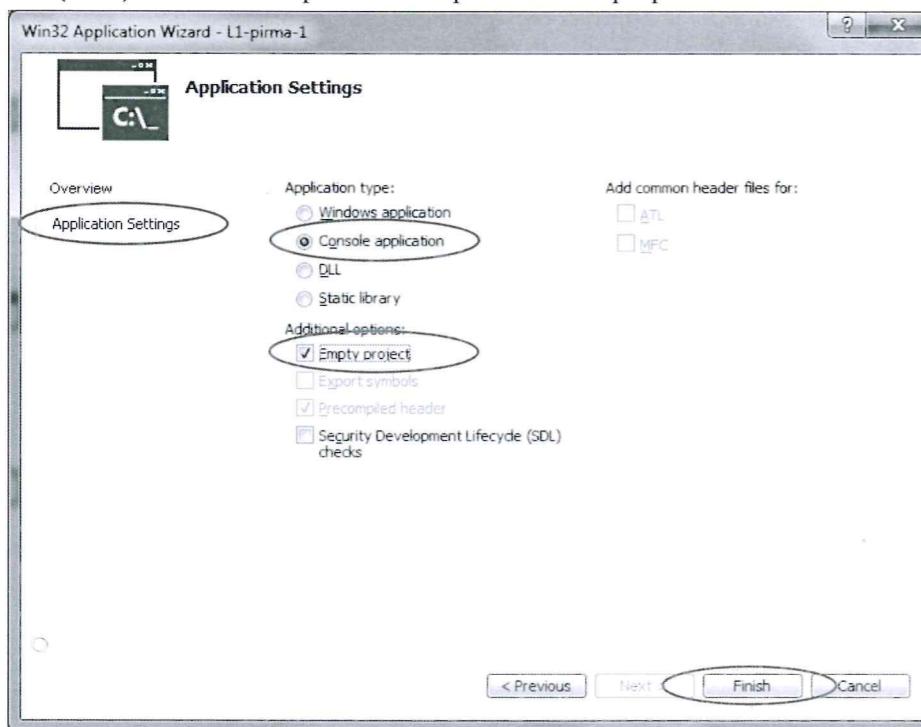


- Iš **Visual C++** projektų tipų pasirinkite **Win32 Console Application**.
- Irašykite naujo projektovardą **L1pirma-1**.

- Pasirinkite diską ir katalogą, kur saugosite programą: spauskite mygtuką **Browse** ir išsirinkite **D** diską, susietą su konkrečiu kompiuteriu. Programas derinimo metu reikia saugoti **D** diske. Pastoviam saugojimui reikia naudoti **Z** diską. **Z** diskas – tai jūsų asmeninė sritis serveryje. Programos saugojimas derinimo metu **Z** diske labai apkrauna tinklą.
- Nuimkite varnelę **Create directory for solution** (papildomi katalogai nereikalingi).
- Patvirtinkite pasirinkimą mygtuku **OK**.



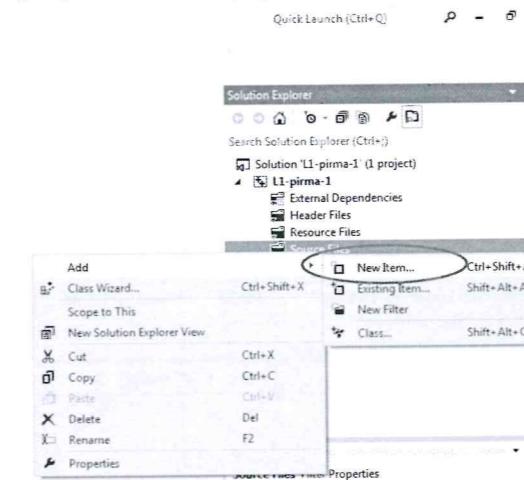
- Win32 Application Wizard** lange pasirinkite **Application Settings**, pažymėkite **Console application** ir **Empty Project** bei numkite pažymėjimą nuo **Security Development Lifecycle (SDL) checks**. Tada patvirtinkite pasirinkimus spausdami **Finish**.



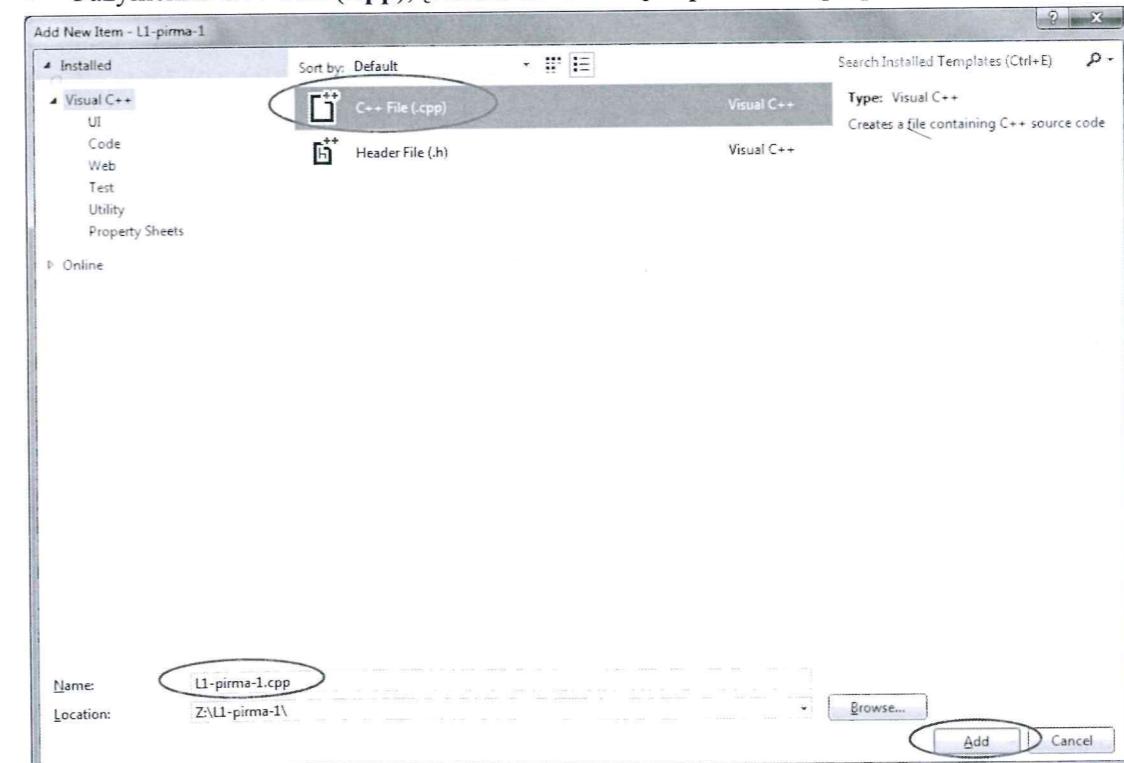
- Projektas jau sukurtas. Jei kairėje lange pusėje **Solution Explorer** nematomas, pasirinkite **View → Solution Explorer**.



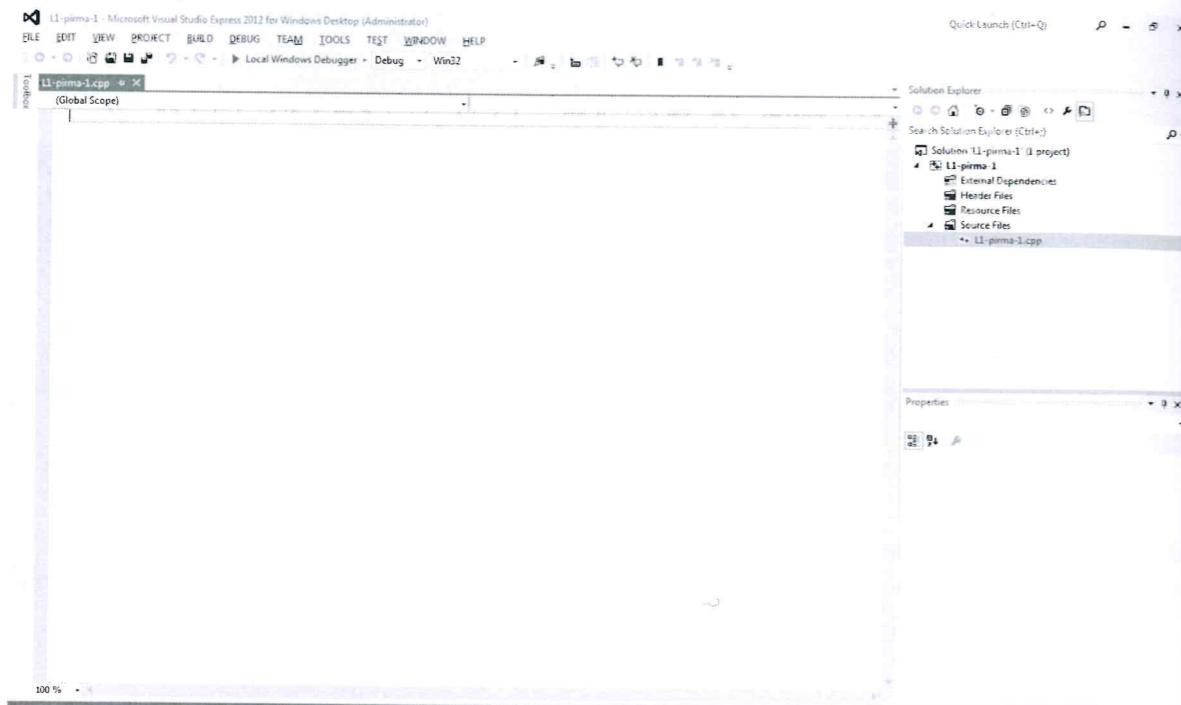
- Projekto papildymas C++ kalbos programos teksto failu. **Solution Explorer** lange ant **Source Files** katalogo spauskite dešinįjį pelės klavišą. Išsirinkite **Add**, paskui spauskite **New Item**.



- Pažymėkite **C++ File (.cpp)**, įveskite failo vardą **L1irma-1** ir spauskite **Add**.

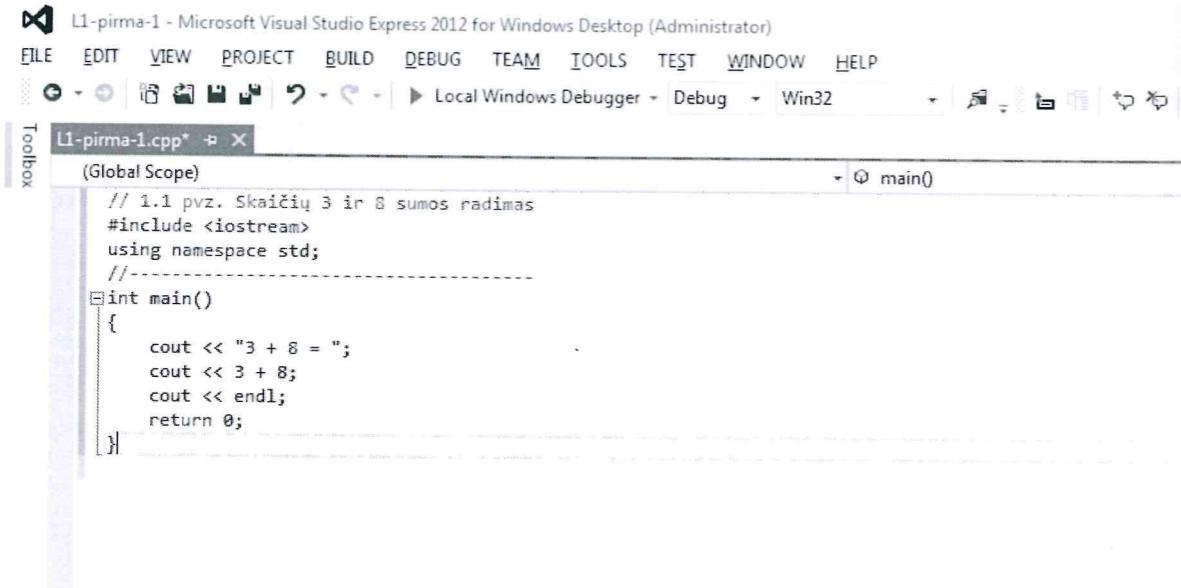


- Source Files kataloge pasirodo failas L1pirma-1.cpp, žymeklis atsiranda didžiajame programos teksto lange.



#### ¶ Antras žingsnis. Pirmoji programa.

- Klaviatūra įveskite programos tekštą, kuris yra paveikslė:



- Išsaugokite įvestą programą: paspauskite Save All piktogramą .
- Suskaičiuokite laukiamą rezultatą:  $3 + 8 = 11$ .
- Kompiliuokite programą: Debug meniu paspauskite Build Solution.
- Visual C++ lango Output dalyje matysite kompiliavimo eigos pranešimus. Jei įvedimo metu nepadarėte klaidų, pranešimų pabaigoje bus nurodyta, kad kompiliavimas ir vykdomojo programos failo sukūrimas buvo sėkmingas:

#### Output

Show output from: Build  
1>----- Build started: Project: L1-pirma-1, Configuration: Debug Win32 -----  
1> L1-pirma-1.cpp  
1> L1-pirma-1.vcxproj -> Z:\L1-pirma-1\Debug\L1-pirma-1.exe  
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====

- Įvykdykite programą: Debug meniu paspauskite Start Debugging.
- Palyginkite gautą rezultatą su laukiamu.
- Pasižiūrėkite į katalogą L1pirma-1 ir nustatykite, kokius failus sukuria sistema.
- Baikite darbą su programavimo aplinka File meniu paspausdami Exit.
- Iš disko D pašalinkite katalogą L1pirma-1.

#### ¶ Trečias žingsnis. Sintaksės klaidų taisymas, diagnostinių pranešimų analizė.

- Sukurkite projektą L1pirma-2, kurį išsaugosite D diske. I projektą įtraukite .cpp failą, kurį pavadinsite L1pirma-2.
- Įveskite tokį programos tekštą:

```
//-----  
// 1.2 pvz. Dvieju kintamuju reikšmių sumos radimas  
// Démesio: programoje yra klaida!  
#include <iostream>  
using namespace std;  
//-----  
int main()  
{  
    setlocale(LC_ALL, "Lithuanian");  
    int a = 3, // pirmasis kintamasis  
        b = 8; // antrasis kintamasis  
    Int suma = a + b // skaiciu sumai saugoti  
    cout << a << " + " << b << " = ";  
    cout << suma  
    cout << endl;  
    return 0  
}
```

Pirmoji klaida –  
reikia kabliataškio

- Išsaugokite įvestą programą.
- Suskaičiuokite laukiamą rezultatą.
- Kompiliuokite programą.
- Išanalizuokite pranešimus apie klaidas. Taisykite klaidas po vieną. Programoje yra 5 klaidos (trūksta 4 kabliataškių int tipas privalo prasidėti mažaja raide).
- Remdamiesi diagnostinių pranešimų informacija, suraskite ir ištaisykite programos tekštą. Kompiliuokite programą ištaisę kiekvieną klaidą.
- Išsaugokite ir įvykdykite pataisytą programą. Palyginkite gautą rezultatą su laukiamu.
- Baikite darbą su sistema.
- Katalogą L1pirma-2 iš disko D perkelkite į diską Z.

#### ¶ Ketvirtas žingsnis. Duomenų įvedimas

- Sukurkite projektą Llantra-1, kurį išsaugosite D diske. I projektą įtraukite .cpp failą, kurį pavadinkite Llantra-1.
- Įveskite tokios programos tekštą:

```
//-----  
// Įvedamų kintamuju reikšmių sumos radimas  
#include <iostream>
```

```
using namespace std;
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    int a; // pirmasis kintamasis
    int b; // antrasis kintamasis
    int suma; // kintamasis skaičių sumai saugoti
    cout << "Įveskite a, b reikšmes:" << endl;
    cin >> a >> b;
    suma = a + b;
    cout << "a + b = " << suma << endl;
    return 0;
}
//-----
```

- Išsaugokite įvestą programą.
- Kompiliuokite ir įvykdykite programą. Vykdymo metu įveskite tokias kintamuju reikšmes: 10 8.
- Pakartokite sumos skaičiavimą su kitomis reikšmėmis.
- Baikite darbą su sistema.
- Katalogą Llantra-1 iš diskos D perkelti į diską Z.

#### Penktas žingsnis. Rezultatų tikrinimas

- Sukurkite projektą Llantra-2, kurį išsaugokite D diske. Į projektą įtraukite .cpp failą, kurį pavadininkite Llantra-2.
- Įveskite nupjautinio kūgio tūrio skaičiavimo programos tekštą:

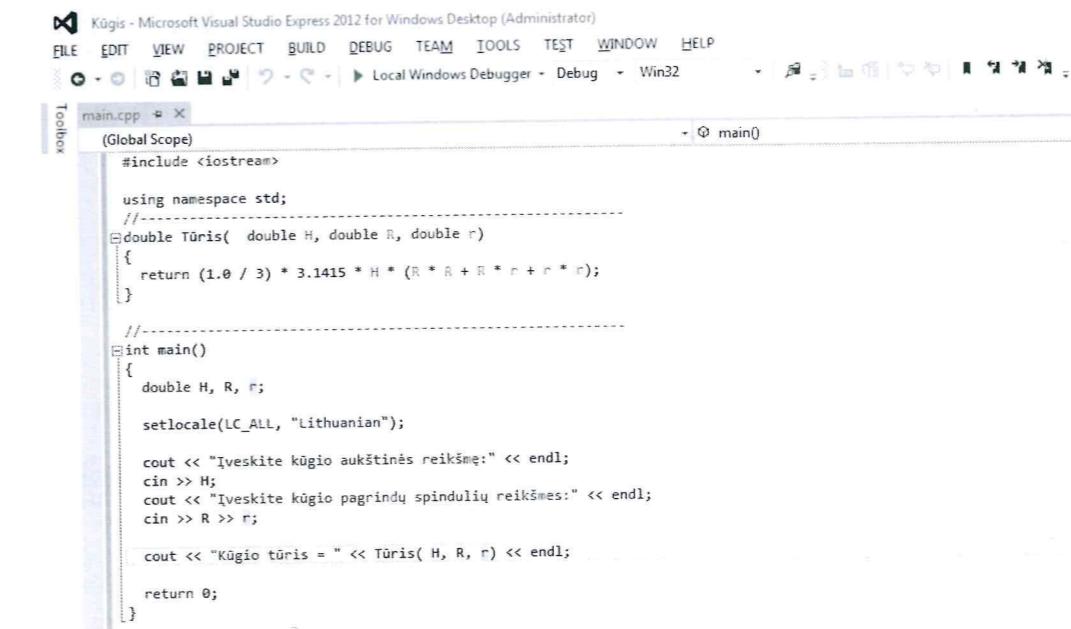
```
//-----
// Nupjautinio kūgio tūrio skaičiavimas
// Programa skaičiuoja kūgio tūri V, kai kūgio pagrindu
// spinduliai R, r ir aukštinė H įvedami klaviatūra
#include <iostream>
using namespace std;
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    double pi = 3.1415; // matematinė konstanta
    double H; // kūgio aukštinė
    double R, r; // kūgio pagrindu spinduliai
    double V; // kūgio tūris
    cout << "Įveskite kūgio aukštinės reikšmę:" << endl;
    cin >> H;
    cout << "Įveskite kūgio pagrindu spinduliu reikšmes:" << endl;
    cin >> R >> r;
    V = (1.0 / 3) * pi * H * (R * R + R * r + r * r);
    cout << "Kūgio tūris = " << V << endl;
    return 0;
}
//-----
```

- Išsaugokite įvestą programą.
- Kompiliuokite ir įvykdykite programą. Vykdymo metu įveskite tokias duomenų reikšmes: aukštinė: 10.5, pagrindų spinduliai: 8.5 ir 2.3. Palyginkite programos gautą kūgio tūrio reikšmę su reikšme, suskaičiuota skaičiuotuvu.
- Pakartokite kūgio tūrio skaičiavimą, kai aukštinė lygi 15, o pagrindų spinduliai – 20.5 ir 6.4. Gautą reikšmę palyginkite su reikšme, suskaičiuota skaičiuotuvu.
- Baikite darbą su sistema.
- Katalogą Llantra-2 iš diskos D perkelti į diską Z.

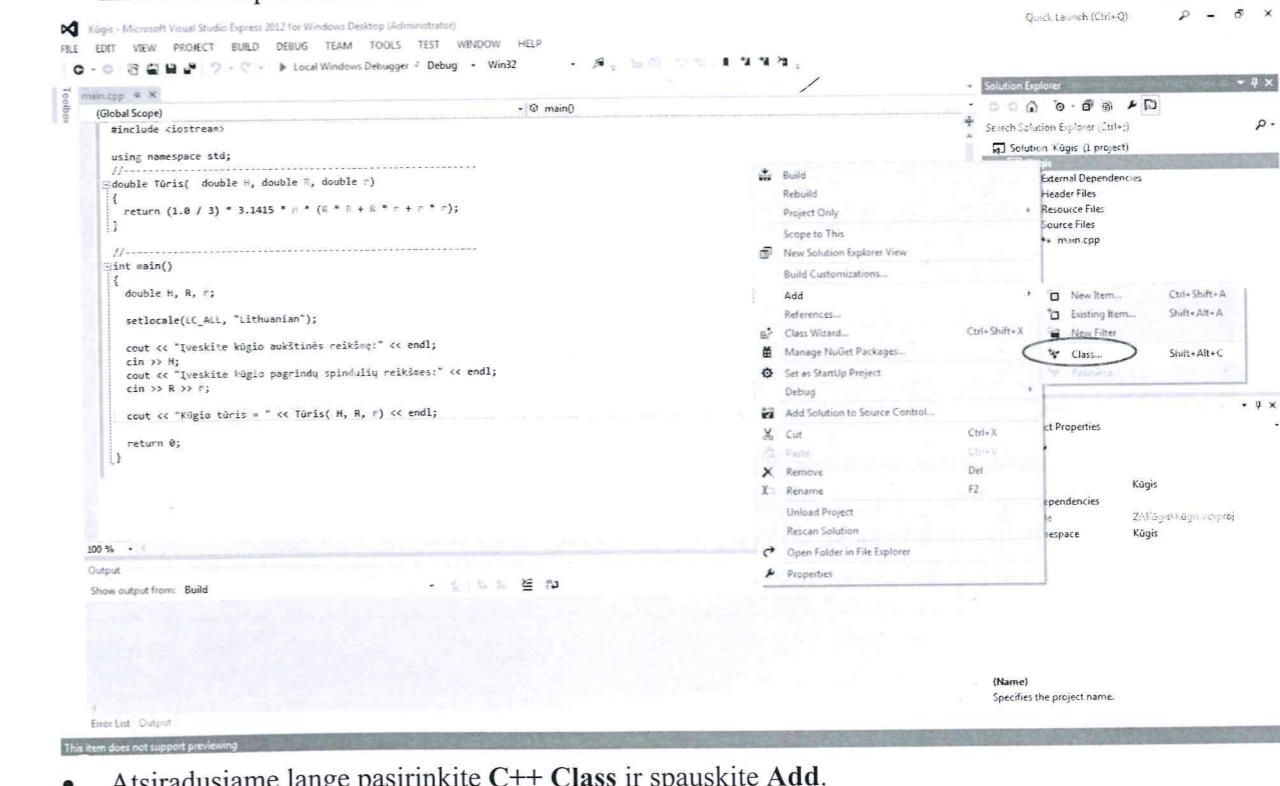
## 2 priekas

### Microsoft Visual C++ 2012 naujų klasų kūrimas (įtraukimas į projekta)

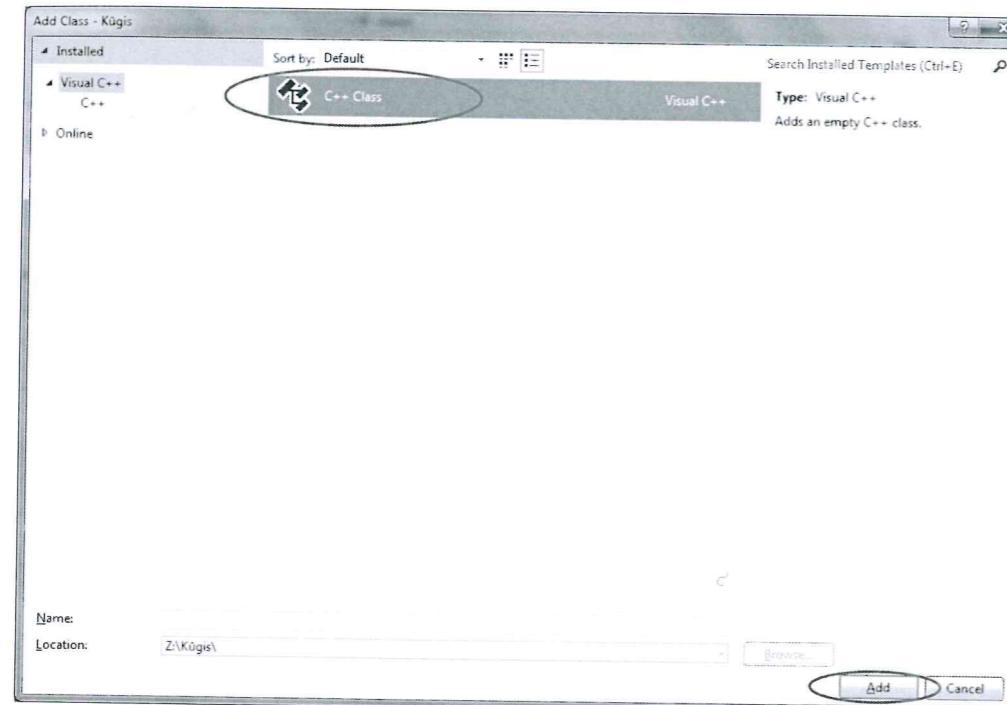
- Sukukite programą, kuri skaičiuoja kūgio tūrį.



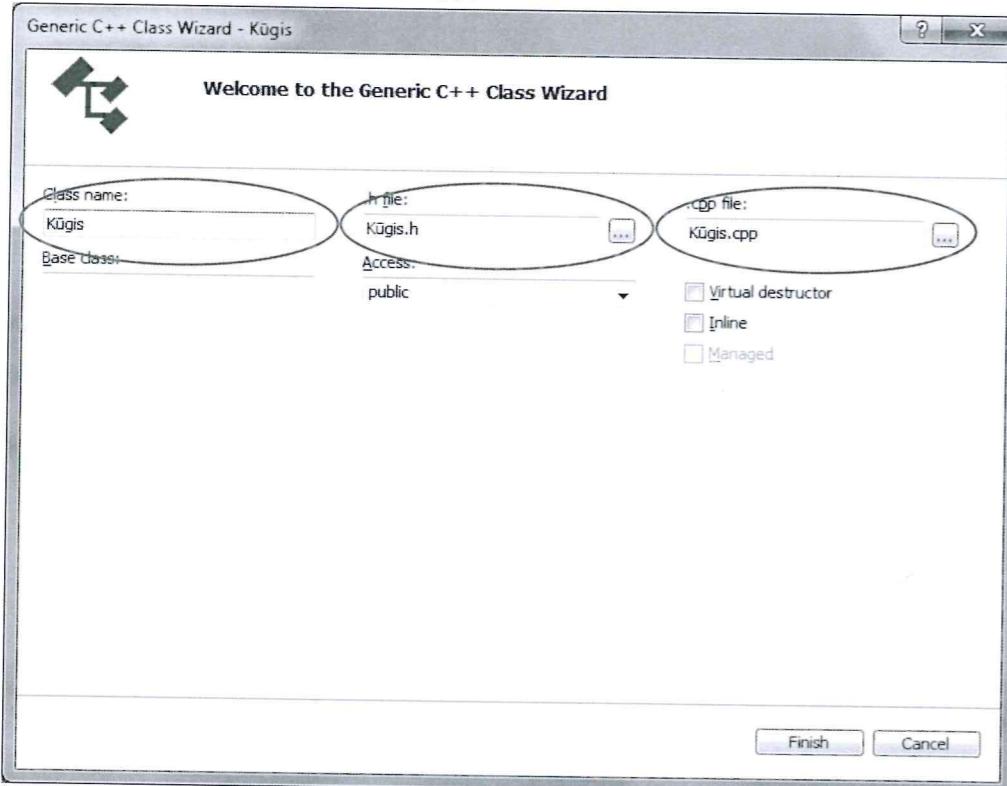
- Šią kūgio tūrio skaičiavimo programą suskaidykite į atskirus modulius. Atskirame moduliuje turi būti klasė Kūgis.
- Projekto navigavimo lange pažymėjus projektą ir paspaudus dešinį pelės klavišą, kontekstiniame meniu reikia pasirinkti Add → Class.



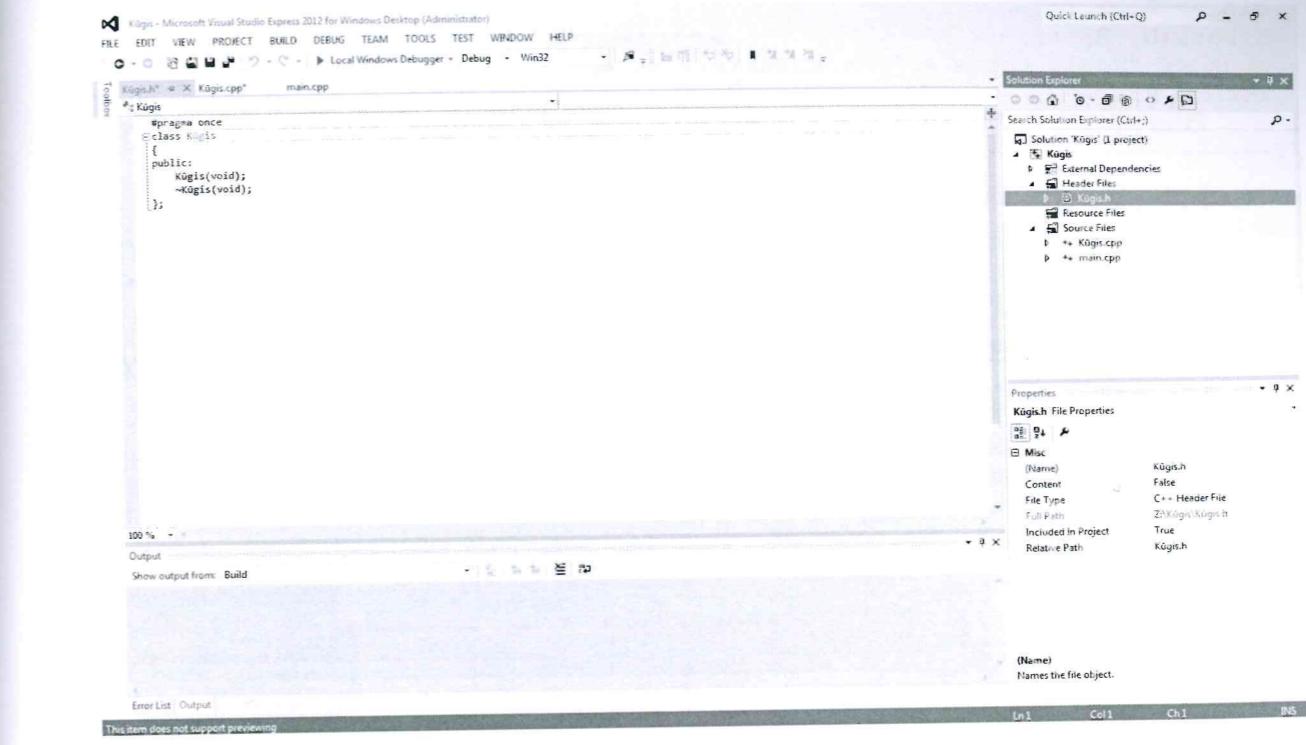
- Atsiradusiam lange pasirinkite C++ Class ir spauskite Add.



- Tolimesniame lange įrašykite klasės vardą (Kūgis). Sistema automatiškai užpildo failų vardu laukelius (Kūgis.h ir Kūgis.cpp).



- Po šiuo veiksmu programos projekto kataloge bus sukurti nurodyti failai, ir atsisas projekto navigavimo lange atitinkamose skiltyse.
- Tam tikra klasės Kūgis teksto dalis sugeneruojama taip pat automatiškai.



- Papildykite failą Kūgis.h.

```
//-----
#pragma once
class Kūgis
{
private:
    double H; // kūgio aukštinė
    double R, r; // kūgio pagrindu spinduliai
public:
    Kūgis(void):H(0), R(0), r(0) {}
    ~Kūgis(void) {}
    void Dėti(double a, double b, double c) { H = a; R = b; r = c; }
    double Tūris();
};
```

- Papildykite failą Kūgis.cpp.

```
//-----
#include "Kūgis.h"
#pragma once
//-----
double Kūgis::Tūris()
{
    return (1.0 / 3) * 3.1415 * H * (R * R + R * r + r * r);
}
```

- Pakeiskite pagrindinės programos failą.

```
//-----
#include <iostream>
#include "Kūgis.h"
using namespace std;
//-----
int main()
{
    setlocale(LC_ALL, "Lithuanian");
    Kūgis K; // objektas
    double H, R, r; // kūgio aukštinė, pagrindo spindulys, nuopjovos spindulys
    cout << "Įveskite kūgio aukštinės reikšmę:" << endl;
    cin >> H;
    cout << "Įveskite kūgio pagrindų spindulių reikšmes:" << endl;
```

```

    cin >> R >> r;
    K.Deti(H, R, r);
    cout << "Kūgio tūris = " << K.Tūris() << endl;
    return 0;
}
//-----

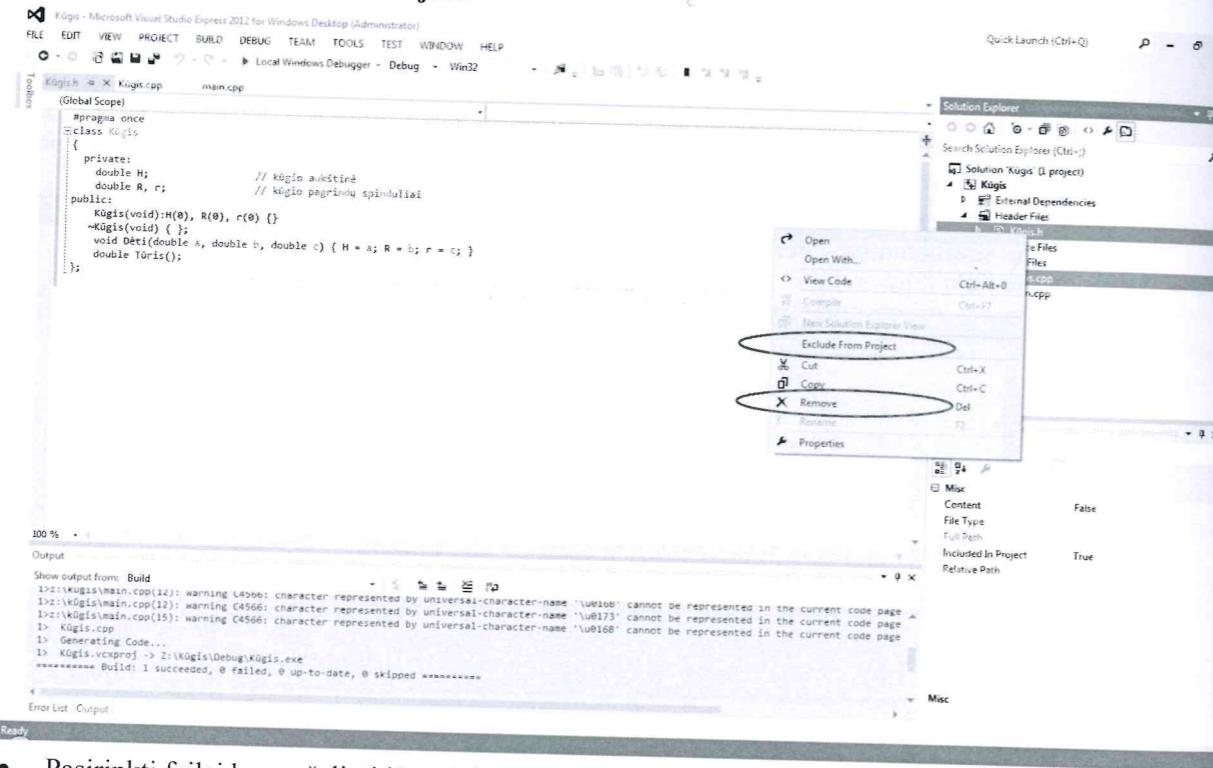
```

- Ivykdykite programą ir palyginkite gautus rezultatus su ankstesnės programos rezultatais.

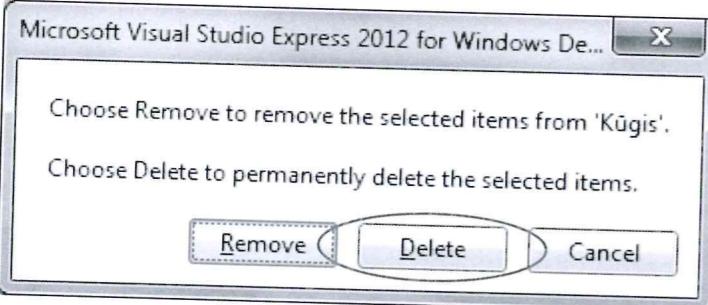
### 3 priedas

#### Microsoft Visual C++ 2012 klasį pašalinimas iš projekto

- Jei tam tikrų klasių (ar failų) jau neberekia, juos galima iš kuriamos programos pašalinti.
- Pažymėjė norimus šalinti failus, paspauskite dešinijį pelės klavišą ir kontekstiniame menui pasirinkite **Exclude from Project**.



- Pasirinkti failai bus pašalinti iš projekto, tačiau liks failų sistemoje.
- Jeigu norite failus pašalinti ir iš failų sistemos, kontekstiniame menui reikia pasirinkti **Remove**, o atsiradusiam lange - **Delete**.



- Jeigu šiame lange pasirinksite **Remove**, failai bus pašalinti iš projekto, tačiau liks failų sistemoje.

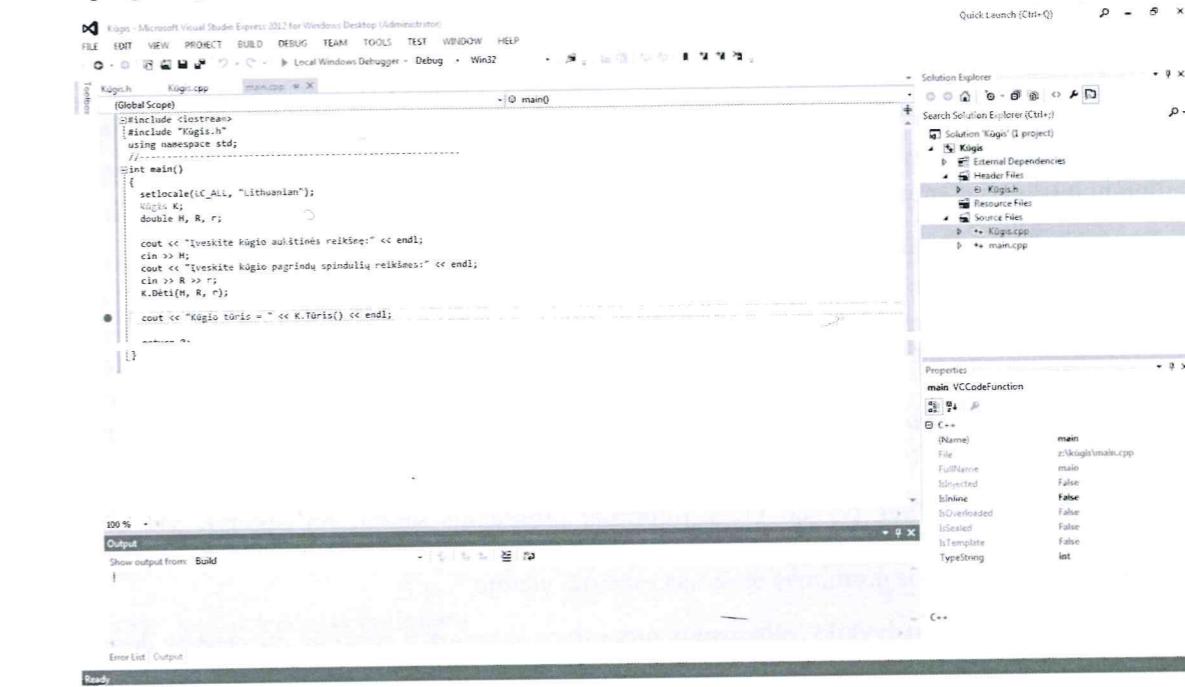
### 4 priedas

#### Microsoft Visual C++ 2012 programų derinimo priemonės

Be programos derinimo priemonių ir įrankių (programos derintuvės) savo darbo neįsivaizduoja nė vienas programuotojas. **Microsoft Visual Studio Express 2012** programos derintuvės pagrindinės teikiamos galimybės:

- sustabdyti programą bet kuriame norimame taške;
- vykdyti programą nuo norimos vietas po vieną komandą;
- matyti ir sekti kintamųjų reikšmes.

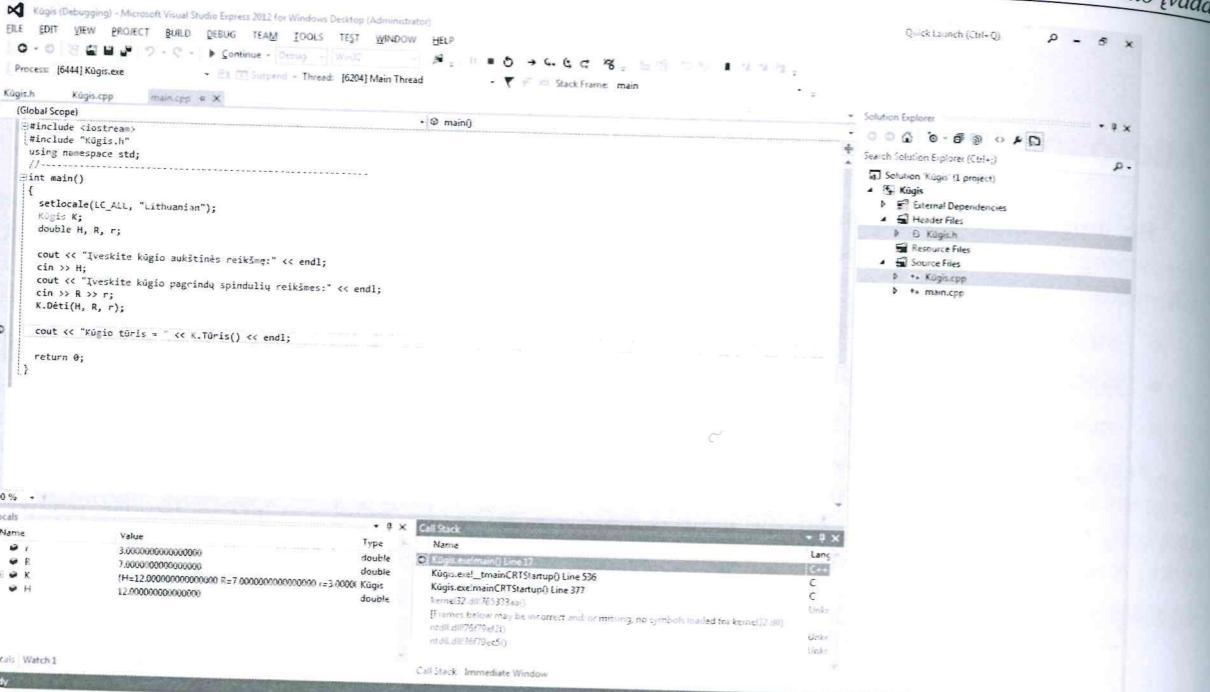
Prieš pradedant programos derinimą, reikia nustatyti programos stabdymo taškus (angl. *breakpoint*). Programos stabdymo taškas nustatomas spragtelėjus pele norimos eilutės kairėje, paraštėje. Paraštėje atsiranda raudonos taškas, žymintis programos stabdymo tašką. Jei stabdymo taškas daugiau nereikalingas, reikia spragtelti pele raudoną tašką, ir jis dinga. Programoje galima nustatyti norimą stabdymo taškų kiekį.



Norint pradeti programos derinimą, reikia iš meniu **Debug** pasirinkti komandą **Start Debugging** arba paspausti klavišą **F5**. Bus atliekami visi programos veiksmai iki pirmojo stabdymo taško. Tada programos darbas laikinai sustabdomas, ir visas valdymas perkeliamas į **Microsoft Visual Studio Express 2012** aplinką. Šiuo metu aktyvių eilutę žymi kairėje paraštėje atsiradusi geltona rodyklė. Lango apačioje matomi keletas svarbių langų:

- Watch** lange galima stebeti bet kurio kintamojo (objekto) duomenis. Stebimi kintamieji gali būti paprasčiausiai pažymimi programos tekste ir įtraukiami į šį langą pele, arba jų vardai įrašomi lange klaviatūra. Lange rodomi kintamųjų vardai, saugomos reikšmės ir tipai.
- Locals** lange yra rodomi šiuo metu aktyvios apibrėžimo sritis (vykdomas funkcijos) kintamieji ir jų duomenys. Čia savo stebimų kintamujų įkelti negalima. Svarbu tai, kad čia galima keisti kintamųjų reikšmes.
- Immediate Window** lange yra leidžiama įrašyti savo sakinius (kodo fragmentus), kurie čia pat gali būti ir įvykdyti, esant laikinai sustabdytai derinamai programai.

## Programavimo įvadas



Programos valdymui persikelus į **Microsoft Visual Studio Express 2012** aplinką, ekrane (arba bendroje įrankių juoste) atsiranda programos derinimo įrankių juosta



Mygtuko → Show Next Statement dešinėje esantys trys mygtukai yra labai svarbūs programos derinimo procesui:

- mygtukas ⏪ Step Into įvykdo vieną tolimesnį programos sakinį. Jei sakinyje yra kreipinių į funkcijas, programos derinimas vykdomas ir tose funkcijose.
- mygtukas ⏴ Step Over įvykdo vieną tolimesnį programos sakinį. Jei sakinyje yra kreipinių į funkcijas, programos derinimas tose funkcijose neatliekamas, t.y. funkcijos įvykdomos, gaunamas jų rezultatas, bet programos derinimas tēsiamas esamoje vietoje.
- mygtukas ⏵ Step Out įvykdo tolimesnius programos sakinius ir perkelia programos derinimą į tą vietą, kurioje buvo kreipinys į šiuo metu derinamą funkciją.
- Naudojant šias priemones įmanomas derinamą programą vykdyti po vieną komandą (eilutę). Derinimo metu galima stebeti kintamųjų reikšmes ir jų pasikeitimus.

## 5 priedas

### Programos skaidymas į modulius

Programos modulis – tai logiškai tarpusavyje susijusių programos funkcijų ir kintamųjų rinkinys, apjungtas į vieną failą. Programos moduliai paprastai saugomi atskiruose failuose. Kiekvieno modulio C++ kodas saugomas .cpp failuose, o sukompliuotas mašininis kodas saugomas .obj failuose.

Į savo programą atskirus modulius galima įtraukti naudojant #include direktyvą. #include yra papildomo programos kodo apdorojimo direktyva. Šis papildomas apdorojimas visada atliekamas prieš kompliliuojant programos kodą C++ kompiliatoriumi. Yra ir daugiau direktyvų papildomam kodo apdorojimui.

#include direktyva įtraukia kitus failus, žinomus kaip antraštės failus (angl. include file, header file, arba header), į programos kodą. Apdorojimo metu #include direktyva yra pašalinama, o vietoje jos įterpiamas tekstas iš antraštės failo, kurio vardas yra nurodomas šalia direktyvos tarp <> arba "" simbolių. Programos modulio kodas nepasikeičia, o kompiliatorius po apdorojimo „mato“ daugiau kodo. Tačiau #include direktyva įtakoja vardu, apibrėžtų antraštės failą, galiojimo sritį ir trukmę.

## Programavimo įvadas

Antraštės (arba \*.h) failai – tai tekstiniai failai, kuriuose taip pat yra C++ kodas. Pagrindinė jų paskirtis – naujų tipų (klasių) apibrėžimas, konstantų paskelbimas ir funkcijų prototipų išvardinimas. Paprastai antraštės failo apibrėžiami tik nauji tipai (klasės), skelbiamos konstantos, išvardinami funkcijų prototipai. Antraščių failuose paprastai neskelbiami kintamieji ir nerašomi vykdomieji sakiniai. Tiesiogiai .h failai nekompliliuojami (tai atliekama tik įtraukus jį į .cpp failą).

Antraštės failų poreikis atsiranda kompliliuojant iš kelių modulių sudarytas programas. Programuotojo apibrėžti duomenų tipai, konstantos ir funkcijos galioja tik tame modulyje, kuriamo jie yra paskelbti. Jeigu tokį pačių duomenų tipų reikės kituose moduliuose, tai juos kiekviename modulyje teks paskelbti iš naujo. Dėl šios priežasties padidėja programos teksto apimtis, atsiranda tikimybė suklysti. Pavyzdžiu, aprašant klasę, viename iš modulių sukeitus kintamuju tvarką, programos veikimas gali būti neprognozuojamas. Antraštės failai panaikina šiuos du trūkumus. Tačiau, jeigu antraštės failai būtų kompliliuojami atskirai, tai kompiliatorius juos laikytų atskirais programos moduliais ir problemos išliktų, t.y. tipai būtų nematomi kituose moduliuose. Papildomo apdorojimo, kuris vykdomas prieš kompliliavimą, direktyva #include išsprendžia šią problemą. Su #include antraštės failas yra įtraukiama į modulio kodą. Kompiliatorius „mato“ modulį, kurio pradžioje įkeltas visas tekstas iš antraštės failo, kaip vieną failą. Tokiu būdu, antraštės failą galima įtraukti į visus modulius, kuriuose jis reikalingas.

Direktyvą #include išprasta rašyti programos modulio failo pradžioje. Galima #include rašyti ir antraštės failo pradžioje, tačiau to reikėtų vengti.

Rašant direktyvą #include tik programos modulių failuose išvengiama programuotojo sukurtų tipų vardų dubliavimo. Naujo duomenų tipo vardas jo galiojimo srityje gali būti paskelbtas tik vieną kartą. Įtraukiant #include direktyvą į antraštės failą lengva suklysti ir sukurti situaciją, kai viename modulio faile tas pats tipas ir jo vardas bus aprašytas kelis kartus.

Direktyvas #include reikia rašyti tokia tvarka, kad tame pačiame modulyje tie patys tipai nebūtų paskelbti kelis kartus. Taip pat reikia atsižvelgti į vienų tipų naudojimą kituose. Pirmiausiai įtraukiami tie antraštės failai, kuriuose yra moduliu reikalingi išvestiniai tipai ir konstantos ir juose naudojami tik baziniai tipai. Po to įtraukiami moduliu reikalingi antraštės failai, kuriuose paskelbti išvestiniai tipai naudoja kitus išvestinius tipus.

## 6 priedas

### CodeBlocks programavimo aplinka\*

Susipažinsite su *CodeBlocks* aplinka:

- sukursite darbo katalogą ir programos failą;
- pakeisite programos pavadinimą, programą įrašysite į darbo katalogą;
- sukompiliuosite ir įvykdysite paprasčiausią programą;
- išmoksite programą taisyti;
- išmoksite išvesti duomenis į ekraną naudodami išvesties srautą cout.

Kiekviena *CodeBlocks* programa sukuria keletą failų. Todėl, prieš pradėdami darbą, susurkite atskirą katalogą kiekvieno praktikos darbo failams laikyti.

**Pirmas žingsnis.** Darbo katalogo kūrimas.

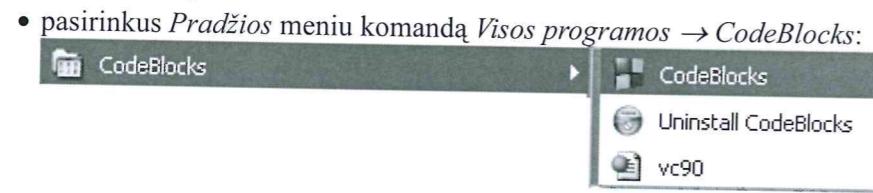
- Kurioje nors laikmenoje, naudodamiesi įprastomis Windows sistemos priemonėmis, susurkite bendrą katalogą, kuriame laikysite visus savo *CodeBlocks* darbus. Katalogą galite pavadinti savo pavarde, pavyzdžiu, *Pavardenis*.
- Šiame kataloge surinkite pakatalogi *Darbas1*, skirtą pirmajam darbui.

**Antras žingsnis.** *CodeBlocks* paleidimas.

Norint pradėti dirbti, reikia, kad kompiuterje būtų įdiegta kuri nors *CodeBlocks* versija. Visi šios knygos pavyzdžiai sukurti naudojantis *CodeBlocks* 10.05 versija.

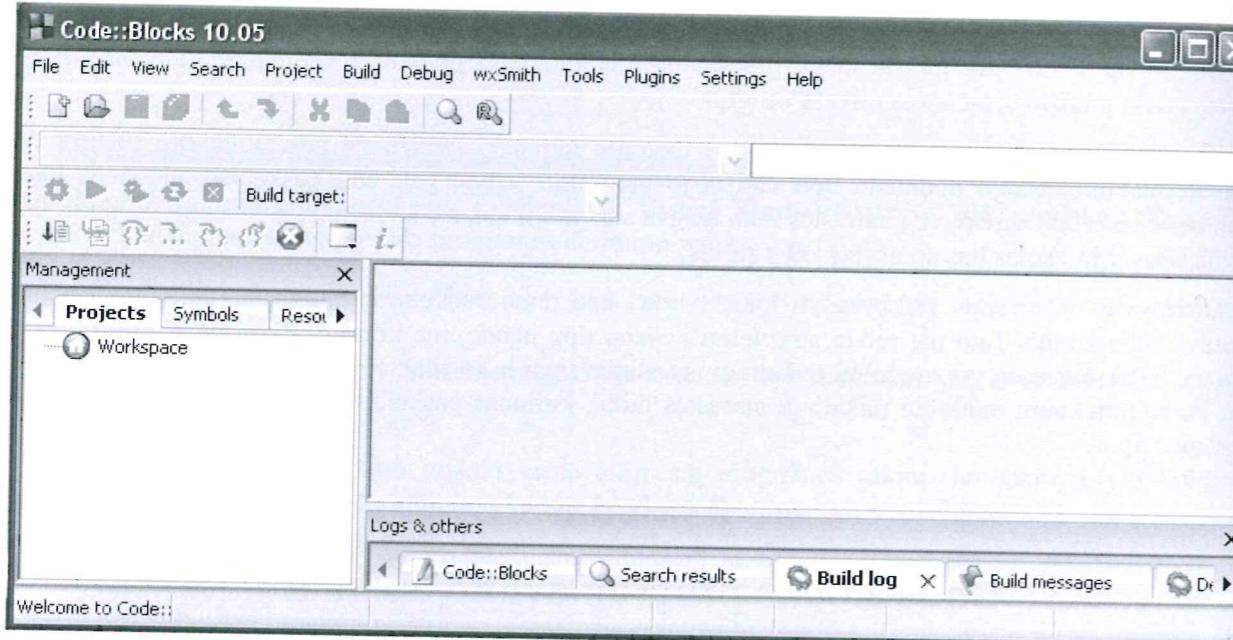
\* J.Blonskis, V.Bukšnaitis, R.Burbaitė. Šiuolaikiškas žvilgsnis į programavimo pagrindus C++. Pasirenkamasis informacinių technologijų kursas IX-X klasėms. Vilnius, TEV, 2011

Ją galima rasti interneite (<http://prdownload.berlios.de/codeblocks/codeblocks-10.05mingw-setup.exe>).  
CodeBlocks paleisti galima:



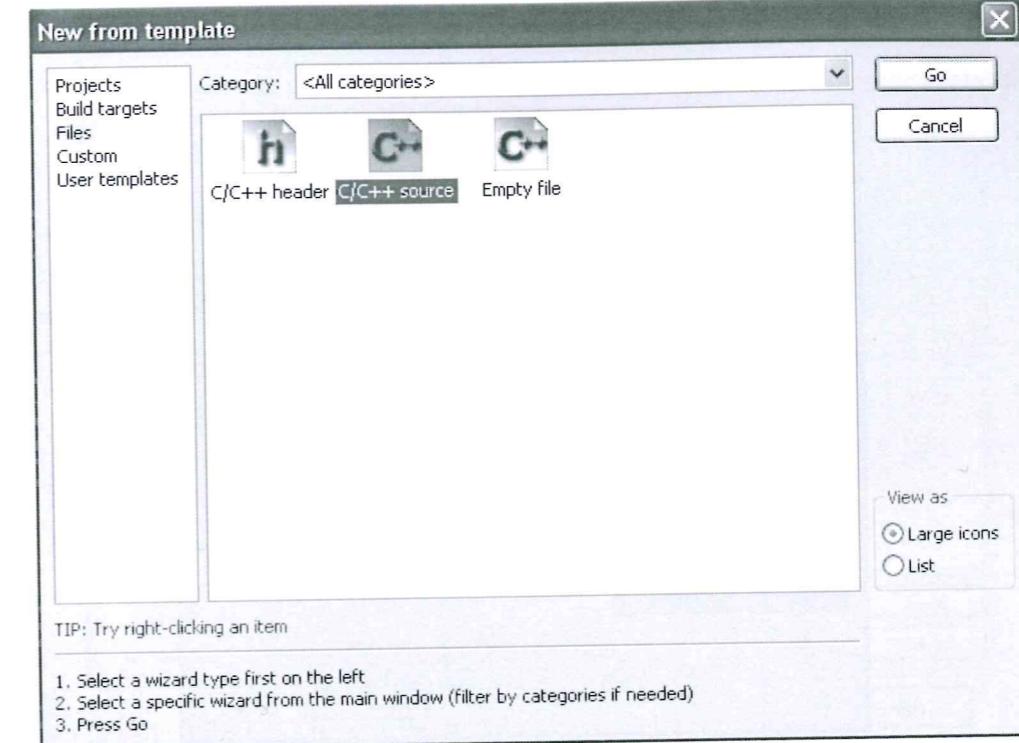
- dukart spragtelėjus darbalaukyje šaukinį CodeBlocks

Sėkmingai įvykdę nurodymus, pateksite į *CodeBlocks* langą, kuris valdomas (padidinamas, sumažinamas, pernešamas ar užveriamas) išprastomis operacinės sistemos *Windows* priemonėmis.

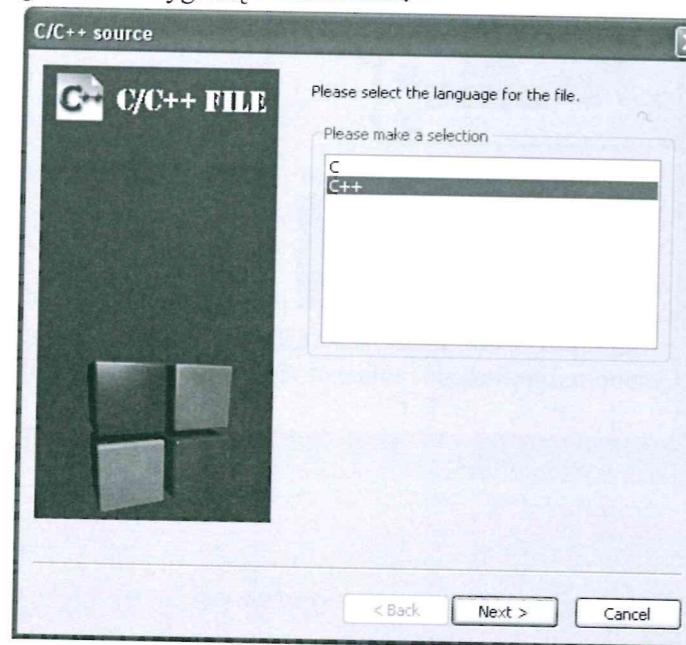


#### ④ Trečias žingsnis. Programos failo kūrimas.

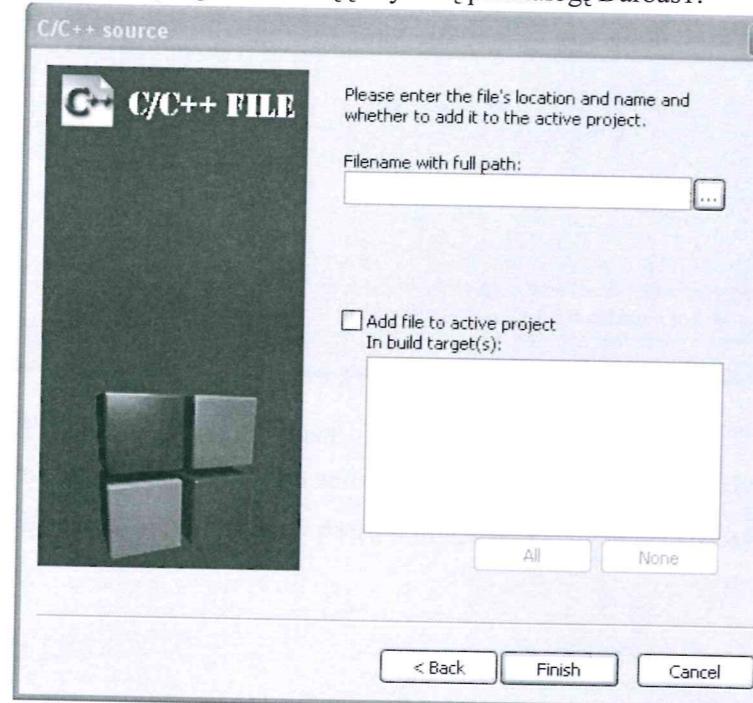
- Pagrindiniame meniu parinkite komandas: *File → New → File...* ir atsivérusiamie dialogo lange pažymėjė *C/C++ source* patvirtinkite pasirinkimą spragtelėdami mygtuką



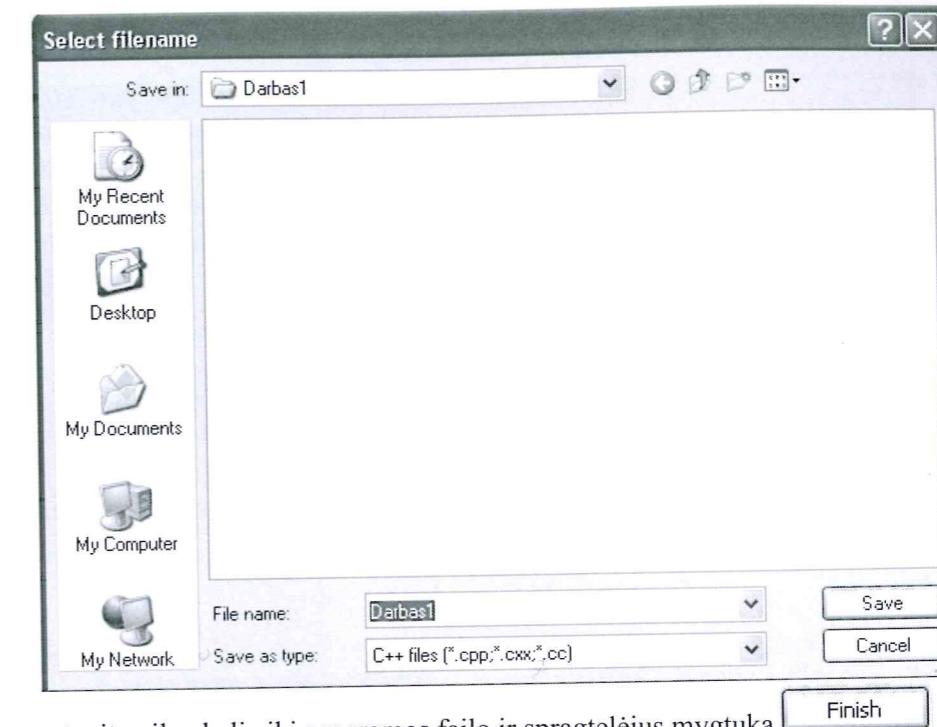
- Atsiveria dialogo langas, kuriame reikia pasirinkti programavimo kalbą. Pasirinkite C++ ir eikite į kitą etapą spragtelėdami mygtuką **Next >**



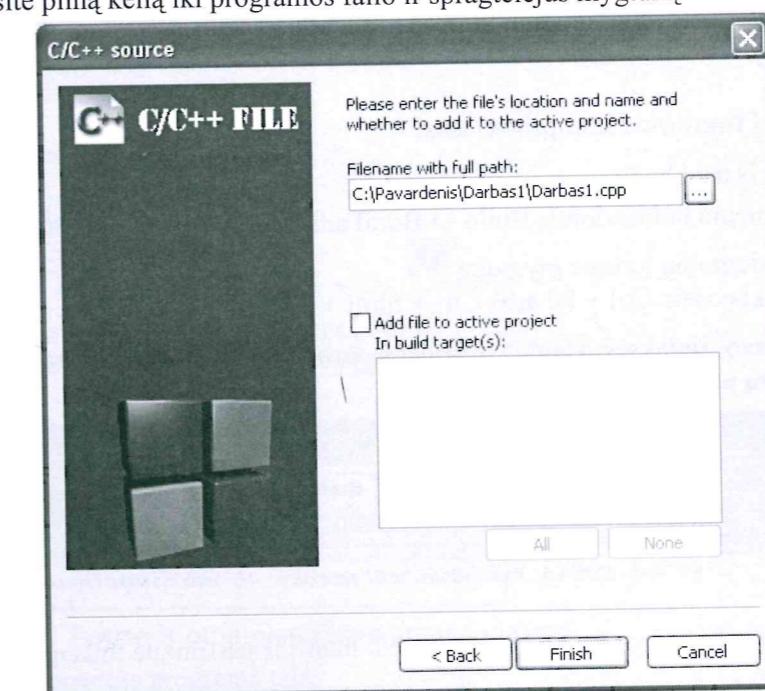
- Šiame etape kuriamos programos failą įrašysite į pakatalogį *Darbas1*:



Spragtelėjė ..., atsivérusiamė dialogo lange nurodykite katalogą *Darbas1*, o laukelyje *Name* įrašykite failo, kuriamė bus sukurta programa, vardą *Darbas1* ir spragtelékite mygtuką *Save*.



Dialogo lange matysite pilną kelią iki programos failo ir spragtelėjus mygtuką **Finish**



*CodeBlocks* aplinkoje atsivers kuriamos programos ruošinys, kurį galésite redaguoti *CodeBlocks* rengykle.

```
*Darbas1.cpp  X
1 #include <fcntl.h>
2 #include <iostream>
3 #include <iostream>
4
5 using namespace std;
6
7 int main ()
8 {
9     _setmode(_fileno(stdout), _O_U16TEXT);
10    wcout << L"Labas";
11
12 }
```

### ¶ Ketvirtas žingsnis.

Programos apibūdinimas. Programos failo įrašymas į katalogą Darbas1.

- Programos lange įterpkite dar vieną eilutę, kurioje yra komentaras:

```

1 // Darbas1. Pažintis su CodeBlocks aplinka
2 #include <fcntl.h>
3 #include <io.h>
4 #include <iostream>
5
6 using namespace std;
7
8 int main ()
9 {
10     _setmode(_fileno(stdout), _O_U16TEXT);
11     wcout << L"Labas";
12     return 0;
13 }

```

- Toliau pasirinkite vieną iš išvardintų pagrindinio meniu komandų:

*File → Save File;*

*File → Save File As...;*

*File → Save all files;*

*File → Save everything.*

Programą įrašyti taip pat galima priemonių juostos mygtukais arba sparčiaisiais klavišais *Ctrl + S, Ctrl + Shift + S, Alt + Shift + S*.

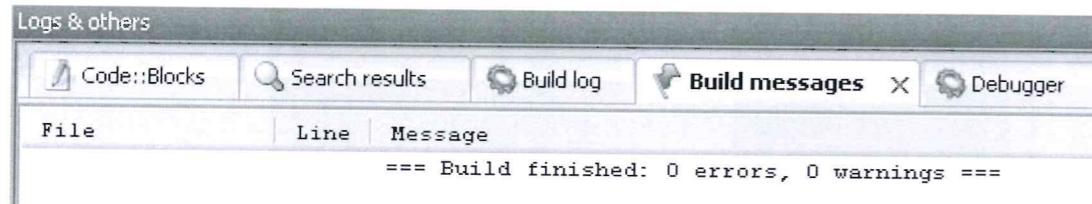
### ¶ Penktas žingsnis.

Programos kompliliavimas.

Tai galima atlikti vienu iš trijų būdų:

- pagrindinio meniu komandomis *Build → Build* arba *Build → Compile current file* ;
- programos priemonių juostos mygtuku ;
- sparčiaisiais klavišais *Ctrl + F9* arba *Ctrl + Shift + F9* .

Jeigu programeje nebuvu sintaksės klaidų, apatinėje darbo lango dalyje bus rodomas pranešimas apie sėkmingai sukompliliotą programą.



- Išbandykite visus tris programos kompliliavimo būdus ir pasirinkite tinkamiausią.

Tačiau, norėdami pamatyti programos darbo rezultatus, ją dar turite įvykdyti.

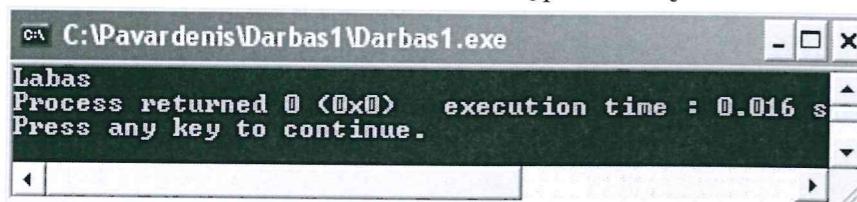
### ¶ Šeistas žingsnis.

Programos vykdymas.

Tai galima atlikti vienu iš trijų būdų:

- pagrindinio meniu komandomis *Build → Run*;
- programos priemonių juostos mygtuku ;
- sparčiaisiais klavišais *Ctrl + F10*.

Jeigu programeje nebuvu vykdymo klaidų, ekrane atsivers juodos spalvos programos naudotojo langas, kuriame išvysite žodį **Labas** ir informacinių pranešimų.



Norint redaguoti programą, pirmiausia reikia užverti rezultatų langą. Tai galima padaryti spustelėjus klaviatūros klavišą *Enter* arba spragtelėjus lango užvérimo mygtuką .

Prieš vykdant programą, kuri buvo redaguota, ji iš naujo sukompliliuojama.

- Išskleiskite darbo pradžioje sukurtą katalogą *Darbas1* ir pažiūrėkite, kiek ir kokių failų sukurta. Svarbiausias failas yra *Darbas1.cpp*, nes Jame yra programos tekstas. Ši failą reikia saugoti. Iš šio failo sukuriama kiti to paties pavadinimo failai *Darbas1* su skirtingais prievardžiais bei kiti pagalbiniai failai.



### ¶ Septintas žingsnis.

Teksto išvedimas į ekraną.

- Norint, kad ekrane šalia žodžio **Labas** būtų užrašytas ir jūsų vardas, programos teksto eilutę

*wcout << L"Labas";*

pakeiskite tokia:

*wcout << L"Labas. Mano vardas Vytautas";*

Atlikus pakeitimą, programa bus tokia:

```

//-----
// Darbas1. Pažintis su CodeBlocks aplinka
#include <fcntl.h>
#include <io.h>
#include <iostream>

using namespace std;

int main ()
{
    _setmode(_fileno(stdout), _O_U16TEXT);
    wcout << L"Labas. Mano vardas Vytautas";
    return 0;
}
//-----

```

Sukompiliuokite ir įvykdykite programą, t. y. pakartokite penktą ir šeštą žingsnius. Ekrane turėtumėte matyti:

**Labas. Mano vardas Vytautas!**

### ¶ Aštuntas žingsnis.

Teksto ir ornamento išvedimas į ekraną.

- Pakeiskite ankstesnę programą taip:

```

//-----
// Darbas1. Pažintis su CodeBlocks aplinka
#include <fcntl.h>
#include <io.h>
#include <iostream>
//-----
using namespace std;

int main ()
{
    _setmode(_fileno(stdout), _O_U16TEXT);
    wcout << L"*****";
    wcout << L"* Labas. Mano vardas Vytautas *";
    wcout << L"*****";
    return 0;
}
//-----

```

- Įvykdykite programą. Ekrane turėtumėte matyti:

Kaip pastebite, vykdant programą, kiekvienu wcout sakiniu, kuris baigiasi manipulatoriumi endl, ekrane atskirose eilutėse išvedamas tekstas, nurodytas tarp apostrofų. Manipulatorius endl perkelia žymeklį į naują srauto eilutę.

- Programoje ištrinkite endl.

```
wcout << L"*****-**-*****-*****-*****-*****-*****-*****-*****-*****";  
wcout << L"** Labas. Mano vardas Vytautas **";  
wcout << L"*****-**-*****-*****-*****-*****-*****-*****-*****-*****".
```

- Įvykdykite programą. Ekrane matysite:

\*\*\*\*\* Labas. Mano vardas Vytautas! \*\*\*\*\*

Pastebėjote, kad neparašius manipulatoriaus endl, užpildoma pirma eilutė (eilutėje telpa 80 ženklų), po to pradedama pildyti antroji.

- Pakeiskite programą taip, kad gautumėte tokį vaizdą:

¶ Devintas žingsnis. Darbo su CodeBlocks pabaiga.

Tai galima atlikti pagrindinio meniu komandomis *File → Quit*, sparčiaisiais klavišais *Ctrl + Q* arba spragtelėjus *CodeBlocks* lango užvérimo mygtuką .

- Įrašykite programą ir baikite darba su CodeBlocks